AI for real-world network operation

WP3– AI assisted human decision making

D3.1– AI4REALNET solutions to augment human decision-making

AI4REALNET has received funding from European Union's Horizon Europe Research and Innovation programme under the Grant Agreement No 101119527, and from the Swiss State Secretariat for Education, Research and Innovation (SERI).

DOCUMENT INFORMATION

DOCUMENT	D3.1– AI4REALNET solutions to augment human decision-making
TYPE	Document
DISTRIBUTION LEVEL	Public
DUE DELIVERY DATE	31/03/2026
DATE OF DELIVERY	24/03/2026
VERSION	V2.0
DELIVERABLE RESPONSIBLE	Delft University of Technology (TUD)
AUTHOR (S)	Clark Borst (Delft University of Technology)
	Giulia Leto (Delft University of Technology)
OFFICIAL REVIEWER/s	Herke van Hoof (University of Amsterdam)
	Sebastiaan De Peuter (University of Amsterdam)
	Marcello Restelli (Politecnico di Milano)

DOCUMENT HISTORY

VERSION	AUTHORS	DATE	CONTENT AND CHANGES
Version 0.1	Clark Borst	30/03/2025	Template and document structure
Version 0.2	Ricardo Bessa, Margarida Costa, Pedro Ferreira	31/01/2026	Uncertainty estimation and forecasting methods
Version 1.0	Ricardo Bessa, Margarida Costa, Pedro Ferreira, Clark Borst, Giulia Leto, Herke van Hoof, Manuel Renold, Julia Usher, Eduardo Vilches, Toni Wafler, Samira Hamouche, Alberto Castagna, Anton Fuxjaeger, Alberto Metelli, Kurt Brendlinger	25/02/2026	Partner contributions to full draft
Version 1.1	Clark Borst	02/03/2026	Finalize full draft, write Executive Summary
Version 1.2	Clark Borst, Ricardo Bessa, Herke van Hoof, Toni Waefler, Kurt Brendlinger, Alberto Metelli, Anton Fluxjaeger, Manuel Renold	19/03/2026	Revision based on internal reviewer comments
Version 2.0	Clark Borst	20/03/2026	Second draft

ACKNOWLEDGEMENTS

NAME	PARTNER
Margarida Costa	INESC TEC
Ricardo Bessa	INESC TEC
Hugo Cardante	INESC TEC
Carla Gonçalves	INESC TEC
Pedro Ferreira	INESC TEC
Clark Borst	TUD
Giulia Leto	TUD
Manuel Renold	FHNW
Julia Usher	FHNW
Toni Waefler	FHNW
Samira Hamouche	FHNW
Herke van Hoof	UvA
Alberto Metelli	POLIMI
Anton Fuxjaeger	enliteAI
Kurt Brendlinger	Fraunhofer IEE
Daniel Boos	SBB

DISCLAIMER

This project is funded by the European Union and SERI. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union and SERI. Neither the European Union nor the granting authority can be held responsible for them.

EXECUTIVE SUMMARY

Artificial Intelligence (AI) is reshaping the operation and management of complex socio-technical systems across many sectors of society. In critical network infrastructures—such as power transmission grids, railway systems, and air traffic management networks—AI offers substantial potential to improve efficiency, reliability, safety, and resilience in the face of growing system complexity and uncertainty. These infrastructures are increasingly characterized by high interconnectivity, dynamic operating conditions, distributed assets, and exposure to rare but high-impact events. While advanced AI methods can process vast data streams and optimize decisions at unprecedented speed, their deployment in safety-critical environments cannot rely on automation alone. Instead, it requires carefully designed human–AI collaboration frameworks that preserve human authority, enable meaningful supervision, support situation awareness, and foster calibrated trust. The AI4REALNET project responds to this need by developing an integrated, human-centered AI framework aimed at augmenting—not replacing—human decision-making in critical network operations.

Human-Centered Vision and Control Modes. AI4REALNET is grounded in the principle that trustworthy AI emerges from the interaction between algorithmic capabilities and human operational practice. High predictive accuracy or optimization performance alone does not guarantee effective deployment. Rather, AI systems must be embedded within workflows in ways that align with human cognitive processes, professional responsibilities, and institutional accountability structures. To operationalize this vision, the project defines three complementary control modes: (i) AI-assisted full-human control, where AI provides structured recommendations, risk indicators, and scenario analyses while the operator retains full decision authority; (ii) interactive and human–AI co-learning, where both AI systems and operators jointly make decisions and adapt over time through feedback and exploration; and (iii) trustworthy full-AI-based control under human supervision, where AI agents execute decisions autonomously within clearly defined oversight and directive constraints. These modes are conceived as configurable settings rather than fixed categories, allowing organizations to tailor the degree and form of autonomy to task criticality, uncertainty levels, regulatory requirements, and operational context. The central insight is that trustworthiness is a systemic property of the human–AI configuration rather than of the algorithm in isolation.

Uncertainty-Aware Decision Support. Uncertainty is inherent in the operation of critical infrastructures. Fluctuating demand patterns, equipment degradation, renewable energy variability, unexpected disruptions, and incomplete sensor data all contribute to decision-making under uncertainty. AI4REALNET therefore treats uncertainty quantification and communication as foundational com-

ponents of trustworthy AI. The project distinguishes between epistemic uncertainty, which reflects knowledge gaps and out-of-distribution states, and aleatoric uncertainty, which captures irreducible stochastic variability in system behavior. In this deliverable, AI4REALNET contributes methods to estimate, separate, and communicate these uncertainties—through reliability indicators, probabilistic forecasts, failure risk estimation, and uncertainty intervals. This transparency enables operators to better assess the robustness of AI recommendations, identify fragile predictions, and make informed judgments about when to rely on automated suggestions, when to seek additional information, and when to intervene. In this way, uncertainty modeling becomes a mechanism for calibrated trust and responsible deployment rather than a purely technical add-on.

Multi-Objective Reasoning and Trade-Off Transparency. Decisions in infrastructure management typically involve balancing multiple competing objectives, including safety margins, service reliability, economic efficiency, environmental sustainability, and regulatory compliance—and the correct course of action may change dynamically depending on the individual situation. Traditional AI optimization approaches often reduce these competing criteria to a single scalar objective through fixed weightings, thereby obscuring underlying trade-offs. In this deliverable, AI4REALNET advances multi-objective reinforcement learning and optimization techniques—via a domain-agnostic toolset—that generate Pareto-optimal solution sets, explicitly revealing the structure of objective conflicts and synergies. By presenting alternative solutions that represent different trade-off configurations that can change depending on the situation, the system enables operators to select or steer outcomes according to situational priorities and stakeholder constraints. This transparent handling of trade-offs supports mixed-initiative decision-making, enhances alignment between AI outputs and human intent, and reduces the risk of unintended optimization bias. Multi-objective reasoning thus provides a principled interface between algorithmic optimization and human value judgment in safety-critical environments.

Interactive and Co-Learning Architectures. Human–AI collaboration is inherently dynamic. Operator expertise evolves over time, organizational practices adapt, and rare or unforeseen events reveal gaps in both human mental models and AI training distributions. AI4REALNET therefore emphasizes interactive and co-learning architectures that enable sustained, bidirectional adaptation. On the AI side, AI4REALNET delivers algorithms that are designed to incorporate explicit operator feedback, implicit behavioral signals, preference information, and human reward structures (via inverse reinforcement learning) into learning processes. On the human side, operators are supported through scenario exploration tools, explanation interfaces, and structured visualizations that enhance understanding of alternative strategies and system dynamics. The architecture integrates feedback processing modules, optimization engines, explanation components, and persistent state management to ensure continu-

ity across sessions. Through this design, the AI system becomes not merely a recommendation engine, but a collaborative partner in reasoning, exploration, and reflection. Such long-term interaction supports competence development, trust calibration, and improved preparedness for rare high-impact events. Beyond algorithmic advancements, AI4REALNET also delivers design and analysis frameworks to elicit interface requirements for supporting human understanding, engagement and learning.

Agent-As-A-Service and Deployment Architecture. Effective deployment of AI in safety-critical infrastructures requires more than algorithmic innovation; it demands robust integration architectures that ensure inspectability, accountability, and controllability. AI4REALNET delivers the domain and agent-agnostic Agent-As-A-Service (A3S) architecture as a modular integration framework that encapsulates autonomous agents within a human-centered service layer. This abstraction exposes recommended actions together with uncertainty estimates, contextual information, and traceable decision pathways. It supports adjustable autonomy levels, enabling operators to increase or decrease the degree of automation depending on context and confidence. The service-oriented design facilitates auditing, logging, and what-if analysis, ensuring that AI-driven processes remain transparent and accountable. By transforming autonomous optimization into supervised decision-support services, A3S provides a scalable and domain-agnostic pathway for embedding AI into operational workflows while preserving meaningful human oversight.

Trustworthy Autonomous Operation and Future Directions. In certain operational contexts—such as high-frequency control tasks or time-critical responses—AI-driven autonomy may be necessary to meet performance requirements. AI4REALNET defines such autonomy as trustworthy only when embedded within structured supervisory mechanisms that preserve human authority and strategic control. In this deliverable, AI4REALNET contributes a concrete realization of trustworthy autonomy through a proposed *director system*, in which human operators remain actively involved by issuing high-level directives that guide autonomous execution. It introduces interpretable primitives derived from hierarchical task analysis to structure human–AI interaction as a transparent and controllable process rather than passive monitoring. Furthermore, it presents a scalable multi-agent reinforcement learning architecture for railway operations, including graph-based environment representations, negotiation-based coordination mechanisms, and a supervision layer that maintains system stability. Together, these elements demonstrate how autonomous AI can be embedded into operational workflows while preserving meaningful human control, transparency, and adaptability.

Overall, AI4REALNET demonstrates that augmenting human decision-making in critical network infrastructures requires a deliberate shift toward human-centered AI. By combining uncertainty-aware reasoning, transparent multi-objective optimization, interactive co-learning, and supervised autonomy

within a coherent architectural framework, the project establishes a foundation for resilient, adaptive, and trustworthy human–AI systems. The contributions presented in this deliverable collectively support the three control modes: they provide uncertainty-aware and interpretable decision support for AI-assisted human control; enable interactive feedback and co-learning for joint human–AI adaptation and decision-making; and realize modular, multi-agent architectures with directive control and supervision for trustworthy autonomous operation. In this way, AI4REALNET's integrated approach ensures that technological progress enhances human expertise, strengthens accountability, and supports long-term operational excellence in safety-critical domains.

At the same time, AI4REALNET acknowledges an important limitation: although the proposed architectures, interaction mechanisms, and algorithmic principles generalize well across domains, the learned models themselves—particularly those based on reinforcement learning—remain largely environment-dependent. Consequently, transferring trained policies across different infrastructures, system dynamics, or directive configurations remains challenging and typically requires retraining or targeted adaptation.

Finally, the methods, architectures, and algorithms presented in this deliverable should be considered as initial realizations of the AI4REALNET approach. Some of them will be further refined in subsequent work (e.g., Deliverable D3.2) and/or evaluated in human-in-the-loop settings (Work Package 4) to validate their effectiveness and expected benefits across the proposed control modes and operational use cases defined in Work Package 1.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The AI4REALNET concept centers on optimizing the degree and form of AI-driven decision support for human operators, with the goal of creating the most effective human–AI team rather than merely deploying automated assistance. To achieve this, the project will analyze the level of human involvement required across different tasks and subtasks in critical infrastructures, evaluate human decision-making performance and limitations, and identify where human cognitive capacities offer strengths—or where they may introduce risks.

A growing body of research in human–AI interaction and teaming highlights that optimal performance arises not from replacing humans, but from designing complementary partnerships between human judgment and algorithmic capabilities. Studies in fields such as aviation, power systems, transport, and emergency management show that humans excel at contextual reasoning, ethical judgment, handling ambiguity, and adapting to unfamiliar situations, while AI systems excel at scalability, pattern recognition, multi-objective optimization, and continuous monitoring. Literature on mixed-initiative systems, interactive machine learning, and human-centered AI further emphasizes the importance of aligning AI assistance with operator intent, maintaining transparency, and ensuring that the human remains engaged and capable of taking over control when needed. Poorly calibrated automation—either too intrusive or too passive—has been shown to decrease trust, increase cognitive workload, and reduce overall system performance.

Building on these insights, AI4REALNET aims to identify where operators can most effectively benefit from AI assistance, where interactive and co-learning processes are advantageous, which tasks may be delegated to autonomous AI, and where skilled human oversight remains essential. To structure this, AI4REALNET has focused on three complementary control modes outlined in deliverables D1.1, D2.1 and scientific publications (Mussi et al., 2025; Leyli-abadi et al., 2025):

- full-human control supported by AI-based decision assistance,
- interactive and human–AI co-learning for shared decision-making, and
- trustworthy full-AI control supervised and directed by human operators.

These modes provide a coherent framework for developing human-in-the-loop decision-support solutions for critical large-scale infrastructures. The control modes are embedded within the overarching AI4REALNET human–AI collaborative system architecture, shown schematically in Figure 1. The figure illustrates how each AI4REALNET Work Package (WP) contributes to this architecture. This document specifically describes the building-block contributions of WP3 in the realization of the three complementary control modes within the AI4REALNET collaborative architecture.

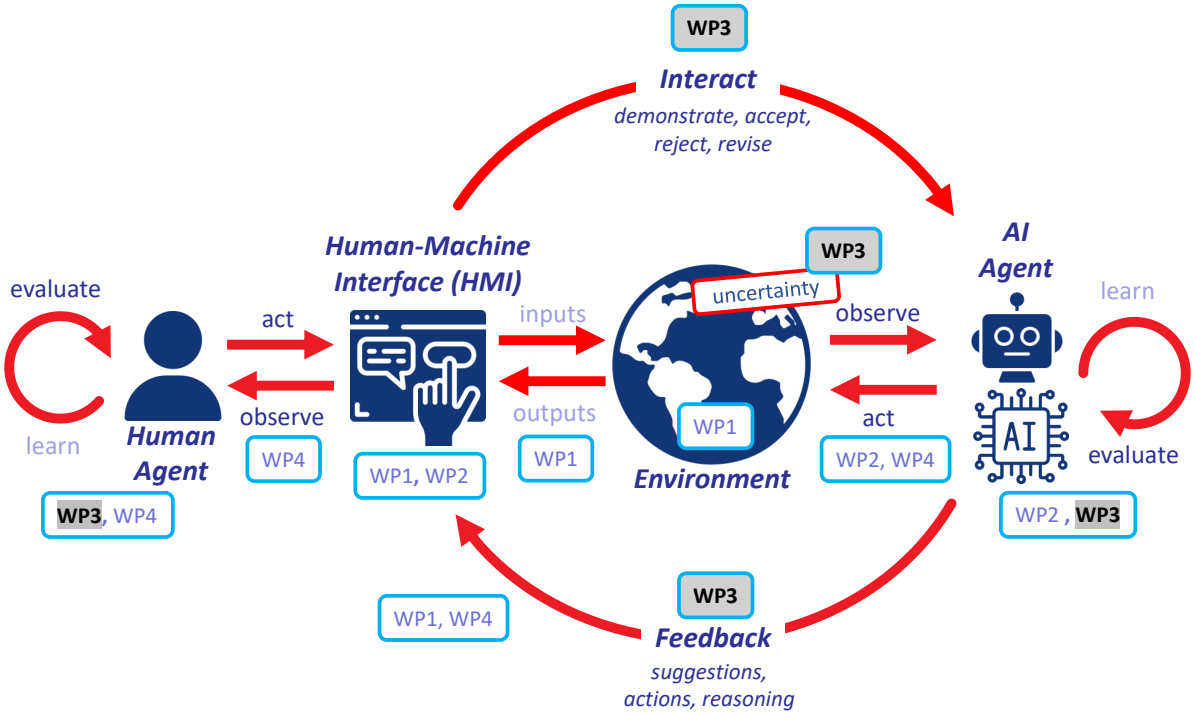


FIGURE 1 - GRAPHICAL OVERVIEW OF AI4REALNET, SHOWING HOW THE WORK PACKAGES (WP) ARE POSITIONED WITHIN THE ENVISIONED HUMAN–AI COLLABORATIVE CONTROL LOOP.

1.1. (AI-ASSISTED) FULL-HUMAN CONTROL

Full-human control refers to operational modes in which the human operator retains primary decision authority, while AI systems act strictly as supportive tools that enhance situation awareness, reduce workload, and surface actionable insights. In this setting, algorithms are designed to respect human agency, provide transparent and interpretable recommendations, and ensure that the operator remains at the center of the decision loop.

Within AI4REALNET, AI systems in full-human control modes focus on continuous monitoring of system states, detecting anomalies, and highlighting situations where attention or intervention may be required. Rather than executing actions autonomously, algorithms should generate well-calibrated recommendations, together with their expected impact on relevant KPIs, allowing the operator to make informed decisions. To be trustworthy, these recommendations must be accompanied by clear indications of uncertainty and risk, for example, through probabilistic modeling or confidence-based estimates.

A key design principle adopted in AI4REALNET is to support, rather than override, human judgment. This requires algorithms that can adapt their level of assertiveness depending on system conditions and estimated operator cognitive load. Literature on adaptive automation and mixed-initiative systems stresses that such adaptation helps prevent overreliance on automation, avoids loss of situation

awareness, and maintains an effective trust balance.

From a methodological perspective, AI components should combine multiple sources of evidence–such as historical case retrieval, model-based scenario rollouts, and multi-objective optimization–to ensure that recommendations are robust and aligned with operator goals. Reinforcement learning, supervised learning, and simulation-based verification can all contribute to generating candidate actions whose implications are easy for the operator to understand and evaluate.

Crucially, uncertainty communication is considered in AI4REALNET as a first-class requirement. Algorithms should not only estimate their own confidence but present it in a way that supports human decision strategies: for instance, highlighting which options carry high risk, which rely on uncertain model predictions, or which align closely with prior successful actions. The overarching goal is to design AI assistants that strengthen operator capability and decision quality, while preserving full human control and fostering long-term, calibrated trust.

1.2. HUMAN-AI INTERACTION AND CO-LEARNING

Interactive and co-learning modes place humans and AI systems in a shared decision-making loop in which both parties jointly make decisions and adapt to one another over time. In these modes, the AI does not simply issue recommendations; instead, it actively interprets human intentions, responds to corrections, and updates its internal models based on ongoing interaction. The aim is to create a collaborative dynamic in which humans guide the AI's behavior while simultaneously gaining insights from the system's reasoning and exploration capabilities.

From a design perspective, AI4REALNET algorithms supporting interaction and co-learning must be able to infer user goals, even when these goals are only partially expressed or evolve over time. When operators adjust parameters, reject suggestions, or modify proposed plans, the AI should treat these actions as meaningful signals. This requires models that can integrate explicit feedback (direct corrections or preference indications) with implicit behavioral cues derived from user interactions with the interface. Algorithms should also be able to offer consistent completions of partially specified human plans, ensuring that manual interventions lead to coherent system-level solutions.

In a co-learning relationship, adaptation flows in both directions. AI4REALNET AI systems should continually refine their understanding of human preferences, decision heuristics, and typical trade-offs in multi-objective problems. Conversely, the human operator can benefit from exposure to alternative strategies, explanations of AI reasoning, scenario-based comparisons, and rapid exploration of "what-if" variations. The design goal is to support mutual learning: the AI becomes better aligned with human goals, and the human gains a deeper understanding of system behavior and possible solution pathways.

Achieving this requires algorithms that are robust to mixed-initiative control—situations where ac-

tions may be chosen partly by the human and partly by the AI. Learning processes must be flexible enough to incorporate human-generated data, preference signals, or corrections without destabilizing the underlying policy. Approaches such as inverse reinforcement learning, preference modeling, and explainable AI are particularly suited to this, as they help systems infer underlying human reward structures and expose the rationale behind AI choices.

Overall, AI4REALNET algorithms designed for interactive and co-learning settings should prioritize interpretability, adaptability, and responsiveness. Their objective is not only to optimize system performance but to cultivate a collaborative process in which human operators and AI systems learn from one another, jointly improving decision quality in complex, safety-critical environments.

1.3. TRUSTWORTHY FULL AI-BASED CONTROL

Trustworthy full-AI control refers to operational modes in which AI systems take primary responsibility for decision-making and action execution, while human operators retain a supervisory and directive role. This mode is essential in complex, high-risk infrastructures where real-time demands, network scale, or system complexity make continuous human control impractical. In such cases, the design challenge is not to remove humans from the loop, but to ensure that autonomous decisions remain transparent, aligned with human objectives, and accountable to human oversight.

AI4REALNET algorithms operating under full-AI control must be capable of *autonomously* navigating dynamic, multi-agent environments, coordinating with other agents, and optimizing both local and global objectives, since these decisions are no longer mediated by human oversight as in the previous settings. Multi-agent reinforcement learning, developed in WP2, is particularly relevant here, as it supports decentralized decision-making, communication between agents when appropriate, and emergent cooperation driven by shared goals.

Although AI operates independently, human supervision remains a core requirement. In AI4REALNET, AI must provide interpretable explanations of its intentions, actions, and predicted outcomes to enable effective human validation. This requires integrating explainability and transparency mechanisms directly into the algorithmic design so that operators can understand why a decision was taken, detect deviations from expected behavior, and intervene when necessary. Clear communication of uncertainty, anomaly detection, and risk levels is essential for maintaining human trust and ensuring safe system operation.

In the AI4REALNET vision, human expertise should remain reflected in autonomous policies. Algorithmic frameworks should therefore incorporate mechanisms to learn from demonstrations, historical operator decisions, and expert-guided preferences. This allows autonomous agents to internalize domain knowledge and safety constraints, providing a foundation for behavior that is both competent and predictable to human supervisors.

Finally, trustworthy autonomy demands robust monitoring, auditing, and logging infrastructures. Algorithms should generate structured, traceable decision records that allow operators to analyze system performance, investigate anomalies, and understand failure mechanisms. These accountability measures are crucial in critical infrastructures, where transparency is not optional but central to operational safety, regulatory compliance, and public trust.

In summary, AI4REALNET trustworthy full-AI control does not imply removing humans from the system; instead, it emphasizes designing autonomous agents whose decisions are intelligible, aligned with human intent, and always subject to meaningful human direction and oversight.

1.4. STRUCTURE OF THE REPORT

This report is organized into nine main chapters that progressively develop the conceptual and technical foundations for human-centered and trustworthy AI-supported decision-making. **Chapter 2** discusses the transition from purely autonomous systems toward human-centered AI, highlighting challenges related to solution styles, trust, cognition, and alignment between human and AI objectives, as well as mechanisms for joint decision-making. **Chapter 3** presents the Agent-as-a-Service (A3S) concept as an architectural layer for exploring simulated futures and interactive decision support, including the associated graphical components to portray on human-machine interfaces as information overlays supporting human-AI interaction and understanding. **Chapter 4** addresses risk and uncertainty in AI-assisted decision-making, providing a literature review and introducing modeling approaches for forecasting agent failures, system uncertainty, and risk using evidential and conformal techniques. **Chapter 5** focuses on multi-objective decision-making with AI, covering theoretical foundations, algorithm design, and implementation within the AI4REALNET multi-objective framework. **Chapter 6** explores interactive AI approaches to augment human decision-making, including human–AI co-learning, inverse reinforcement learning, preference-based learning, and real-time interactive preference learning. It also examines how humans learn from AI and how co-learning concepts can be operationalized through dedicated overlays and extensions on human–machine interfaces developed within WP1 and WP2. **Chapter 7** then considers fully autonomous AI-driven decision systems, describing architectural requirements, system design, and negotiation mechanisms between agents and operational constraints. Finally, **Chapter 8** discusses the generalization and transferability of the proposed methods across domains and network-structured systems, while **Chapter 9** concludes the report by summarizing key findings and outlining future research directions.

Together, these chapters provide a comprehensive overview of the methodologies, algorithms, and tools developed within AI4REALNET to enable human-centered, interactive, and trustworthy AI in complex decision environments, thereby supporting the realization of the three complementary control modes within the AI4REALNET collaborative architecture.

2. FROM AUTONOMOUS TO HUMAN-CENTERED AI

As explained in Section 1, there are several ways in which AI approaches developed in the autonomous setting can be used to support human-centered modes, such as AI acting as decision support or joint decision-making, as in the co-learning setting. However, this transition also introduces challenges, many of which translate to the activities from AI4REALNET Work Package 3. This section discusses the challenges involved in enabling autonomous AI approaches to support human-centered modes of operation, and outlines what potential solutions–or mitigation strategies–could look like.

2.1. CHALLENGES FROM DIFFERING SOLUTION STYLES

When human and AI solution styles differ, and humans follow the AI suggestions some, but not all of the time, the system might end up in states unfamiliar to the AI (resulting in low-quality AI recommendations) or unfamiliar to the human operator (resulting in low-quality human decisions). This can also affect explanations that provide expected future outcomes if they assume the AI recommendations are always followed (e.g., (Khan et al., 2009; Yau et al., 2020)).

Possible ameliorations to the problem of low-quality AI recommendations in unfamiliar states include training the AI on a larger state distribution to reduce the number of unfamiliar states and using quantification of epistemic uncertainty (discussed in Section 4 and developed in Task 3.1) to detect where the AI lacks coverage. The user interface could then display a confidence score indicating the AI's confidence in its recommendation.

Possible amelioration to the problem of reaching states unfamiliar to the operator would be first to recognize when such states are reached (e.g., by using evidential networks (developed in Task 3.1) to estimate the operator's uncertainty, or by measuring the operator's cognitive state (studied in Task 2.3). The AI could then be trained to avoid such states, e.g., by training with a human in the loop, or by inferring the human's goals or strategy (Tasks 3.2 and 3.3).

The problem of explanations that assume the AI's suggestions will always be followed could be mitigated by considering a range of policies during training of the explanation module, or by inferring the operator's strategy (Task 2.3).

2.2. CHALLENGES FROM MOTIVATION AND TRUST

For an operator to make productive use of AI support, the operator needs to be motivated to engage with the AI system and to appropriately trust the system. Here, both inappropriately high trust leading

to over-reliance, and inappropriately low trust leading to a disregard for the AI system, can be detrimental to decision-making quality. As is discussed in detail in Section 3.2 of (Bessa et al., 2024), solely providing recommendations (with or without explanations or an interpretable model) is insufficient to engage the user and to establish appropriate trust (Miller, 2023). In particular the AI solution should aim for calibrated trust rather then just increased trust (Endsley, 2023). In general, a joint decision scenario is thus preferred over a human-in-control scenario where the agent solely provides recommendations.

Better results are anticipated if the AI acts in a 'cognitive forcing' or 'evaluative AI' manner. In these scenarios, the initiative is with the operator to make an initial decision or a hypothesis. The role of the AI agent is then to provide recommendations and explanations regarding the decision, or to provide evidence for or against the hypothesis.

To support these styles, autonomous AI could be trained to provide multiple suggestions and its internal value function could be leveraged to evaluate decisions suggested by the user. Values indicating the AI's confidence in its decisions might aid in establishing appropriate trust (Task 3.1). Going beyond this, AI agents could support the human cognitive processes of decision-making, learning, and motivation in co-learning scenarios (Tasks 3.3 and 3.4).

2.3. CHALLENGES FROM COGNITIVE ASPECTS

It is important that an AI-based decision support system supports human decision-making and learning processes. Again, based on the earlier discussion in Section 3.2 of (Bessa et al., 2024), solely providing recommendations (even with explanations or an interpretable model) falls short of this goal. Methods based on 'cognitive forcing' or 'evaluative AI' are here, too, more promising (Miller, 2023). This suggests a joint decision-making scenario rather than the *human-in-control* scenario. Furthermore, it is important that the operator can understand and process the information provided by an AI agent. While information from an AI agent can be helpful, an overload of information should be avoided, especially when the operator is under high stress or a high cognitive load. To also cope with rare critical events with high impact in safety-critical infrastructures, we need to ensure that situational awareness is not reduced (Endsley, 2023).

Again, this means that AI agents need to support the human cognitive processes of decision-making and learning, as studied in Tasks 3.3 and 3.4. The AI could also take into account the operator's cognitive state by measuring their stress and cognitive load using biosensors (Task 2.3). However, care must be taken to ensure this does not lead the operator to experience external control, as this will decrease motivation.

2.4. CHALLENGES FROM HUMAN-AI ALIGNMENT

AI agents are developed to perform tasks in a certain way, implicitly or explicitly assuming a certain goal or objective. For example, reinforcement learning approaches require an explicit reward function that describes the objective to be maximized. However, real tasks often require a careful trade-off of different objectives, risks, and stakeholders' interests. Human operators will make these trade-offs, perhaps in a way that is hard to quantify explicitly. There is a risk that the precise objective is not fully aligned between the human operator and the AI agent, leading to AI recommendations or other outcomes that do not adequately support the operator's goals.

An amelioration of these challenges can come from two perspectives. First, AI agents could explicitly consider the various objectives at play, which need to be traded off, resulting in a multi-objective approach where the operator is involved in making the final trade-off ((Hayes et al., 2022), Task 3.2). Second, the human goals or objectives could be inferred from data, for example, through inverse reinforcement learning (Ng and Russell, 2000). These inferred goals or objectives could then be used to train the AI model and/or as input for the AI's behavior (Task 3.3).

2.5. SUPPORT FOR JOINT DECISION MAKING

As noted several times in this section, to appropriately support human decision-makers, a joint decision making is required where AI and human operator share the responsibility for decisions. For humans to be able to take and share responsibility, we need to go beyond recommendations, and thus beyond the functionality provided by the autonomous agents from Work Package 2. Additional functions are required to support human exploration of the decision process and mirroring of the quality of human operator actions.

Such additional functionalities could include the ability to explore 'what if' scenarios (e.g., using the simulation tools developed in Work Packages 1 and 2, and the interactive AI interface from Task 1.3), answering specific operator queries and providing evidence for and against operator hypotheses, adapting to constraints provided by the user (e.g., using safe AI methods from Task 2.3), and evaluating human decisions. AI agents that support such human decision-making and learning processes are studied in Tasks 3.1-3.4.

3. AGENT-AS-A-SERVICE

The AI4REALNET project is structured to advance both autonomous and AI-supported decision-making in critical infrastructures. The project has defined six use cases across three domains: railway, power grid, and air traffic control. Beyond improving the performance of autonomous controllers and AI-based support systems, AI4REALNET directly addresses the gap between these two modalities by turning autonomous models into systems that remain interpretable and steerable by human operators, so that their internal reasoning and assumptions can be examined rather than merely inferred from their outputs.

To this end, the project develops model-independent tools that provide transparency to otherwise opaque agents, revealing decision pathways, limitations, and failure modes in a consistent way. To fulfill this goal, AI4REALNET developed the concept of *Agent-as-a-Service* (*A3S*), in which a human-centered layer wraps existing agents, extending agents' recommendations with associated uncertainty, risks, and relevant context. Additionally, *A3S* supports configuring how agent recommendations are used in practice (e.g., advisory use, operator overrides, and interactive simulation-based checking), allowing human operators to retian control over final decisions. *A3S* allows operators to understand not only what the system suggests, but also why and how strongly.

A3S is a domain-agnostic toolkit that re-purposes autonomous AI for accountable, human-guided decision processes. Uncertainty estimates are exposed at each decision step, so operators are aware of AI limitations, turning full autonomy into a supervised process where humans can assess both the recommendation and the reliability behind it.

Within this framework, operators gain the ability to inspect the model's reasoning, probe alternative inputs and scenarios, and recognize when confidence collapses or a distributional shift appears. The *Agent-as-a-Service* abstraction preserves the efficiency and speed of autonomous systems, while ensuring that the human remains the informed authority rather than a passive recipient of the model's output. In doing so, AI4REALNET anchors decision-making in transparency, controllability, and accountability, in line with a European vision of human-centered AI (European Union, 2024).

The *A3S* architecture is depicted in Figure 2. *A3S* is designed as a modular and extensible approach for encapsulating AI agents within service-oriented interfaces suitable for integration with the HMIs developed in the AI4REALNET project. At its core, the service combines a simulation environment and an AI agent, which together form the queryable intelligence layer. Communication between this core and operators through an HMI or external systems is mediated via well-defined endpoints, such as *restore*, *simulate*, and *get_action_space*, exposing essential actions and internal states in a flexible manner.

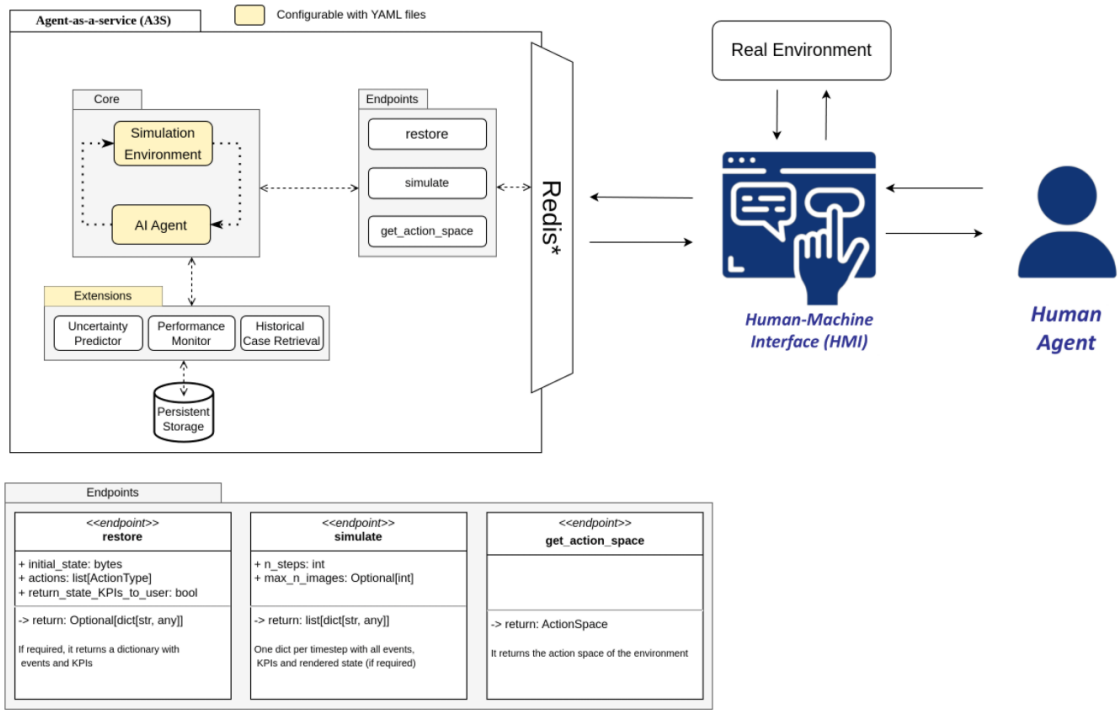


FIGURE 2 - *A3S*: ARCHITECTURE OF THE SERVICE.

An important feature is the ease with which extensions can be integrated. For example, the uncertainty predictor (using the algorithms devised in Section 4) can continuously assess and report the agent's confidence at every decision point. This allows human operators to recognize when the agent's recommendations are robust, uncertain, or potentially affected by changes in context or data. By surfacing confidence information, the framework supports transparent decision-making, which strengthens human-centered oversight.

The architecture also allows integration of additional services, such as performance monitoring and historical case retrieval. These services connect to persistent storage for reliable tracking, traceability, and accountability. All modules, including the uncertainty predictor, can be dynamically configured using YAML files. This flexibility ensures that the architecture is domain-agnostic and easily adaptable. The endpoints allow for direct inspection of states, actions, and key performance indicators (KPIs), and can include uncertainty information alongside states, actions, and KPIs. As a result, human operators can base their decisions on both system outputs and a clear quantification of reliability.

The *A3S* service is designed to communicate asynchronously with external systems and operator tools. This supports interactive supervision and intervention in simulated roll-outs and replay settings, where operators can test alternatives before committing to a decision. Operators are thus empowered to interpret, challenge, and guide AI behavior using a full set of integrated, uncertainty-aware services. This architecture delivers the standard of transparency and accountability required for critical decision-making in infrastructure operations.

The framework bridges the divide between full autonomy and human-centered control, empowering operators to oversee, interpret, and guide AI-driven processes in real-time. By integrating uncertainty estimation and auxiliary services within a modular, domain-agnostic architecture, AI4REALNET sets a benchmark for trustworthy, resilient, and accountable decision systems, answering the long-term needs for safety, digitization, and sustainable innovation in Europe's essential networks.

3.1. *A3S* AS A ROLL-OUT LAYER FOR SIMULATED FUTURE EXPLORATION

In AI4REALNET, the contribution of *A3S* is primarily situated on the *roll-out* side: rather than changing how agents are trained, *A3S* provides the intermediate layer required to *use* trained WP2 agents in a human-in-the-loop decision-making process. Concretely, *A3S* wraps an existing agent together with a simulator/digital environment and exposes a small set of service endpoints that make the agent's decision process *inspectable* and *queryable* at runtime. This enables operators to interact with the agent not only at the current time step, but also by probing *possible futures* through short-horizon simulation and KPI evaluation.

A key design choice is that the service returns a *single* recommended action per decision step (i.e., the next action the policy would take in the current state), which is then augmented with uncertainty and context information such as KPIs. While this does not provide a full plan or ranked list of alternatives, it supports a practical interaction paradigm in operational settings: operators can treat the agent's next action as a *candidate intervention*, and use simulation to explore how this choice propagates into near-term outcomes.

3.1.1. EXPLORING THE SIMULATED FUTURE WITH *A3S*

A3S enables a workflow in which the simulator is used as a lightweight verification and exploration mechanism, tightly coupled to the agent's recommendation. In the AI4REALNET setting, we consider three complementary usage patterns:

1. **Continuous short-horizon look-ahead with KPI-based alarms.** Using the *simulate* endpoint, the current operational state can be rolled forward for a small number of steps (e.g., H steps) by repeatedly applying the agent's single-step recommendations. The resulting rollout produces predicted KPI trajectories (e.g., performance, safety or rule-violation indicators). These predicted KPIs can be compared to predefined thresholds to *raise alarms* when the near-term outlook is undesirable. This pattern supports proactive monitoring: rather than waiting for KPI degradation to materialize, operators can be warned when the simulated future indicates a high likelihood of degradation, or when uncertainty becomes sufficiently large to warrant attention.

2. **Operator "what-if" exploration with agent continuation.** Operators can use *A3S* to assess the consequences of a *manual* decision at the present time. Specifically, an operator can apply an action A' in the current state and request a forward simulation in which the first step is fixed to A', after which the policy resumes by producing its standard single-step recommendations. The resulting rollout allows operators to answer questions of the form: *"What happens if I do action A' now, and then let the agent continue from there?"* This supports decision making under uncertainty by allowing rapid comparison of near-term KPI consequences, without requiring the operator to manually specify a long sequence of subsequent actions.

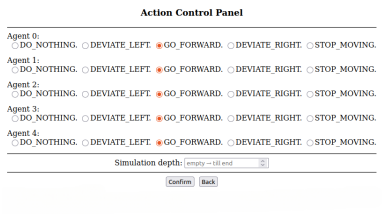
3. **Post-hoc counterfactual exploration on recorded trajectories.** Beyond live operation, the same interface supports retrospective analysis. From a recorded rollout, operators or analysts can restore a historical state using *restore* and then simulate counterfactual branches that correspond to alternative actions at a chosen decision point. This enables structured "what-if" analysis after the fact, supporting debugging, audit-oriented reflection, and improved understanding of when and why a given decision led to an observed outcome. Importantly, this counterfactual use relies on the same service contract as live exploration, ensuring that offline analysis remains consistent with the deployed system behavior.

Across these patterns, uncertainty estimates (cf. Section 4) can be exposed alongside the simulated KPI trajectories. For example, it can be setup that such that H timesteps are simulated into the future for every action proposed. For this horizon H KPIs and uncertainty estimates are calculated and estimated such that they can be fed into the control system: when the agent's confidence collapses, or when predicted KPIs indicate risk, the system can provide a clear signal to the operator that additional scrutiny is required. Conversely, when simulated KPI outcomes are stable and uncertainty is low, the system can reduce cognitive load by presenting a compact summary (recommended action, predicted KPI trend, and uncertainty), rather than requiring deep inspection of raw state variables.

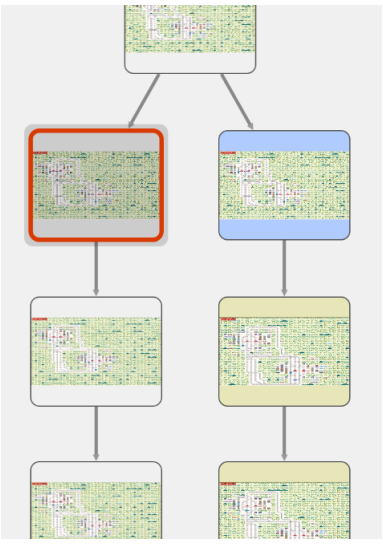
3.1.2. SCOPE AND CURRENT LIMITATIONS

The current *A3S* implementation focuses on single-step action recommendation coupled with forward simulation and KPI evaluation. This design is intentionally lightweight and compatible with existing WP2 agents, but it also implies that the system does not (yet) provide ranked action sets, or policy ensembles by default. Nevertheless, by combining single-step recommendations with operator-selected interventions and short-horizon rollouts, *A3S* still supports rich exploration of potential futures through repeated simulate–inspect cycles.

(a) INITIAL STATE OF THE TRAJECTORY.



(b) ACTION CONTROL PANEL FOR HUMAN INPUT.



(c) UPDATED GRAPH.

FIGURE 3 - *TRACERL* - EXAMPLE OF HUMAN–AGENT INTERACTION IN *TRACERL*.

3.2. TRACERL AS A LAYER FOR INTERACTIVE *A3S*-BASED SIMULATION

To make *A3S*-based simulation and counterfactual exploration accessible to operators, AI4REALNET integrates the *A3S* backend within an interactive user interface overlay developed in Task 2.3, referred to as *TraceRL*. TraceRL consumes the *A3S* endpoints to (i) query the admissible actions (*get_action_space*), (ii) simulate forward from a selected state with or without human input (*simulate*), and (iii) restore previously recorded states (*restore*). This enables operators to explore short-horizon future outcomes and KPI changes by running targeted "what-if" simulations at specific decision points.

Figure 3 illustrates an example of human input in *TraceRL*. First, a trajectory is selected and visualized. The user then selects a decision block to override one or more actions (block highlighted in red in Fig. 3a). Subsequently, the system queries the available actions and presents the user with a control panel (Fig. 3b). After the user selects an alternative action to explore, the *A3S* backend processes the input, runs a simulation and returns the new trajectory. Finally, *TraceRL* updates the live graph accordingly (Fig. 3c).

The human-influenced cells are shown with a light-blue background, whereas the AI-generated actions produced in response to human input are highlighted in light yellow. The decision blocks from the original trajectory retain a neutral background. In combination with KPI readouts and uncertainty annotations, this GUI-based interaction provides a concrete mechanism to perform post-hoc counterfactual analysis on recorded episodes. In this sense, TraceRL operationalizes the *A3S* concept by

turning a service-oriented agent wrapper into a practical, interactive tool for uncertainty-aware decision support in simulated and replay settings. The structured audit trail of operator decisions, AI recommendations, and uncertainty estimates that TraceRL produces also makes it a natural fit for regulatory compliance and post-incident review in safety-critical domains.

4. RISK AND UNCERTAINTY IN DECISIONS

Integrating Artificial Intelligence (AI) agents into real systems demands clarity about how AI makes choices and where its limits lie, because trust collapses the moment those boundaries become opaque. This chapter introduces the core concepts of risk and uncertainty in decision-making for AI-assisted operations, focusing on situations in which humans use AI to guide actions that shape critical infrastructure. The goal of this chapter is to explain how uncertainty arises from both the model (i.e., RL agent) and the environment, how it influences the reliability of AI recommendations, and why human operators must understand these uncertainties to make robust decisions.

4.1. MOTIVATION

In autonomous decision-making, where AI agents hold full control over the decision loop, uncertainty tends to play a secondary role. Fully autonomous systems prioritize strong calibration and robust decision policies, relying on the agent's ability to estimate aleatoric uncertainty, i.e., the randomness rooted in the environment, such as weather variability or sensor noise. Accurate estimation of this environmental uncertainty strengthens the agent's robustness.

When AI shifts into a supporting role for humans, uncertainty becomes central. Beyond aleatoric uncertainty, the human operator must also understand the system's epistemic uncertainty, which reflects what the agent does not know due to limited training data or under-explored scenarios. High epistemic uncertainty signals that a recommendation is unreliable because the agent has not encountered comparable situations before, and its internal model may generalize poorly. Uncertainty estimators can provide operational self-awareness (Endsley, 2023), allowing the system to detect when it is operating outside its trained boundaries and to communicate those limitations directly to the human operator.

Decision-making typically imposes a high cognitive load on humans, who must maintain situational awareness and understand the evolving context to make informed and meaningful decisions. Current practices in critical infrastructures often involve multiple screens and applications that amplify this load, forcing operators to prioritize, organize, and correlate information before making decisions (Andrade et al., 2022).

As system complexity increases, AI-assisted decision making should reduce rather than amplify operators' cognitive burden – the mental effort required to interpret system states and evaluate possible actions. When this burden becomes excessive, it can lead to higher operator stress and reduced decision quality. Clear knowledge of an AI system's boundaries and limitations can help mitigate this effect by indicating when its recommendations can be trusted and when further human evaluation is

necessary (Zhou et al., 2017).

In AI-assisted decision making, the aim is to optimize how humans and AI share responsibility. When uncertainty is properly quantified and communicated, it can support rather than replace human judgment. If uncertainty is ignored or poorly conveyed, it can lead to over-reliance on automation or excessive skepticism, both of which undermine safety and performance. Effective assistance, therefore, requires both accurate modeling of uncertainty and user interface interactions that intelligibly support operators in interpreting and acting on it Operators must be able to interpret the magnitude and source of uncertainty to decide effectively whether to trust, defer to, or override an AI recommendation.

4.2. LITERATURE REVIEW

The deployment of AI models in safety-critical infrastructure requires more than high predictive accuracy; it requires rigorous guarantees of reliability and safety. Recent surveys in the field emphasize the critical shift from point forecasting to probabilistic forecasting to mitigate operational risks (Petropoulos et al., 2022). Following the widely accepted taxonomy proposed by Hüllermeier and Waegeman (Hüllermeier and Waegeman, 2021), predictive uncertainty is fundamentally decomposed into two distinct categories: epistemic uncertainty and aleatoric uncertainty.

Epistemic uncertainty, often called model uncertainty, arises from epistemic ignorance stemming from limited data coverage, insufficient exploration of the state–action space, or model misspecification. In the context of reinforcement learning (RL), this uncertainty reflects the agent's lack of knowledge regarding appropriate actions in unfamiliar or out-of-distribution system states. A defining characteristic of epistemic uncertainty is its reducibility; in theory, it can be mitigated by acquiring additional training data or refining the model architecture.

Several methodological paradigms have been proposed to estimate this form of uncertainty. Bayesian Neural Networks (BNNs) are considered the theoretical gold standard, replacing deterministic weights with probability distributions (Blundell et al., 2015); however, exact inference in BNNs is often computationally intractable. To address this, Gal and Ghahramani (Gal and Ghahramani, 2016) proposed Monte Carlo Dropout (MCD), which approximates Bayesian inference by keeping dropout layers active during testing. While effective, this method is an approximation and known to produce poorly calibrated uncertainty estimates, especially in areas where training data is sparse (Djupskas et al., 2026). A robust alternative is found in Deep Ensembles, where multiple independent networks are trained with random initializations (Lakshminarayanan et al., 2017). Although ensembles often provide superior uncertainty calibration, they incur high training and memory costs. To overcome the computational overhead of sampling-based methods, Sensoy et al. (Sensoy et al., 2018) introduced Evidential Deep Learning (EDL), an approach that treats network outputs as evidence to parameterize

a higher-order probability distribution (e.g., Dirichlet). This allows for the deterministic estimation of epistemic uncertainty in a single forward pass, making it highly suitable for real-time applications.

In contrast to the reducible nature of model knowledge, aleatoric uncertainty captures the inherent stochasticity of the physical process generating the data. In power systems, this manifests as sensor noise, stochastic fluctuations in load demand, or the intrinsic volatility of renewable generation. Unlike its epistemic counterpart, aleatoric uncertainty is irreducible from a data collection perspective, as adding more training examples allows for a better estimation of the noise distribution but does not eliminate the variability itself (Kendall and Gal, 2017). The literature addresses the quantification of this uncertainty predominantly through three distinct methodologies: parametric estimation, quantile regression, and residual-based modeling.

The parametric approach assumes that the target variable follows a known probability distribution, such as Gaussian or Poisson, whose parameters are estimated directly by the model. In this paradigm, a neural network or boosting model is trained to minimize the negative log-likelihood, simultaneously predicting the mean and the conditional variance as a function of the input. This formulation captures heteroskedasticity, the varying dispersion of error across the input space, which is critical in power grids where uncertainty often correlates with peak load hours. Recent advances, such as NGBoost (Duan et al., 2020), have extended this capability to Gradient Boosting algorithms, enabling full probabilistic forecasting for tabular data without the complexity of deep learning ensembles. As an alternative to rigid distributional assumptions, Quantile Regression has established itself as a standard in Probabilistic Load Forecasting (Hong and Fan, 2016). Rather than estimating distribution parameters, this method directly predicts conditional quantiles (e.g., the 90th percentile) by minimizing the pinball loss function (Koenker and Hallock, 2001). While robust to outliers and distribution-free, quantile regression may suffer from the quantile crossing problem. Finally, a third methodological avenue involves residual-based modeling or two-stage regression. This technique decouples point forecasting from uncertainty estimation: a model is first trained to predict the conditional mean, and a second model is then trained to predict the magnitude of errors, typically using squared residuals. This approach offers significant flexibility, allowing the use of specific loss functions for the variance, such as Poisson regression to enforce non-negativity, and the application of distinct algorithms for the central trend and the dispersion (Hastie et al., 2009).

In parallel, conformal prediction (CP) (Shafer and Vovk, 2008; Angelopoulos et al., 2023) has emerged as a distribution-free framework for constructing statistically valid uncertainty intervals for forecasted outcomes, and has recently been extended to support risk-based verification of operational boundaries in autonomous systems (Lindemann et al., 2025). Given a pre-trained predictive model and a user-defined significance level $\alpha \in [0, 1]$, CP provides a prediction interval that contains the true outcome with probability at least $(1 - \alpha)$, under the assumption of exchangeability. An advantage of CP

is that it operates as a post-hoc calibration layer, providing uncertainty intervals independently of the internal structure or quality of the underlying predictive model.

The simplest CP approach, commonly referred to as split conformal prediction (Vovk et al., 2005), constructs global prediction intervals by calibrating nonconformity scores on a held-out dataset. While this method guarantees marginal coverage, the resulting intervals have a constant width across the input space and may be overly conservative. To improve adaptivity, several extensions of CP have been proposed. Locally adaptive conformal methods provide tighter intervals in well-modeled regions and wider intervals in regions of higher uncertainty by scaling the nonconformity scores by local difficulty estimates, allowing adaptation to heteroskedastic target behavior while preserving marginal coverage guarantees (Renkema et al., 2024). Also focusing on heteroskedasticity in the data, Conformalized Quantile Regression (CQR) has been proposed to combine the flexibility of quantile regression with the statistical guarantees of CP, enabling the construction of asymmetric and input-dependent prediction intervals (Romano et al., 2019).

Another important line of work addresses non-exchangeable and time-dependent settings, where classical CP assumptions are violated. Adaptive Conformal Inference (ACI) introduces online updates to the significance level, maintaining long-term coverage guarantees under distribution shifts, making it particularly suitable for time series and evolving systems (Gibbs and Candes, 2021). This method is especially relevant for power system applications, where operating conditions change continuously due to demand variability, contingencies, and external disturbances.

In the context of RL, the work in (Strawn et al., 2023) creates safety filters by predicting other agent trajectories and providing intervals of uncertainty (using CP) around these trajectories, so that the agent can avoid these areas of higher uncertainty, in order to avoid collisions in their safety-critical setting. In (Silva et al., 2020), the epistemic uncertainty associated with each RL action is estimated using a bootstrapped DQN. When the uncertainty level exceeds a predefined threshold, the agent requests guidance from a demonstrator and uses this advice to guide exploration. In (Ibrahim et al., 2023), both epistemic and aleatoric uncertainties is explicitly quantified using an ensemble of distributional critics within an actor-critic framework. Epistemic uncertainty is leveraged to guide exploration via an upper-confidence-bound strategy, while aleatoric uncertainty is used to learn a risk-sensitive policy through conditional value-at-risk-based optimization in continuous action spaces.

In power grids, particularly in the context of congestion management, the forecasting of failures or the quantification of operational uncertainty in RL-based agents has been explored only to a very limited extent. For example, recent challenges from the L2RPN (Learning to Run a Power Network) competition, promoted by RTE and other partners, have incorporated the ability of AI agents to issue alarms to system operators into the evaluation score. However, in the 2021 competition, the proposed solutions were predominantly rule-based, relying on expert knowledge and not explicitly modeling uncertainty

or leveraging data-driven learning approaches (Marot et al., 2022). More recently, in the 2023 Energy Transition competition, most teams employed custom heuristic agents and rule-based strategies tailored to specific contingencies or seasonal periods (Pavão et al., 2025). A more advanced, data-driven approach based on supervised learning was introduced in (Lehna et al., 2024). The authors formulate a multi-class classification task, using several machine learning models (e.g., random forests, gradient boosting trees, and a categorical embedding model) to forecast whether the power grid will remain operational or experience a failure within a horizon of 5, 3, or 1 time steps, based solely on observations of the grid state. In addition, a clustering analysis was conducted to identify distinct grid failure modes, grouping them into five interpretable clusters. However, the model does not incorporate any explicit representation of the uncertainty associated with the RL agent's actions.

Finally, uncertainty estimates are used to implement Safe RL. For instance, in voltage-control applications, epistemic uncertainty can limit the agent's exploration, preventing it from taking actions that violate operational limits in unexplored network topologies (Su et al., 2025). Other approaches use Gaussian processes or BNN to model renewable generation uncertainty and ensure system stability under stochastic disturbances (Pepiciello and Domínguez-García, 2024).

4.3. UNCERTAINTY MODELING

4.3.1. CONTRIBUTIONS

We now describe a comprehensive framework, developed within Task 3.1, for assessing and forecasting the reliability of pre-trained RL agents, using power grid use cases from AI4REALNET as an illustrative application example. The framework is designed to support human operators by providing early warnings when RL agent decisions may become unreliable, without requiring modifications or retraining of the control policy. The main contributions are as follows:

- We introduce a reliability forecasting method that jointly integrates epistemic and aleatoric uncertainty estimates with indicators of network operational "complexity" to predict the probability of RL agent failure over future time horizons (e.g., the next hour). A Large Language Model (LLM) is employed to imitate a machine learning model and to generate fully interpretable outputs, including natural language explanations, that inform human operators about the conditions under which AI recommendations may fail. In contrast to related state-of-the-art approaches (e.g., (Pavão et al., 2025; Lehna et al., 2024)), our method explicitly incorporates epistemic uncertainty and leverages LLM-based imitation as a novel mechanism for transparent and operator-oriented decision support.

- A CP-based framework for providing intervals of uncertainty around forecasted power line load-

ings under contingencies, explicitly incorporating the effects of RL agent actions. The framework enables multi-hour ahead overload risk assessment, supported by the conformal guarantees provided by the used CP methods. This contrasts with existing alerting approaches in the state of the art (e.g., (Marot et al., 2022; Lehna et al., 2024; Pavão et al., 2025)), which mainly aim to estimate the probability of agent failure, without explicitly characterizing the resulting impact on the power grid state.

While this work is centered on RL, the proposed methodology and principles are not limited to this setting and can be transferred to other AI paradigms and deployment configurations.

4.3.2. FORMULATION OF THE PROBLEM

We consider a power grid whose operational state at a discrete time step t is denoted by $s_t \in \mathcal{S}$, where $\mathcal{S}$ represents the state space comprising the grid topology, power injections, and line flows. The topology and generator redispatch actions are defined by a pre-trained RL agent, governed by a fixed deterministic policy π:

$$\pi : \mathcal{S} \to \mathcal{A}, \qquad a_t = \pi(s_t), \tag{1}$$

where the π is treated as a "black box" and is not modified or re-trained in this work and $a_t \in \mathcal{A}$ represents the action taken at time t, given state s_t.

The objective is not to optimize the control performance of the RL agent, but rather to forecast the probability of operational failure introduced by deploying this policy. This forecast must account for the stochasticity of the power grid resulting from two distinct sources: epistemic uncertainty (u_t) and aleatoric uncertainty (σ_t).

For any given state s_t, we aim to forecast, at a lead time $t + k$ (where k is the prediction horizon) and using only the information available at time step t, a safety tuple $\langle \hat{p}_{fail,t+k|t}, \mathcal{C}_{l,t+k|t} \rangle$. The term $\hat{p}_{fail,t+k|t}$ denotes the predicted probability that the RL agent's action fails to maintain system stability (e.g., resulting in a blackout due to a cascading event) at time $t + k$. The set $\mathcal{C}_{l,t+k|t}$ represents a probabilistic uncertainty forecast interval for the line loading values at time $t + k$.

4.3.2.1 Forecasting RL Agent Failure Probability. The first component of the safety tuple aims to predict the likelihood of a system failure event. Let $Y_{t+k} \in \{0, 1\}$ be a binary variable representing the failure event at time $t+k$, where $Y_{t+k} = 1$ indicates a failure. We seek to estimate the conditional probability:

$$\hat{p}_{fail,t+k|t} = \mathbb{P}(Y_{t+k} = 1 \mid \mathbf{v}_t), \tag{2}$$

where $\mathbf{v}_t$ represents the vector of features available at time t.

To operationalize this, the feature vector $\mathbf{v}_t$ is constructed as a concatenation of heterogeneous data

sources, formally defined as:

$$\mathbf{v}_t = [\mathbf{x}_{topo} \oplus \mathbf{x}_{grid} \oplus \mathbf{x}_{stress} \oplus \mathbf{u}_{pred}], \tag{3}$$

where each component represents a distinct information domain:

- $\mathbf{x}_{topo} \in \mathbb{Z}$: The categorical encoding of the specific line l disconnected at time $t+k$, capturing local topological dependencies.
- $\mathbf{x}_{grid} \in \mathbb{R}^5$: The global operating point vector, comprising total active and reactive load (P_L, Q_L), total generation (P_G, Q_G), and the load-to-generation imbalance ratio ($r_{LG} = P_L/P_G$).
- $\mathbf{x}_{stress} \in \mathbb{R}^4$: Statistical descriptors of the thermal stress distribution across all transmission lines $\mathcal{L}$. Let ρ_i be the loading rate of line $i \in \mathcal{L}$. This vector includes the maximum loading ($\max_i \rho_i$), average loading ($\bar{\rho}$), variance (σ^2_ρ), and the cardinality of the critical set $N_{crit} = |\{i : \rho_i \geq 0.95\}|$.
- $\mathbf{u}_{pred} \in \mathbb{R}^3$: The uncertainty quantification vector containing the aleatoric uncertainty (σ_a) and the epistemic uncertainty prior to contingency (σ_e^{pre}).

The ground truth labels Y_{t+k} are obtained via a contingency analysis projected to a one-hour horizon $(t+12)$.
The central hypothesis of this approach is that incorporating aleatoric and epistemic uncertainty metrics into the supervised classifier enriches the model, thereby enabling more effective discrimination of situations in which the RL agent is prone to failure.

4.3.2.2 Uncertainty Forecast Interval of Line Loadings. The second component of the safety tuple addresses the risk of line congestion. Let $\rho_{l,t+k}$ denote the realized loading for transmission line $l \in \{1, \ldots, N_{line}\}$ at lead time $t+k$. Given a forecasted state of the grid $\hat{s}_{t+k|t}$, and the action $a_{t+k} = \pi(\hat{s}_{t+k|t})$ selected by the RL agent in that state, we obtain a point forecast $\hat{\rho}_{l,t+k|t}(sim(\hat{s}_{t+k|t}, a_{t+k}))$, in which $sim(\cdot)$ represents the new state obtained by simulating an action a_t in state s_t. For notational simplicity, we denote this forecast by $\hat{\rho}_l(\mathbf{x}_{t+k|t})$, where $\mathbf{x}_{t+k|t}$ represents the predicted system state for lead-time $t+k$, with information available at time step t, which results from simulating the RL agent's action a_{t+k} in the predicted state $\hat{s}_{t+k|t}$. We also use the notation $\mathbf{x}_t$ to generally denote the predicted state of the grid at timestep t, influenced by action a_t. The objective is to construct a prediction interval $\mathcal{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t})$ such that the realized line loading satisfies

$$\mathbb{P}(\rho_{l,t+k} \in \mathcal{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t})) \geq 1-\alpha,$$

where $\alpha \in [0, 1]$ is a user-specified significance level (e.g., 0.1 for 90% coverage). These intervals provide a probabilistic upper bound on line loadings, allowing us to quantify the risk of thermal limit violations that can lead to grid congestion.

4.3.3. UNCERTAINTY MODELING

This section describes the two complementary uncertainty estimation methods used to assess the reliability of a pre-trained RL agent for managing power grid congestion problems. It focuses on epistemic (model) uncertainty and the conformal uncertainty assessment framework used to quantify target (outcome) uncertainty, capturing uncertainty in future line loadings resulting from stochastic injections and system dynamics, conditioned on the agent's actions. These components are estimated independently and combined to form a safety tuple that supports risk-aware decision-making without modifying the underlying control policy.

4.3.3.1 Aleatoric and Epistemic Uncertainty. Epistemic uncertainty reflects the RL agent's confidence in its selected action when operating in a given grid state. High epistemic uncertainty indicates that the current state lies outside the distribution encountered during training, increasing the risk of unpredictable or unsafe control outcomes.

To quantify this uncertainty, EDL is adopted, an approach based on the Dempster-Shafer Belief Theory and Subjective Logic that treats neural network predictions not as simple point probabilities but as higher-order probability distributions (Sensoy et al., 2018). The developed model is trained to parameterize a Dirichlet probability density function over the possible output classes. By directly parameterizing a predictive distribution rather than relying on sampling-based techniques, EDL avoids the computational overhead of methods such as Monte Carlo dropout or ensemble models, which results in significantly faster inference that is particularly well suited for real-time and near–real-time power system applications.

To estimate aleatoric uncertainty, which captures the data's inherent stochasticity, a two-stage regression approach capable of modeling non-constant variance was employed. After training the mean prediction model, Histogram-based Gradient Boosting (HGB), the squared residuals of the training set predictions are calculated to approximate local variance. A second HGB model is then trained to predict these residuals using a Poisson loss function. This gradient boosting-based variance estimation aligns with recent advances in probabilistic boosting machines (Duan et al., 2020; Sprangers et al., 2021), which demonstrate that treating variance as a learnable parameter significantly improves calibration.

4.3.3.2 Evidential Modeling of Agent Behavior. Epistemic uncertainty, u, quantifies the agent's confidence, where high values signal out-of-distribution (OOD) states and increased operational risk. Following Sensoy et al. [7], u is derived from the total evidence $S = \sum \alpha_k$ and normalized by K classes as $u = K/S$. This metric yields an interpretable scalar: in familiar states, high evidence ($S \gg K$) drives $u \to 0$, while in unknown scenarios—such as novel post-contingency topologies—minimal evidence ($e_k \approx 0$) results in $S \approx K$ and maximum uncertainty $u \approx 1$.

For risk quantification, we adopt EDL Sensoy et al. (2018), which uses Dempster-Shafer Theory and Subjective Logic to treat predictions as higher-order probability distributions Sensoy et al. (2018). The model is trained to parameterize a Dirichlet density function over output classes. By directly parameterizing this distribution, EDL avoids the computational overhead of sampling-based methods like MCD or ensembles. This enables the significantly faster inference required for real-time power system applications.

We employ an EDL framework for behavioral cloning. Rather than optimizing for classification accuracy, the primary objective is to build a shadow model that replicates the fixed policy π, enabling the extraction of high-fidelity uncertainty metrics during inference. The network is trained strictly on the same state distribution $\mathcal{D}_{train}$ used to train the original RL agent, mapping the current grid state s_t to the predicted action a_t. To ensure non-negative outputs, a `softplus` activation is applied to the final layer, producing an evidence vector $\mathbf{e} = [e_1, \ldots, e_K]$ for K possible actions. This evidence is then mapped to the concentration parameters of a Dirichlet distribution, $\boldsymbol{\alpha}$, such that $\alpha_k = e_k + 1$. This formulation quantifies the density of the training data within the feature space, effectively modeling the knowledge boundary of the original RL agent and its reliability in novel states.

4.3.3.3 Conformal Risk Assessment. While epistemic uncertainty reflects the "confidence" in the action suggested by the RL agent, it does not capture uncertainty in the physical evolution of the power grid, which arises from both aleatoric and epistemic sources. Even for a fixed control action, future line loadings remain uncertain due to stochastic demand and variability in renewable generation. To quantify this uncertainty, CP is employed to construct forecast intervals around the line loading predictions, $\hat{\rho}_l(\mathbf{x}_t)$ obtained in the forecasted system states $\mathbf{x}_t$. In practice, grid operating conditions evolve over time and may violate the strict exchangeability assumption underlying CP, causing the marginal coverage guarantees to no longer hold exactly. To evaluate this limitation empirically, three CP variants with different notions of adaptivity are explored, as detailed in the following subsections.

The Split CP constructs prediction intervals using the residuals of the point forecasts made on a calibration dataset. Let $\mathcal{T}_{\text{cal}}$ denote the calibration set. For each transmission line l, the absolute residuals

are used as nonconformity scores, and a global quantile is computed as

$$\hat{q}_{l,1-\alpha} = quantile(\underbrace{\cup_{t\in\mathcal{T}_{cal}}\{|\rho_{l,t} - \hat{\rho}_l(\mathbf{x}_t)|\}}_{\text{abs residuals in calibration set}}, 1-\alpha). \tag{4}$$

At test time, this quantile is used to construct a prediction interval around the point forecast $\hat{\rho}_l(\mathbf{x}_{t+k|t})$:

$$\hat{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t}) = [\hat{\rho}_l(\mathbf{x}_{t+k|t}) - \hat{q}_{l,1-\alpha}; \hat{\rho}_l(\mathbf{x}_{t+k|t}) + \hat{q}_{l,1-\alpha}]. \tag{5}$$

Since the quantiles are global and independent of the test points, the resulting interval width remains constant across all test inputs for a given line, which may lead to overly conservative estimates. This essentially means that this method provides marginal coverage guarantees (assuming exchangeability), but does not achieve conditional coverage under general assumptions. Conditional coverage is defined as:

$$\mathbb{P}(Y_{\text{test}} \in C(X_{\text{test}})|X_{\text{test}} = x) \geq 1-\alpha. \tag{6}$$

When considering Split CP, achieving conditional coverage is impossible without additional assumptions (Foygel Barber et al., 2021), because information is gathered across the entire input space, leading to only marginal coverage guarantees.

The normalized CP variant with uncertainty scalars (using k-nearest neighbors – KNN) scales nonconformity scores using a local uncertainty estimate, following (Renkema et al., 2024). This estimate represents the difficulty of forecasting a specific data-point and is calculated for each $\mathbf{x}_t$ by averaging the residuals of its k most similar neighbors in the calibration dataset,

$$\text{difficulty}_l(\mathbf{x}_t) = \max\Big(\text{mean}\big(\underbrace{\cup_{t_s\in\mathcal{N}_k(\mathbf{x}_t|\mathcal{T}_{cal})}|\rho_{l,t_s} - \hat{\rho}_l(\mathbf{x}_{t_s})|}_{\substack{\text{set of } k \text{ closest neighbors}\\ \text{in calibration set}}}\big), \varepsilon\Big), \tag{7}$$

where $\mathcal{N}_k(\mathbf{x}_t|\mathcal{T}_{cal})$ represents the set of k closest points to $\mathbf{x}_t$ in the calibration set and $\varepsilon > 0$ prevents numerical instability.

Each calibration residual is normalized as

$$\text{score}_l^{(norm)}(\mathbf{x}_t) = \frac{|\rho_{l,t} - \hat{\rho}_l(\mathbf{x}_t)|}{\text{difficulty}_l(\mathbf{x}_t)}, \tag{8}$$

and a global quantile is then computed from the normalized scores:

$$\hat{q}_{l,1-\alpha}^{(norm)} = quantile(\underbrace{\cup_{t\in\mathcal{T}_{cal}}\{\text{score}_l^{(norm)}(\mathbf{x}_t)\}}_{\text{normalized residuals}}, 1-\alpha). \tag{9}$$

At test time, given a new forecasted system state $\mathbf{x}_{t+k|t}$, the prediction interval is obtained by rescaling

this quantile:

$$\hat{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t}) = \Big[\hat{\rho}_l(\mathbf{x}_{t+k|t}) - \hat{q}^{(norm)}_{l,1-\alpha} \times \text{difficulty}_l(\mathbf{x}_{t+k|t}); \\ \hat{\rho}_l(\mathbf{x}_{t+k|t}) + \hat{q}^{(norm)}_{l,1-\alpha} \times \text{difficulty}_l(\mathbf{x}_{t+k|t})\Big]. \tag{10}$$

It is expected that using these uncertainty scalars will lead to greater adaptivity, thereby approximating conditional coverage.

The previously proposed methods do not provide exact guarantees for time series data due to the violation of the exchangeability requirement. ACI addresses this limitation by dynamically updating the significance level α^* in an online fashion (Gibbs and Candes, 2021), using a gradient descent-like procedure in which: (i) if intervals are too narrow (miss too often) α^* decreases, producing wider intervals; and (ii) if intervals are too wide (coverage above target) then α^* increases, shrinking intervals. i.e., the adaptive significance level is updated as

$$\alpha^*_{l,t} = \alpha^*_{l,t-1} + \gamma\left(\alpha - \mathbf{1}\left(\rho_{l,t-1} \notin \hat{C}_{l,1-\alpha^*_{l,t-1}}(\mathbf{x}_{t-1})\right)\right), \tag{11}$$

where γ is the step size parameter. ACI starts with $\alpha^* = \alpha$. The quantile is calculated for all forecasted states $(\mathbf{x}_t, ..., \mathbf{x}_{t+k})$ using the adaptive significance level at time t, corresponding to $\alpha^*_{l,t}$

$$\hat{q}_{l,1-\alpha^*_{l,t}} = quantile(\underbrace{\cup_{t_s \in \mathcal{T}_{cal}}\{|\rho_{l,t_s} - \hat{\rho}_l(\mathbf{x}_{t_s})|\}}_{\text{abs residuals in calibration set}}, 1-\alpha^*_{l,t}), \tag{12}$$

and is updated sequentially in a delayed fashion when the real simulation reaches time $t+k$, using Eq. 11.

The prediction interval is then

$$\hat{C}_{l,1-\alpha^*_{l,t}}(\mathbf{x}_{t+k|t}) = [\hat{\rho}_l(\mathbf{x}_{t+k|t}) - \hat{q}_{l,1-\alpha^*_{l,t}}; \hat{\rho}_l(\mathbf{x}_{t+k|t}) + \hat{q}_{l,1-\alpha^*_{l,t}}].$$

This adaptive mechanism enables long-term (asymptotic) coverage guarantees under distribution shift, making ACI particularly interesting for evolving power system operations.

4.3.4. RL AGENT UNCERTAINTY AND FAILURE FORECASTING

In this section, we describe the methodology in more detail, explaining how future forecasted grid states are obtained, how epistemic uncertainty is employed, and how the CP framework is built.

4.3.4.1 Future Grid States. The first step is to forecast the network's future states to enable contingency analysis of look-ahead time steps. To this end, a regression model using the HGB algorithm was developed. This algorithm was selected for its computational efficiency with large data sets and its ability to capture the complex nonlinearities inherent in the dynamics of the power grid. Nevertheless,

any other machine learning algorithm for load and generation forecasting could be used.

The forecasting problem is formulated as a multi-output regression, where the model receives a vector of historical observations and temporal variables as input and simultaneously predicts the active and reactive power injections for all loads and generators over a 1-hour time horizon. The attribute vector is constructed using sliding time windows that capture short-term (5-minute lag), medium-term (1-hour lag), and long-term (one-week lag) dynamics. Additionally, the temporal periodicity is encoded through trigonometric transformations (sine and cosine) of the hour of the day, the minute of the hour, and the day of the week, ensuring that the model learns the seasonality of consumption and generation.

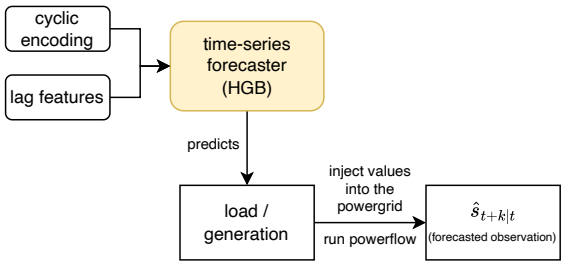


FIGURE 4 - OBTAINING THE FORECASTED OBSERVATIONS.

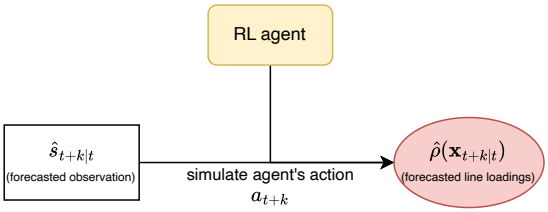


FIGURE 5 - OBTAINING THE FORECASTED LINE LOADINGS.

From these estimates, the grid state is deterministically reconstructed using a power flow solver (*LightSimBackend* ()lightsim2grid). This process is illustrated in Fig. 4.

In order to obtain forecasted line loading values for the CP framework, the action a_{t+k} that would be taken by the RL agent in this reconstructed state $\hat{s}_{t+k|t}$ is simulated, resulting in the state $\mathbf{x}_{t+k|t}$, from which the values of $\hat{\rho}_l(\mathbf{x}_{t+k|t})$ are collected, as illustrated in Fig. 5.

4.3.4.2 Failure Forecasting. The proposed system consists of three sequential blocks: (a) a power injection prediction module (load and generation) based on HBG; (b) an uncertainty quantification module that estimates aleatoric and epistemic components, which leads to the development of an alarm system; and (c) the extraction of interpretable decision rules using an LLM.

The central component of security assessment lies in the ability to anticipate future vulnerabilities. Rather than assessing only the current state, the methodology projects contingency analysis ($N-1$ criterion used for power grid operators) to a one-hour time horizon. This means that a line is disconnected at timestep $t+12$ to understand whether the system remains online under this contingency.

The supervised dataset generation process follows the predictive logic shown in Fig. 6. As illustrated, the process is divided into three stages described in Figure 7: (i) state forecast, where the HBG model estimates load and generation injections for a 60-minute horizon; (ii) future contingencies simulation, applying critical line disconnections to the forecasted state at $t+12$; and (iii) feasibility verification,

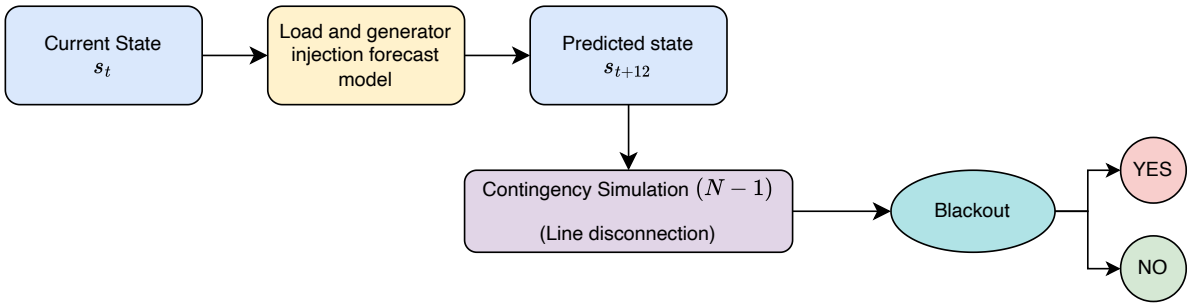


FIGURE 6 - FLOWCHART OF THE PREDICTIVE CONTINGENCY ANALYSIS METHODOLOGY.

assessing whether the network maintains operational stability or whether the disturbance triggers a cascading event (potentially leading to a blackout).

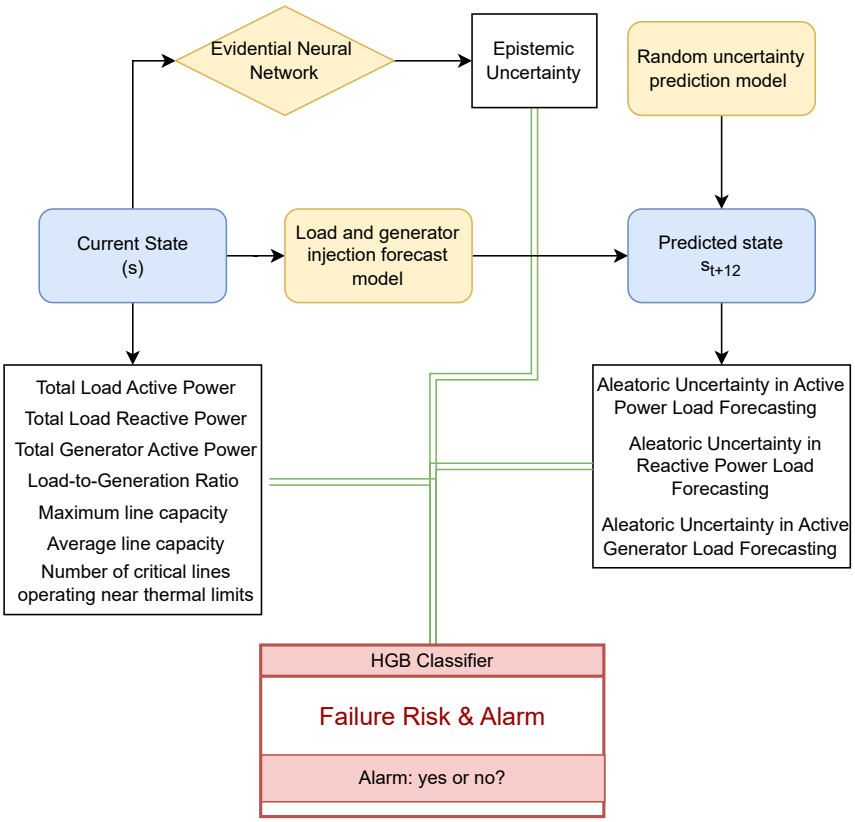


FIGURE 7 - ARCHITECTURE OF THE ALARM SYSTEM.

The classifier aggregates, at a specific time step (e.g., $t+12$), the following metrics (i.e., model inputs) to generate a binary risk alert:

- **Power balance:** aggregated metrics characterizing the system operating point, specifically the total active and reactive load and generation injections. Additionally, the *load-to-generation ratio* is introduced as a derived feature to explicitly quantify the system-wide power balance

and capture deviations between total demand and available generation.

- **Thermal stress:** statistical descriptors of the line loading rates (ρ) across the entire network. These include the maximum and average ρ, variance, and the count of critical lines operating near thermal limits ($\rho \geq 0.95$). These variables capture the proximity of the grid to its physical boundaries.

- **Predictive uncertainty context:** Probabilistic indicators derived from the upstream estimators. This includes the *aleatoric uncertainty* (inherent noise in load/generation forecasts) and the *epistemic uncertainty* estimated by the EDL prior to the contingency (*epistemic_before*).

- **Grid contingencies:** The specific identifier of the disconnected line, encoded to allow the model to learn local vulnerability patterns associated with individual grid components.

Based on this historical dataset, the HGB classifier is trained to directly distinguish between stable and critical operating points. The model learns the non-linear decision boundary separating the two classes, mapping the input feature vector to a binary prediction. Consequently, the system operates as an immediate early warning mechanism: whenever the model predicts a future failure event (positive class), an alarm is triggered to prompt preventive operator intervention.

Following the implementation of the HGB model, a critical challenge remains, i.e., the inherent "black-box" nature of ensemble methods, which limits transparency and hinders the interpretability needed for human operator validation. To address this, we implement a post-hoc interpretability framework that uses an LLM as a "rule generator" (Bessa, 2025).

The objective is to create a transparent surrogate model that approximates the HGB's predictive behavior. This approach tasks the LLM with analyzing the input features and the HGB's output to synthesize executable, human-readable Python code in a strict `if-then` format (Bessa, 2025). This transforms complex non-linear patterns into explicit operational protocols.

To translate the HGB predictions into interpretable logic for line disconnections, two distinct architectural strategies are defined, formalized in Algorithms 1 and 2. These options trade off between the breadth of the context window and the granularity of the specific physical rules:

Unified contingency logic (global call). In this approach (see Algorithm 1), the LLM processes a dataset containing identifiers for all relevant network components simultaneously. The model acts as a global classifier, generating a single hierarchical decision tree that first branches based on the topological state before evaluating physical variables.

- **Structure:** The generated code prioritizes the categorical variable (the line identifier) to segment the logic, followed by nested conditions for load and uncertainty.
- **Logic Example:**

* *If* `line_id == 1`: Check `sum_load_p >= 200` AND **uncertainty_before > 0.5**.
* *If* `line_id == 2`: Apply distinct thresholds relevant to this specific line's capacity.

- **Implications:** This mimics a global decision tree, consolidating all logic into one script. However, this may exceed the LLM's reasoning capabilities when the number of lines is large, as the context window becomes saturated with topological distinctions rather than physical-state analysis.

Algorithm 1 LLM-based rule extraction: Unified global approach with feedback

Input: Dataset $\mathcal{D} = \{X, L\}$, Black-box Model $\mathcal{M}_{HGB}$, Target $F2_{target}$
Output: Global Decision Tree Code $\mathcal{R}_{global}$

▷ Step 1: Generate Surrogate Targets
$Y_{target} \leftarrow \mathcal{M}_{HGB}.predict(\mathcal{D})$
$\mathcal{D}_{aug} \leftarrow \mathcal{D} \cup Y_{target}$

▷ Step 2: Construct Hierarchical Prompt
Select stratified sample $\mathcal{S} \subset \mathcal{D}_{aug}$ containing all $l \in L$
Initialize prompt history $\mathcal{H}$ with $\mathcal{S}$ and hierarchical instruction

▷ Step 3: Iterative Refinement Loop
for $i \leftarrow 1$ to N **do**
    $\hat{c} \leftarrow$ LLM$(\mathcal{H})$ ▷ Generate Global Script
    $Y_{pred} \leftarrow$ Execute$(\hat{c}, \mathcal{D})$ ▷ Validate on full dataset
    $Score_{F2} \leftarrow$ CalculateF2(Y_{target}, Y_{pred})

    if $Score_{F2} \geq F2_{target}$ **then**
        $\mathcal{R}_{global} \leftarrow \hat{c}$
        break
    else
        $F_{back} \leftarrow$ GenerateFeedback$(Score_{F2}, Precision, Recall)$
        Update $\mathcal{H}$ with $\hat{c}$ and F_{back} ▷ Refine Global Logic
    end if
end for

return $\mathcal{R}_{global}$

Modular asset-specific logic (iterative calls). As detailed in Algorithm 2, this strategy treats each line disconnection as an independent problem. We employ a "sliding window" or filtered context approach, where the LLM is called individually for each line.

- **Structure:** The LLM is provided only with the HGB's behavior regarding a specific asset (e.g., only data where `line_id == 1` is disconnected). It generates a specialized function for that specific contingency.
- **Logic Example:** For the `line_1_3_3` call, the rule set focused immediately on physics: `if sum_load_q > 100` then `Alert`.
- **Implications:** This yields a library of modular, high-precision rules. By removing the noise

introduced by other topologies, the LLM can focus on capturing subtle non-linear relationships between load patterns and uncertainty that are specific to each line.

Algorithm 2 LLM-based local rule extraction with feedback

Input: $\mathcal{D}, \mathcal{M}_{HGB}, F_2^\star, N$
Output: $\mathcal{R}_{global}$
$Y \leftarrow \mathcal{M}_{HGB}.predict(\mathcal{D})$ ▷ Surrogate labels
$\mathcal{D}^+ \leftarrow \mathcal{D} \cup Y$ ▷ Augmented dataset
Select stratified sample $\mathcal{S} \subset \mathcal{D}^+$ covering all $l \in L$
Initialize prompt history $\mathcal{H}$ with $\mathcal{S}$ and global rule instructions
for $i = 1$ to N **do**
    $\hat{c} \leftarrow \text{LLM}(\mathcal{H})$ ▷ Generate rule script
    $\hat{Y} \leftarrow \text{Execute}(\hat{c}, \mathcal{D})$ ▷ Evaluate on full data
    $F_2 \leftarrow \text{CalculateF2}(Y, \hat{Y})$
    if $F_2 \geq F_2^\star$ **then**
        $\mathcal{R}_{global} \leftarrow \hat{c}$
        break
    else
        $\mathcal{F} \leftarrow \text{GenerateFeedback}(F_2, \text{Precision}, \text{Recall})$
        Update $\mathcal{H}$ with $\hat{c}$ and $\mathcal{F}$ ▷ Refine prompt
    end if
end for
return $\mathcal{R}_{global}$

Legend: $\mathcal{D}$: input dataset; $\mathcal{M}_{HGB}$: trained HGB classifier; Y: surrogate labels predicted by $\mathcal{M}_{HGB}$; $\mathcal{D}^+$: augmented dataset; $\mathcal{S}$: stratified sample; L: set of line identifiers; $\mathcal{H}$: prompt history; $\hat{c}$: rule script generated by the LLM; $\hat{Y}$: predictions from executing $\hat{c}$; F_2: achieved F2-score; $F_2^\star$: target F2-score; $\mathcal{F}$: feedback message; N: maximum number of refinement iterations; $\mathcal{R}_{global}$: final global rule set.

Constraints and iterative refinement To ensure the generated rules are operationally safe and mathematically robust, the framework enforces a "Feedback Loop" methodology following recent research on LLM optimization (Bessa, 2025). The generation process adheres to the following protocol:

- **Syntactic constraints:** The LLM is strictly prohibited from using `elif` statements. It must generate "pure nested if/else" structures. This constraint forces the model to create a structure identical to a binary decision tree, which is essential for visual auditing by human operators.
- **Automated feedback:** The system operates cyclically. After the LLM proposes a rule set, a secondary "Feedback Agent" evaluates the set of these rules against the HGB baseline. If the error is high, the agent instructs the generator to refine specific thresholds in the next iteration.
- **Justification:** Every rule set is preceded by a natural language justification. The LLM must explain *why* a combination of various variables triggers a specific classification, bridging the gap between raw data and operator intuition.

4.3.4.3 Line Loadings Uncertainty Intervals Forecasting. A framework is developed for applying CP to forecasted values of $\hat{\rho}_l$ in order to assess the risk of congestion in power lines (see the overview in Fig. 8).

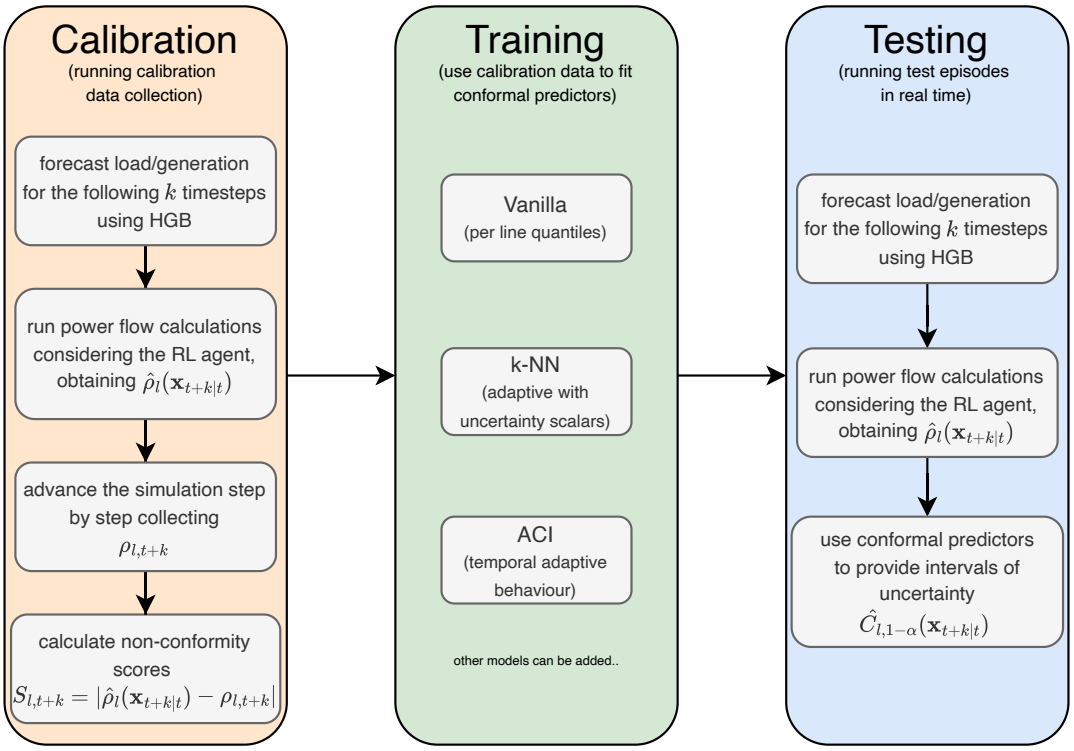


FIGURE 8 - OVERVIEW OF THE THREE MAIN PHASES IMPLEMENTED IN THE CP FRAMEWORK.

As explained before, the framework employs a forecasting process that uses a load and generation time-series forecaster (HGB). In the calibration phase, the calibration set is collected as the result of running n episodes, using the rolling forecast process illustrated in Fig. 9.

At a given timestep t, the forecaster generates future load and generation values for horizons $k \in (1, \ldots, 12)$ (i.e., $(t+1, \ldots, t+12)$), representing one hour of real time at 5-minute interval resolution. From these predictions, the values of $\hat{\rho}_l(\mathbf{x}_{t+k|t})$ are collected (shown in green in Fig. 9). The simulation then advances, step by step, in which the actual values of $\rho_{l,t+k}$ are collected (shown in purple for each timestep $(t+1, \ldots, t+12)$). When all the real values of the line loadings have been collected (i.e., $\rho_{l,t+k}$ with $k \in (1, \ldots, 12)$), a new forecast is started at $t+12$ for timesteps $(t+13, \ldots, t+24)$ (represented in green). This cycle occurs repeatedly for the collection of the calibration dataset.

In many of the timesteps, the RL agent chooses not to take any action, which can be interpreted as a "Do Nothing" action. In order to give emphasis to periods in time where the RL agent chooses or is forced to choose a non-empty action, every time that the real simulation advances and a non-empty agent action is detected, a new forecast is started (the remaining previous forecast is ignored), and

Algorithm 3 Calibration Phase

Input: Environment $\mathcal{E}$, Agent π, Forecaster, Episodes n, Mode (single/ensemble)
Output: Calibration set

if mode is single **then**
    Get attacked lines $\mathcal{A} \subseteq \mathcal{L}$
    $\mathcal{D}_{cal} \leftarrow$ RunCalibrationEpisodes($\mathcal{E}, \pi, n, \mathcal{A}$)
    return $\mathcal{D}_{cal}$
else ▷ Ensemble Mode
    for line $l \in \mathcal{L}$ **do**
        $\mathcal{D}_{cal}^{(l)} \leftarrow$ RunCalibrationEpisodes($\mathcal{E}, \pi, n, \{l\}$)
    end for
    return $\{\mathcal{D}_{cal}^{(l)} : l \in \mathcal{L}\}$
end if

function RunCalibrationEpisodes($\mathcal{E}, \pi, n, Lines$)
    for $e = 1$ to n **do**
        Set attacks on Lines $\subseteq \mathcal{A}$
        while episode not done **do**
            Get observation s_t
            Get action $a_t \leftarrow \pi(s_t)$
            Get actual $\rho_{l,t}$
            if new forecast is needed **then**
                Forecast $\mathbf{x}_{t+k|t}$ for $k \in \{1, \ldots, 12\}$
                Get $\hat{\rho}(\mathbf{x}_{t+k|t})$
                if a_t is not "Do Nothing" **then**
                    Set timesteps $(t+1, \ldots, t+12)$ as action-influenced
                end if
            end if
            Add $(\mathbf{x}_t; \hat{\rho}_l(\mathbf{x}_t); \rho_{l,t})$ to $\mathcal{D}_{Cal}$
            Step environment with a_t
        end while
    end for
    return $\mathcal{D}_{Cal}$
end function

the 12 subsequent forecasted timesteps are marked as action-influenced. This process is exemplified in Fig. 10, in which a non-empty action is detected when collecting the real line loadings at timestep $t+2$, triggering a new forecast for timesteps $(t+3, \ldots, t+14)$. This allows capturing the behavior of the conformal models in the timesteps that succeed a non-empty action.

Furthermore, the calibration dataset can be collected with or without adversarial attacks. Let $\mathcal{L}$ be the set of all power lines, and let $\mathcal{A}$, with $\mathcal{A} \subseteq \mathcal{L}$, be the subset of the lines that are subjected to adversarial attacks during calibration. In the case that $\mathcal{A} = \emptyset$, this corresponds to obtaining the calibration data under normal operational conditions, while $\mathcal{A} \neq \emptyset$ means that the calibration data is collected under adversarial attacks on the lines in $\mathcal{A}$.

The different conformal predictors studied are fit using the data collected during the calibration phase. In this framework, two types of training are employed: single predictor and an ensemble of predictors. In the single predictor setting, the data is collected (in the calibration phase) using one of two ap-

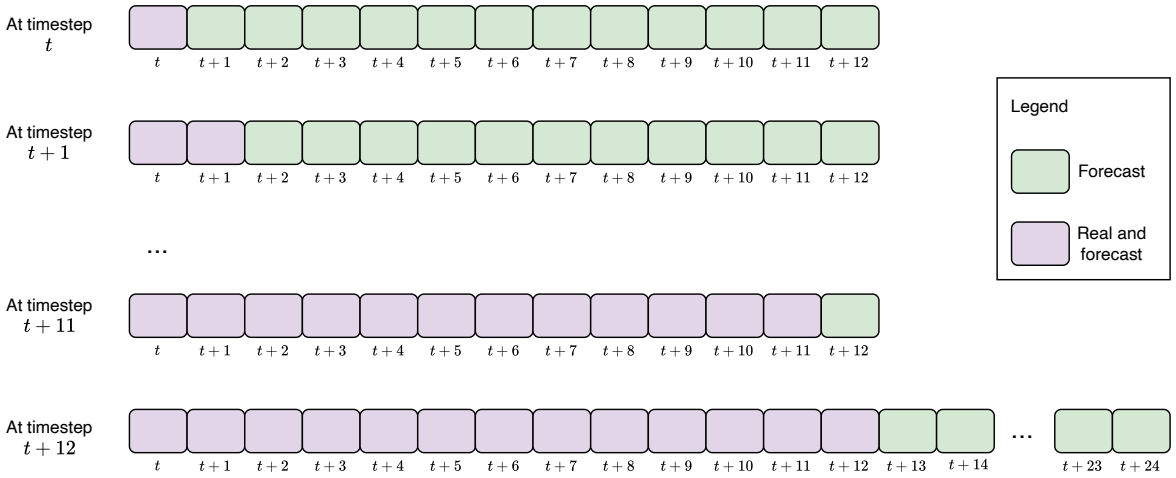


FIGURE 9 - VISUAL REPRESENTATION OF THE $(\rho_{l,t+k}, \hat{\rho}_l(\mathbf{x}_{t+k|t}))$ PAIRS COLLECTION PROCESS.

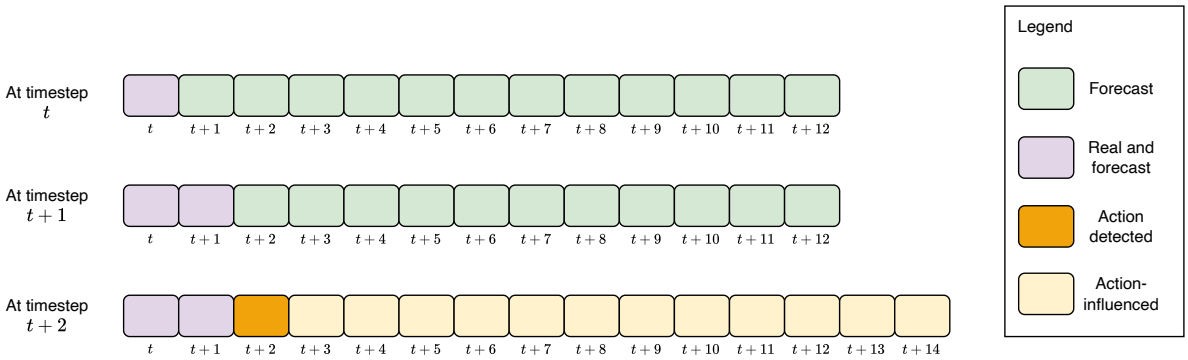


FIGURE 10 - EXAMPLE OF ACTION-INFLUENCED TIMESTEPS $(\rho_{l,t+k}, \hat{\rho}_l(\mathbf{x}_{t+k|t}))$ PAIRS COLLECTION PROCESS WHEN A NON-EMPTY AGENT'S ACTION IS DETECTED (AT $t+2$).

proaches: (a) uses data that was collected under normal operational conditions, which includes natural load fluctuations and generation dispatch changes (i.e., $\mathcal{A} = \emptyset$), and (b) some of the lines (user specifiable) are subjected to adversarial attacks during the calibration set data collection (i.e., $\mathcal{A} \neq \emptyset$). In fact, the first approach can be viewed as a special case of the second approach in which no power lines are subjected to adversarial attacks.

The ensemble training follows a different strategy. When using this type of training, an ensemble of models is created, in which each model is trained in a scenario where only one of the power lines is subjected to adversarial attacks (i.e., $|\mathcal{A}| = 1$). For this reason, the ensemble has one model for each power line, and each of these models is meant to capture the impact that attacking a particular power grid line has on the grid (i.e., how the power grid globally responds to attacks on a specific line).

Similar to the procedure from the calibration phase, load and generation forecasts are made for the short-term (from which the values of $\hat{\rho}_l(\mathbf{x}_{t+k|t})$ are obtained) as illustrated in Fig. 9, and the conformal predictors fitted during the training phase are used to provide intervals of uncertainty $\hat{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t})$ around the values of $\hat{\rho}_l(\mathbf{x}_{t+k|t})$. The simulation advances in real time, step by step, in which we collect

Algorithm 4 Training Phase

Input: Calibration data (from Algorithm 3), significance level α, Mode (single/ensemble)
Output: Conformal Predictors

if single mode **then**
    Fit $\mathcal{C}$ on $\mathcal{D}_{cal}$ with significance level α
    return $\mathcal{C}$
else ▷ Ensemble Mode
    for each line $l \in \mathcal{L}$ **do**
        Fit $\mathcal{C}^{(l)}$ on $\mathcal{D}_{cal}$ with significance level α
    end for
    return $\{\mathcal{C}^{(l)} : l \in \mathcal{L}\}$
end if

the actual system outcomes (the true values of $\rho_{l,t+k}$), which we use to evaluate the performance of our models. As in the calibration phase, when a non-empty action from the RL agent is detected, a new forecast is performed (as illustrated in Fig. 10). The test episodes happen under an adversarial attack configuration that is defined by a subset $\mathcal{A} \subseteq \mathcal{L}$, with $|A| \geq 0$ (user-definable).

Algorithm 5 Testing Phase

Input: Environment $\mathcal{E}$, Agent π, Forecaster, Predictor $\mathcal{C}_m$, Episodes n, Attacked lines $\mathcal{A} \subseteq \mathcal{L}$

for $e = 1$ to n **do**
  Set attacks on lines $\in \mathcal{A}$
  while episode not done **do**
    Get observation s_t
    Get action $a_t \leftarrow \pi(s_t)$
    Get actual $\rho_{l,t}$
    if new forecast is needed **then**
      Forecast $\mathbf{x}_{t+k|t}$ for $k \in \{1, \ldots, 12\}$
      Get $\hat{\rho}(\mathbf{x}_{t+k|t})$
      if a_t is not "Do Nothing" **then**
        Set timesteps $(t+1, \ldots, t+12)$ as action-influenced
      end if
      if mode is single **then**
        Compute intervals $\hat{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t})$
      else
        for each model $\mathcal{C}_m^{(i)}$, with $i \in \mathcal{L}$ **do**
          Get intervals $[L^{(i)}, U^{(i)}] = C_{l,1-\alpha}^{(i)}(\mathbf{x}_{t+k|t})$
        end for
        $L = \min\{L^{(1)}, L^{(2)}, \ldots, L^{(|\mathcal{L}|)}\}$ ▷ minimum lower bound
        $U = \max\{U^{(1)}, U^{(2)}, \ldots, U^{(|\mathcal{L}|)}\}$ ▷ maximum upper bound
        Compute intervals $\hat{C}_{l,1-\alpha}(\mathbf{x}_{t+k|t}) = [L, U]$
      end if
    end if
    Step environment with a_t
  end while
end for

5. MULTI-OBJECTIVE DECISION-MAKING WITH AI

5.1. MOTIVATION

Reinforcement Learning (RL) has emerged as a powerful framework for sequential decision-making in complex and uncertain environments. Traditionally, RL focuses on optimizing a single reward function, aiming to maximize cumulative returns over time. However, many real-world scenarios involve multiple, often conflicting objectives. For instance, an autonomous vehicle must balance safety, speed, and energy efficiency; a recommendation system strives to provide relevant yet diverse content; and power grid management requires optimizing for reliability, cost, and environmental impact.

These multifaceted challenges necessitate an extension of the standard RL framework to accommodate multiple objectives simultaneously. Multi-Objective Reinforcement Learning (MORL) addresses this need by enabling agents to learn policies that consider various criteria concurrently. This extension introduces new complexities, such as representing and balancing conflicting goals, adapting to dynamic preferences, and evaluating policies based on multiple performance metrics.

This section aims to provide an overview of MORL concepts, discussing its theoretical foundations, methodological approaches, practical applications, and current research challenges. In addition, the approach taken by the project is explained in more detail, and the corresponding software for realizing multi-objective reinforcement learning agents is described. A multi-objective approach will enable adaptive, situational flexibility for operators of critical infrastructure.

5.1.1. MULTI OBJECTIVE REINFORCEMENT LEARNING FUNDAMENTALS

Reinforcement Learning (RL) is a machine learning paradigm where an agent learns to make decisions by interacting with an environment to maximize cumulative rewards. The environment is typically modeled as a Markov Decision Process (MDP), defined by a set of states S, a set of actions A, a transition function $T(s^{'}|s,a)$, a reward function $R(s,a)$, and discount factor $\gamma \in [0,1)$. At each time step, the agent observes the current state s, selects an action a, receives a reward r, and transitions to a new state $s^{'}$. The agent's goal is to learn a policy $\pi : S \rightarrow A$ that maximizes the expected cumulative discounted reward over time.

Multi-Objective Reinforcement Learning (MORL) extends the traditional RL framework to scenarios where decisions must be made considering multiple objectives simultaneously. In MORL, the scalar reward function R(s,a) is replaced with a vector-valued reward function:

$$\vec{R}(s,a) = [r_1(s,a), r_2(s,a), ..., r_n(s,a)] \tag{13}$$

Each component $r_i(s,a)$ corresponds to a distinct objective, such as efficiency, safety, or cost. This formulation leads to a Multi-Objective Markov Decision Process (MOMDP), where the agent must learn policies that balance these objectives effectively.

Unlike single-objective RL, where the goal is to find a policy that maximizes a single cumulative reward, MORL seeks to identify policies that achieve a desirable trade-off among multiple objectives. Two primary approaches are employed (Hayes et al. (2022)):

- **Scalarization Methods**: These techniques convert the multi-objective problem into a single-objective one by combining the multiple rewards into a scalar value, often through weighted sums. This approach typically combines the individual objectives using a linear combination of the individual rewards: $V_w = w^\intercal \vec{R}$, where the weight vector w captures the relative importance of individual rewards. While pre-determining the relative importance of reward weights is straightforward, it locks the resulting policy into a fixed prioritization of objectives. However, the linear scalarization of objectives, as a mathematical construct, can be used as a tool to develop a set of optimal policies, as described below.

- **Pareto-Based Methods**: These approaches aim to find a set of Pareto-optimal policies, where no single policy is superior across all objectives. A policy is Pareto-optimal if no other policy improves one objective without worsening others. This method provides a diverse set of solutions, allowing for flexibility in choosing policies based on specific preferences or constraints.

In multi-objective optimization, the **Pareto front** represents the set of non-dominated solutions, where improving one objective would lead to the deterioration of at least one other. In the context of MORL, the Pareto front comprises policies that offer different trade-offs among objectives, providing decision-makers with a spectrum of optimal choices. For example, in autonomous driving, one policy may prioritize speed over fuel efficiency, while another emphasizes safety at the expense of travel time. Both policies could be Pareto-optimal, catering to different user preferences. Identifying the Pareto front enables the development of agents capable of adapting to varying priorities and constraints. Evaluating the quality of the Pareto front often involves metrics such as hypervolume, which measures the space covered by the Pareto-optimal solutions, and diversity indicators, which assess the spread of solutions across the objective space. Understanding and approximating the Pareto front is crucial for deploying MORL in real-world applications, where flexibility and adaptability to changing objectives are essential.

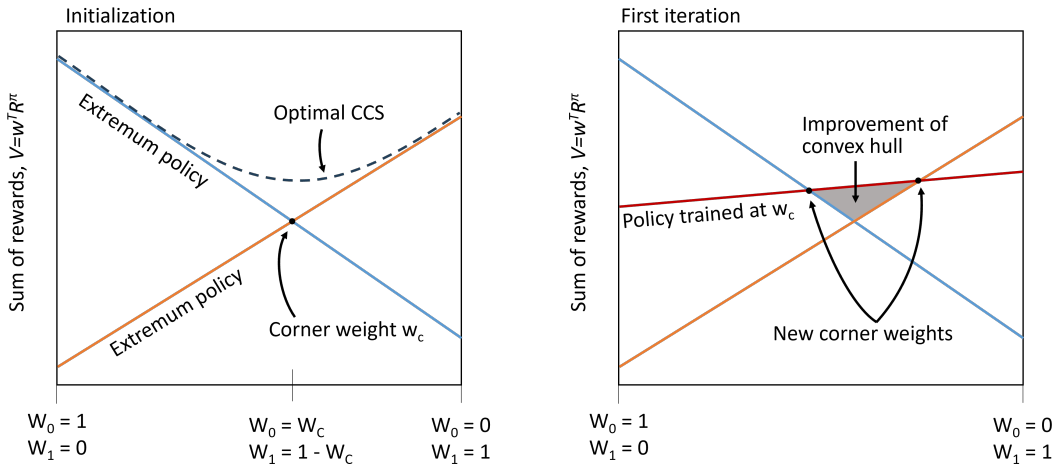


FIGURE 11 - SCHEMATIC OF THE OPTIMISTIC LINEAR SUPPORT ALGORITHM.

5.2. MULTI-OBJECTIVE ALGORITHM DESIGN

In the following, we outline the strategy employed in AI4REALNET to train AI-based agents to solve multi-objective reinforcement learning problems. The approach follows that described in (Lautenbacher et al. (2025)) for the electrical grid optimization use case.

5.2.1. FINDING CONVEX COVERAGE SETS

The approach taken in the project is to sample the vector-valued reward space defined using the linear scalarization formulation described above. In other words, the reward space is defined using the weights w used to scalarize the vector-valued reward function in the equation $V_w = w^\intercal \vec{R}$). The goal of multi-objective optimization becomes one of finding the **convex coverage set** (CCS): a set of algorithm policies that is optimal for any combination of weights. This can be performed by randomly sampling the reward space and retraining for a policy each time, but this is computationally expensive, especially when the number of objectives is large. A procedure known as optimistic linear support, described below, is used to significantly improve the efficiency of this process.

5.2.2. DEEP OPTIMISTIC LINEAR SUPPORT

The optimistic linear support (OLS) algorithm (Mossalam et al. (2016), Roijers (2016)) uses an iterative procedure to efficiently find the optimal convex coverage set. Illustrated in Fig. 11 for a two-objective scenario, the process is initialized by first considering the cases in which $w_i = 1$ and all other weights are set to 0, for each multi-objective weight w_i. In each case, an algorithm is trained within the reinforcement learning environment, and the resulting set of "extremum policies" is added to the convex coverage set. As shown in Fig. 11, each policy performs optimally for its corresponding objective, and

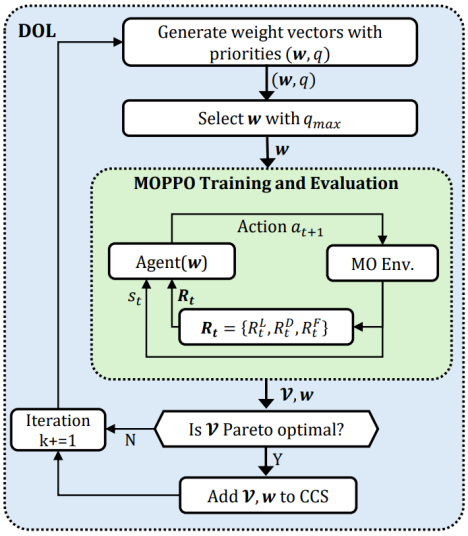


FIGURE 12 - TRAINING LOOP FOR DETERMINING THE CCS USING THE DEEP OPTIMISTIC LINEAR SUPPORT ALGORITHM (Lautenbacher et al. (2025)).

then degrades linearly when the relative weights of competing objectives is increased. The piecewise function defined by the maximum sum of rewards from each policy defines the current CCS.

The OLS algorithm then identifies "corner weights", where the edges from each policy meet. These locations in weight space offer the best opportunity to improve the convex hull of the CCS, and therefore efficiently reach the optimal convex coverate set. Thus, in the next iteration, a new policy is trained using the weights defined at each corner weight location. If this policy improves the convex hull (as it does in the example), then it is added to the convex coverage set. New corner weights are identified and ordered by their potential q for improving the CCS, and the procedure is repeated until reaching a predefined stopping condition, or after a fixed number of iterations. When the OLS algorithm is used to train one or more neural networks, the method is referred to as "deep" OLS.

The complete deep optimistic linear support procedure is illustrated in Fig. 12. The inner loop, describing training and evaluation using a multi-objective proximal policy optimization (MOPPO), is independent from the linear support loop, and can be replaced by any desired environment, agency and optimization strategy.

5.3. THE AI4REALNET MULTI-OBJECTIVE PACKAGE

The following describes the current version of the domain-agnostic multi-objective reinforcement learning package, as well as the plans for further development and implementation in the AI4REALNET use cases. The AI4REALNET-MORL package is the outcome of research on how to address the multi-objective optimization problem within the framework of Reinforcement Learning solutions, as well as the extrapolation of a common baseline solution to different Use Cases. This project is currently avail-

able in the internal GitLab repositories, and the final version is expected to be made public at the time of the official release.

The package `ai4realnet-morl` employs the `MO-Gymnasium` package for working with multi-objective environments, and the associated `morl-baselines` software (Felten et al. (2023)). The Deep Optimistic Linear Support algorithm is used to create a convex coverage set, as described in the previous section.

The software itself is demonstrated using a **multi-objective proximal policy optimization** (PPO) technique, implementing an actor-critic network and optimizing using advantage to improve stability. Although this choice of network architecture and optimization are implementations of a popular paradigm, the software can accommodate an arbitrary multi-objective optimizer / agent pair. At each iteration of the OLS, a critic agent learns a vector of advantages corresponding to each objective, which is then scalarized using a fixed set of weights and used for the actor training. After each loop of the OLS algorithm, the policy π and weights of the neural networks are stored before moving to the next iteration. The deep OLS algorithm offers the opportunity to "warm-start" an iteration by setting the weights of the current iteration to the weights of the closest previously-trained policy.

Future directions. The `ai4realnet-morl` has been successfully demonstrated in the electrical grid use-case and documented in Lautenbacher et al. (2025). Implementations for the BlueSky and Flatland environments are planned for the future. This code deployment aims at developing a solution that is agnostic to the decision-making environment, so that the specific properties of each use case can be inserted and processed as augmented arguments. Future development plans include adjustments to improve interoperability between use-cases, and reduce the need for customization on an individual use-case basis. Finally, plans to integrate multi-objective functionality into the AI4REALNET's InteractiveAI framework are under development as well.

6. INTERACTIVE AI TO AUGMENT DECISION-MAKING

Operational environments across aviation, mobility, and complex network management are increasingly shaped by uncertainty, variability, and multi-factor trade-offs. Traditional automation seeks to replace or replicate human decision-making, but such fully autonomous approaches remain brittle when confronted with contextual nuances, tacit knowledge, or rapidly shifting operational conditions. Recent European regulatory and policy developments, including the EU Ethics Guidelines for Trustworthy AI and IATA's guidance on human-in-command operations, highlight the need for systems that support human decision-makers rather than substitute them.

Against this backdrop, interactive AI emerges as an alternative paradigm. Instead of static, offline-trained models, interactive AI systems evolve through real-time dialog with human operators. They do not merely deliver answers; they help humans reason about complex options, reveal trade-offs, and adapt to user input. This chapter introduces the principles of interactive AI for decision augmentation, focusing on human–AI co-learning, interactive learning loops, sparse expert data, inverse reinforcement learning to elicit human rewards, evolutionary strategies for generating new alternatives, and the role of the human–machine interface (HMI) as a medium for communication and understanding.

6.1. HUMAN-AI COLLABORATION CONCEPTS

The field of human-machine interaction is one with many names for similar concepts, oftentimes used interchangeably despite slight difference in meanings. Usually the terms begin with Human-Machine (the "machine" sometimes substituted with "AI") and some suffix indicating the collaborative nature of the system described, such as "teaming", "interaction" or "collaboration". The term "collective intelligence" is also used to refer to human-AI teams. In the context of the AI4REALNET project, we refer to the interaction of human and AI agents as collaboration through mixed initiative.

Effective human–AI collaboration is built on the premise that humans and AI systems bring complementary strengths to complex decision-making. Humans excel at contextual reasoning, ethical judgment, dealing with ambiguity, and adapting to unforeseen situations. AI systems, in contrast, offer scalability, rapid computation, multi-objective optimization, and continuous monitoring. Human–AI collaboration aims to combine these strengths in a way that enhances decision quality, maintains human oversight, and ensures that autonomous contributions remain aligned with operator goals.

Within the broader paradigm of mixed-initiative collaboration, both the human and the AI may contribute to planning, monitoring, learning, or acting, depending on the context. Mixed-initiative systems enable flexible transfer of control, shared interpretation of tasks, and fluid interaction between

human expertise and machine intelligence. Within this paradigm, AI4REALNET considers two key mechanisms: **interactive human–AI learning** and **human–AI co-learning**, which represent closely related but distinct forms of mutual adaptation and engagement.

6.1.1. INTERACTIVE HUMAN-AI LEARNING

Interactive learning refers to collaborative scenarios in which humans play an active role in shaping the AI's learning trajectory. The adaptation flows primarily toward the AI, which incorporates human feedback, corrections, and preferences to refine its internal models. The human provides guidance, but their own decision strategies remain largely unchanged. In this mode, the AI adapts to human input, and human behavior is not fundamentally altered. Feedback can be implicit (e.g., acting differently to a proposed solution) or explicit (e.g., choosing a preferred option). Finally, the interaction typically occurs at discrete points when the AI requests guidance or when the human intervenes.
Interactive human–AI learning becomes particularly important when data is scarce, incomplete, or difficult to label—a common situation in critical infrastructure management. In many operational contexts, historical data is limited, inconsistent, or not representative of rare events and edge cases. Human demonstrations may also be sparse because expert time is limited and because certain situations (e.g., emergencies, rare failures, extreme network states) cannot be safely or frequently recreated. Here, interactive learning provides a mechanism for the AI to obtain targeted, high-value guidance at key moments, allowing it to generalize beyond the available training samples.
Interaction is also essential when:

- **Human preferences are nuanced or context-dependent**, meaning they cannot be easily captured in a static reward function.
- **The AI encounters unfamiliar or ambiguous states**, where human steering prevents undesirable outcomes.
- **The system must learn operator-specific styles or heuristics**, such as risk appetite, prioritization strategies, or domain-specific trade-offs.
- **Real-time adaptation is required**, for instance in dynamic network states where human judgment provides clarity not encoded in the model.

In this mode, the nature of the human input may vary:

- **Direct feedback**, such as correcting a proposed solution or indicating which option is preferred.
- **Interaction signals**, such as adjusting interface elements or selecting certain constraints, which the AI can use as preference signals.

- **Sparse or occasional interventions**, especially in real-time settings where continuous oversight is not feasible.

These forms of limited yet meaningful human input help the AI improve its alignment with operator expectations, which cannot always be fully specified in advance through formal goals or constraints, even when large annotated datasets or extensive demonstrations are unavailable. As an example, in AI4REALNET's dynamic airspace sectorization use case (defined in WP1), where expert demonstrations are sparse, the operator selects one of several Pareto-optimal solutions generated by AI. The AI updates its preference model based on this selection, learning which trade-offs the operator values. The operator's overall approach remains stable while the system learns to generate solutions that are more aligned to human preferences.

6.1.2. HUMAN–AI CO-LEARNING

Co-Learning is a less common term in the literature, yet it concisely describes the goal of the systems being developed: mutually beneficial interaction between the human and AI agents in a system. Co-learning extends beyond interactive learning by fostering mutual adaptation. Here, both the human and the AI refine their strategies, mental models, and expectations as a result of their interaction. The AI learns from human decisions, and the human also learns from AI explanations, alternative solutions, and insights derived from large-scale data or simulation. In this mode, adaptation is bidirectional where both agents evolve through interaction. AI functions are informed by insights from the psychology and cognitive science literature on human learning, and explicitly support exploration and experimentation, allowing the human to extend their understanding of the system's dynamics. The human can thereby develop improved situational awareness and decision strategies, while the AI becomes more attuned to human goals.

An example of such an interaction in the railway example is the management of rerouting trains after a storm has disrupted traffic on a track. The AI system evaluates the effects of a human decision and provides a statistical evaluation, as well as providing alternative solutions with the same evaluation. This allows the operator to compare their hypothesis (solution) with alternatives and identify the best course of action. After the event, the system provides additional functions with which the operator can re-analyze their chosen solution, spend more time exploring the system dynamics and simulate alternative actions. Self-reflection is an important part of learning (Kolb and Kolb (2009)), and the AI system supports this process with a self-reflection module. Logging and anonymizing this data allows it to be evaluated and shared with other operators, enabling knowledge sharing throughout the organization.

Abich and Sikorski define co-Learning as "*an iteratively emergent process in which group members adapt their behavior and provide feedback to each other*", highlighting that the agents adapt their be-

havior to each other over time and engage in bidirectional communication with the goal of establishing mutual understanding (Abich and Sikorski, 2023). The authors list the following factors required for effective human-AI co-learning:

1. **Trust:** must be established and maintained within the human-AI team, defined as the willingness of one party to be vulnerable to the actions of the teammate, regardless of their ability to monitor or control that party (Mayer et al., 1995).

2. **Common Ground:** refers to the establishment and maintenance of a shared understanding of the task and each other.

3. **Group Awareness:** describes the team dynamics, which encompasses the division of expertise, understanding of social structures, knowledge on group member's information, opinions and intentions.

4. **Communication:** the minimum requirement for co-learning is explicit bidirectional communication between human and AI agent, allowing for knowledge and experiences to be shared between the team members.

5. **Mutual Adaptation:** as mentioned within the initial co-learning definition, the adaptation over time in a continuous learning loop is one of the key differences between co-learning and other similar fields. This mutual adaptation means that the AI can adapt to individual human's preferences and needs, aligning its strategy with theirs. On the other side, humans can improve their own performance and evolve their understanding and perception of the AI agent.

In a similar thread, van den Bosch et al. provide a detailed description of co-learning as well as proposing a series of principles thereof in their 2019 paper titled "*Six Challenges for Human-AI Co-Learning*". While named differently, the meaning of principles described by the authors strongly overlap with those listed above. The authors extend the list to include the dynamic of human-AI teams learning from each other, for example via a knowledge base shared by all AI agents. Beyond these principles, the authors develop six models a co-learning-capable AI agent must posses, corresponding to its six key challenges; taxonomy model, team model, task model, self-model, theory-of-mind model and communication model.

For any team to function, a common language and a shared understanding of team dynamics is required. To structure the language, the taxonomy model manages the shared language, pertaining to concepts and relations important for a common understanding of the task. This taxonomy can then be used by the communication module to enable mutually understandable interaction. The "rules" of this interaction is managed by the human agent via the team model, which defines work agreements, team organization, hierarchy, task distribution, and delegation. With a common taxonomy and the

agent's place in the team defined, it can begin to solve tasks. To do so, a task model is required, which is comprised of knowledge about the task and the relations between states, actions, and outcomes, including solution strategies and representation of state knowledge.

The final two models proposed by the authors allow for the artificial agent to understand itself and its team members: the self-model, depicting the inner state of the artificial agent, and the "Theory of Mind"-model, which covers knowledge about the inner state of other agents. Both models contain information about the goals, values, capabilities, resources, and intentions of the agents. The Theory-of-Mind-model differentiates itself from the self-model in that the information can be provided directly by the human agent or inferred by the artificial agent through behavioral observation. It also considers aspects of emotion and personality. The knowledge of self and of others enables productive alignment and adaptation within the team (van den Bosch et al., 2019).

Generally it can be said that the vast majority of the literature is concerned with defining requirements for and identifying potential methods for co-learning. There are few studies which develop systems for co-learning.

6.1.3. ARCHITECTURAL CONSIDERATIONS

Interactive AI systems that support interaction and co-learning require a modular and extensible architecture capable of enabling continuous bidirectional interaction. Unlike conventional autonomous optimization pipelines, such systems operate within a closed feedback loop in which human expertise, preferences, and contextual knowledge iteratively shape algorithmic behavior, while AI-generated solutions inform human understanding and decision-making. To support this co-learning paradigm, the architecture must allow real-time feedback integration, solution visualization, and adaptive learning across sessions. The A3S solution introduced in Section 3 plays an important role in realizing this paradigm by providing supportive functions for exchanging information between human operators and AI agents, enabling human inputs and observations to be incorporated into the learning and optimization process. A schematic overview of the overarching architecture is portrayed in Figure 13.

At the center of this architecture lies the **Human–Machine Interface (HMI)**, which enables users to inspect, compare, manipulate, and propose candidate solutions. The HMI should present AI-generated outputs in domain-relevant visual forms, such as airspace sector layouts, railway or power grid topologies, or workload distributions, ensuring that humans can interpret and evaluate solutions operationally. In addition to visualization, the interface must support interaction mechanisms through which humans can revise, reject, augment, or nudge AI-generated solutions, as well as label configurations as preferred or undesirable. These interactions form structured feedback signals that guide subsequent optimization. Capabilities such as *TraceRL* described in Section 3 can augment such interfaces by providing visualization overlays that expose internal aspects of the learning and decision process.

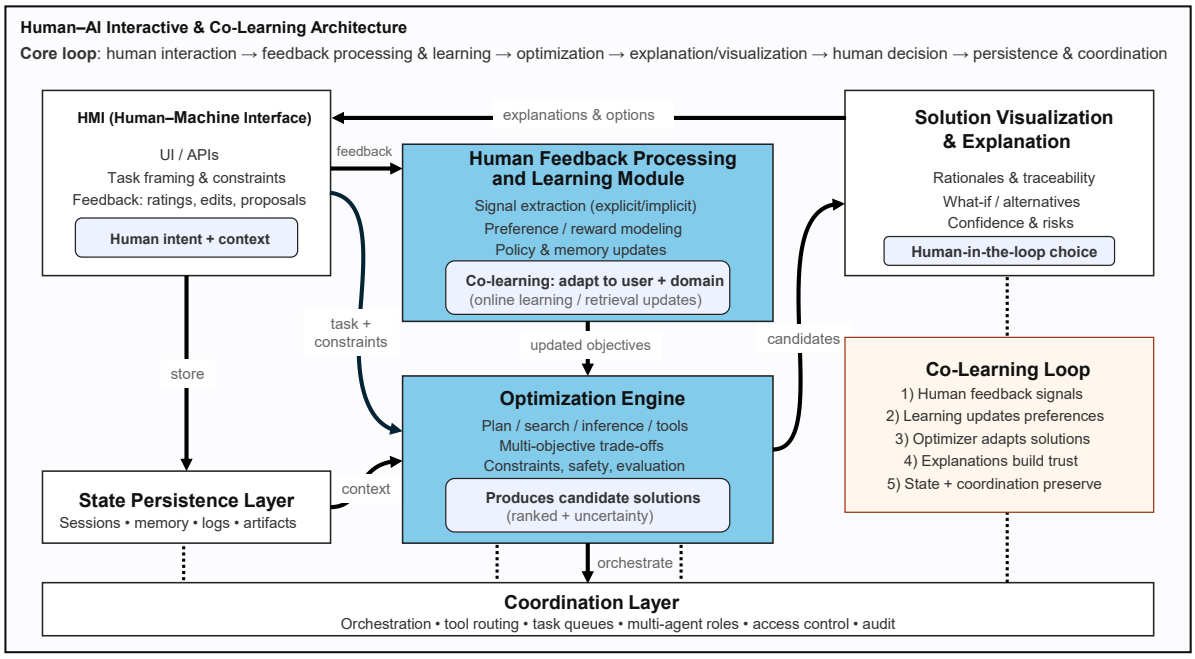


FIGURE 13 - HIGH LEVEL INTERACTIVE AND CO-LEARNING ARCHITECTURE.

Rather than constituting a standalone HMI, TraceRL can be integrated into an existing interface as an analytical layer that visualizes agent trajectories, decision traces, or reward signals, thereby helping human operators interpret system behavior and provide more informed feedback.

A dedicated **human feedback processing and learning module** needs to interpret these signals and converts them into machine-interpretable guidance, such as preference rewards, constraints, or biasing of the search distribution. This module enables the AI system to adapt dynamically to human intent and may include mechanisms for preference learning, meta-learning, or continual learning across sessions. Complementing this is the **optimization engine**, responsible for generating candidate solutions under both intrinsic objectives and human-derived guidance, and capable of being paused, resumed, or redirected in response to user interaction.

The architecture also requires a **solution visualization and explanation component** that translates abstract outputs into interpretable domain structures and highlights trade-offs, uncertainties, or the effects of human feedback. A **state persistence layer** stores interaction history, learned preferences, and system parameters, enabling continuity and cumulative learning across sessions. Finally, a **coordination layer** ensures seamless communication between modules and supports extensibility so that new learning or interaction components can be integrated without redesign.

Such a modular architecture provides the structural foundation necessary for interactive learning and co-learning, enabling flexible and transparent collaboration in which human expertise and AI-driven optimization continuously inform and refine one another.

6.1.4. CONCLUDING REMARKS

The preceding discussion outlined architectural and interaction-level requirements for enabling interactive learning and co-learning between humans and optimization-driven AI systems. In such settings, optimization is embedded within a continuous feedback loop in which human expertise and algorithmic search mutually influence one another. This co-adaptive paradigm requires both technical infrastructure for real-time interaction and a conceptual framework that treats human–AI collaboration as a bidirectional learning process linking architectural design with concrete algorithmic methods.

In AI4REALNET, two complementary learning threads define this perspective. The first, *AI learning from humans*, captures how human judgments, preferences, and domain knowledge shape the behavior of the optimization system. Through labeling solutions, steering search trajectories, or introducing constraints, humans progressively guide the optimizer toward operationally acceptable regions of the solution space. This alignment can occur through real-time adaptation, preference shaping, and meta-learning within the optimization loop rather than through heavy offline training.

The second thread, *humans learning from AI*, reflects how interaction with AI-generated solutions enhances human insight and decision-making. By exploring candidate solutions, visualizing trade-offs, and observing system responses to feedback, human operators develop a deeper understanding of the design space and refine their own preferences and strategies. The AI thus functions not only as an optimizer but also as an exploratory partner and decision-support tool.

Together, these threads establish a co-learning framework in which humans and AI continuously adapt to one another. Architecturally, this requires modular integration of optimization engines, preference-learning components, visualization interfaces, and persistent memory. Methodologically, it motivates algorithms that remain robust under sparse and evolving human input while maintaining efficient exploration. The following sections build on this foundation by presenting specific computational methods and implementation approaches that operationalize these principles in practice.

6.2. AI LEARNING FROM HUMANS

AI learning from humans is a key theme in the AI4REALNET project, focusing on building systems that can continuously improve through human interaction and feedback. Rather than relying solely on static training data, AI4REALNET explores human-in-the-loop and co-learning approaches where expert input, user preferences, and operational experience are integrated into adaptive learning cycles. These solutions enable AI systems to refine decision-making, optimize performance, and maintain alignment with human goals in complex, dynamic environments such as energy, infrastructure, and networked systems. This section outlines the specific methods used in AI4REALNET, focusing on the AI-side approaches for learning from human input and feedback.

6.2.1. INVERSE REINFORCEMENT LEARNING

Inverse Reinforcement Learning (IRL) is a paradigm in machine learning and sequential decision-making that seeks to infer the underlying objectives or preferences driving observed behavior (Ng and Russell, 2000). Unlike traditional reinforcement learning (RL), which assumes the reward function is given and seeks an optimal policy, IRL takes demonstrations as input and aims to recover the latent utility function that best explains the demonstrated actions. The core idea is that behavior—whether of humans, animals, or automated systems—is shaped by optimization of some objective, and understanding that objective can enable imitation, adaptation, and transfer to new settings.

6.2.1.1 Formal Setup IRL is typically formalized in a Markov Decision Process (MDP), as introduced in Section 5. An expert generates demonstrations $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$, where each trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$ is assumed to be induced by a policy π_E that is optimal or near-optimal with respect to the unknown reward r.

Given a candidate reward function r_θ parameterized by θ, the corresponding action-value function is

$$Q^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_\theta(s_t, a_t) \mid s_0 = s, a_0 = a, \pi\right],$$

and the optimal policy satisfies

$$\pi_\theta^* \in \arg\max_\pi \ \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_\theta(s_t, a_t)\right].$$

The goal of IRL is to recover r_θ such that the demonstrated behavior is optimal or approximately optimal under π_θ^*.

6.2.1.2 Ambiguity The practical and theoretical challenges of IRL distinguish it from standard RL. A central difficulty arises from *ambiguity*: many different reward functions can rationalize the same behavior. For example, a reward that is a positive scaling or simple transformation of the true utility can induce identical policies, making exact recovery impossible without additional constraints.

Formally, the IRL problem is ill-posed because the mapping $r \mapsto \pi_r^*$ is many-to-one. For instance, for any reward r and any $\alpha > 0$, $\beta \in \mathbb{R}$, the transformed reward

$$\tilde{r}(s,a) = \alpha r(s,a) + \beta$$

induces the same set of optimal policies as r. As a result, IRL algorithms typically recover rewards only up to an equivalence class unless additional constraints or priors are imposed.

To address this, research has proposed solution concepts and regularizations that constrain the space of plausible utilities. One perspective frames IRL as seeking a reward that induces behavior consistent

with a set of optimality or near-optimality criteria, often under assumptions about the demonstrator's rationality and stochasticity. Maximum entropy formulations (Ziebart et al., 2008) exemplify this approach by resolving ambiguity through preference for distributions that explain demonstrations while remaining maximally non-committal beyond observed data.

6.2.1.3 Feasible Sets Recent theoretical treatments have formalized these intuitions, providing frameworks to characterize the identifiability of rewards and develop algorithms with provable guarantees. For instance, Metelli et al. (2023) articulated foundational results on the structure of the IRL problem, its solution sets, and conditions under which particular reward representations can be consistently learned from data. This work clarified ambiguities in earlier formulations and set the stage for subsequent developments in efficient IRL algorithms.

A key axis of development has been the characterization of solution concepts that balance fidelity to demonstrations with statistical and computational tractability. In this context, Lazzati et al. (2024a) introduced new solution concepts for *Offline IRL*, providing algorithms that operate with finite demonstration datasets and establish efficiency guarantees under realistic sampling assumptions. Their work shows that definitions of what it means to recover a reward can lead directly to methods that are both statistically sound and computationally feasible.

Another frontier is extending IRL to settings with large or high-dimensional state spaces, where naive approaches suffer from the curse of dimensionality. Addressing this, as a part of the AI4REALNET project, Lazzati et al. (2024b) proposed provably efficient techniques that scale inverse learning to expansive environments, leveraging structure in the dynamics and representations that support generalization. These contributions help bridge the gap between theoretical IRL and applications in complex domains like robotics and autonomous driving.

6.2.1.4 Risk-Sensitive Inverse Reinforcement Learning Traditional IRL implicitly assumes that the underlying utility reflects expected performance under uncertainty. However, many real agents exhibit *risk-sensitive* behaviors: they may weigh outcomes not merely by expectation but also by variability, worst-case scenarios, or other risk criteria. Risk-sensitive IRL extends classical formulations to capture these nuanced preferences, building on broader connections between IRL, utility theory, and decision-making under uncertainty.

A notable recent contribution in this direction, as a part of the AI4REALNET project, is by Lazzati and Metelli (2025), who formalize the problem of learning utilities from demonstrations in Markov Decision Processes with risk considerations. Their approach acknowledges that demonstration data may embody not just preferences over rewards but also attitudes toward risk and uncertainty. By broadening the IRL framework to encapsulate risk sensitivity, this work provides tools to derive utility representations that align with demonstrators' behavior under uncertainty. This is particularly salient in

safety-critical domains—such as finance, healthcare, and autonomous systems—where risk attitudes materially affect decisions.

Risk-sensitive IRL introduces additional challenges beyond standard reward inference. The learner must disentangle the objective function from risk profiles embedded in the behavior. For instance, two agents might exhibit similar actions because of different reward scales and opposing risk attitudes. Approaches like those in Lazzati and Metelli (2025) begin to articulate solution concepts and learning methods that can separate these factors, enabling more expressive and accurate modeling of demonstrator utility.

6.2.1.5 Discussion and Future Directions As IRL research matures, several themes emerge. First, theoretical clarity around solution concepts and identifiability is essential for both understanding what can be learned and designing effective algorithms. Work such as Metelli et al. (2023) and Lazzati et al. (2024a) ground the field in rigorous foundations that support further innovation.

Second, practical scalability and robustness remain active areas of exploration. Approaches that address large state spaces (Lazzati et al., 2024b) and leverage sub-optimality as informative variation (Poiani et al., 2024) illustrate how IRL can adapt to real-world complexities.

Third, extending IRL to capture richer preference structures, such as risk sensitivity, opens avenues for modeling and imitating behavior in environments where agents care about more than expected reward. As applications grow in autonomy and human-machine interaction, integrating risk, ambiguity, and safety criteria into inverse learning will be increasingly important.

In conclusion, Inverse Reinforcement Learning has evolved from conceptual frameworks to a vibrant area of theoretical and algorithmic research. By combining foundational insights with new models of utility and behavior, the field continues to advance our ability to reveal latent objectives from observed actions.

6.2.1.6 AI4REALNET-Relevant Applications Within the AI4REALNET project, IRL provides a principled framework for discovering latent operational objectives in complex, networked real-world systems where reward functions are neither explicitly specified nor centrally available. For instance, in smart energy grids, mobility networks, or distributed logistics systems, human operators and legacy controllers implicitly balance efficiency, robustness, cost, and safety. By observing historical trajectories of system states and interventions, IRL can infer surrogate reward representations that capture these trade-offs, enabling data-driven digital twins and decision-support tools. In safety-critical infrastructures, risk-sensitive IRL is particularly aligned with AI4REALNET goals: operators often exhibit conservative or precautionary behavior under uncertainty, and modeling such risk profiles is essential for realistic simulation and policy transfer. Moreover, offline and large-scale IRL methods are directly relevant in settings where only logged data from networked sensors and controllers are available, and

interaction with the live system is constrained. This supports AI4REALNET's emphasis on trustworthy, interpretable, and data-efficient learning for real-world networked environments.

6.2.2. PREFERENCE-BASED REINFORCEMENT LEARNING

Reinforcement Learning traditionally relies on numerical reward signals to guide an agent toward desirable behavior. In many real-world applications, however, specifying an accurate reward function is difficult, unintuitive, or even infeasible. Preference-Based Reinforcement Learning (PbRL) addresses this limitation by replacing explicit rewards with comparative feedback, such as preferences between trajectories, actions, or outcomes.

The central idea of PbRL is that while humans and other decision-makers may struggle to provide scalar evaluations, they are often reliable when expressing relative judgments. For instance, an expert may indicate that one behavior is preferable to another without assigning numerical scores. PbRL leverages this form of feedback to learn policies that align with latent preferences, enabling learning in domains such as robotics, human–computer interaction, and decision support systems.

PbRL is closely related to Inverse Reinforcement Learning, as both aim to infer latent objectives from indirect signals. However, PbRL typically treats preferences as the primitive feedback signal rather than demonstrations of full behavior. This distinction leads to different modeling assumptions, learning objectives, and theoretical challenges.

6.2.2.1 Formal Setup In Preference-Based Reinforcement Learning, the learner does not observe numerical rewards but instead receives preference feedback over pairs of trajectories. Let τ_i, τ_j denote two trajectories and $\tau_i \succ \tau_j$ indicate that τ_i is preferred.

Preferences are commonly modeled via an underlying latent utility function $U(\tau)$. Under a Bradley–Terry–Luce (BTL) or logistic preference model, the probability of observing $\tau_i \succ \tau_j$ is

$$\mathbb{P}(\tau_i \succ \tau_j) = \frac{\exp(U(\tau_i))}{\exp(U(\tau_i)) + \exp(U(\tau_j))}.$$

A standard assumption is that utilities decompose additively along trajectories,

$$U(\tau) = \sum_{t=0}^{T} r_\theta(s_t, a_t),$$

where r_θ is a latent reward function.

Early work in PbRL formalized the problem as learning from pairwise comparisons, often under assumptions inspired by utility theory. Preferences are commonly interpreted as noisy observations of an underlying latent utility function that ranks trajectories or actions. This interpretation allows PbRL to connect naturally with classical preference learning and ordinal regression.

A comprehensive overview of these foundations is provided by Wirth et al. (2017), who survey preference-based learning methods in reinforcement learning settings.

From a practical standpoint, feedback efficiency is a central concern. Since preferences are typically elicited from humans, minimizing the number of queries is crucial. Techniques such as active query selection and semi-supervised learning have been explored to reduce reliance on explicit feedback while maintaining performance.

Given a dataset of preference comparisons $\mathcal{P} = \{(\tau_i, \tau_j, y_{ij})\}$, where $y_{ij} = 1$ if $\tau_i \succ \tau_j$ and 0 otherwise, the reward parameters can be learned by maximizing the log-likelihood

$$\mathcal{L}(\theta) = \sum_{(\tau_i,\tau_j,y_{ij})\in\mathcal{P}} y_{ij} \log \mathbb{P}(\tau_i \succ \tau_j) + (1 - y_{ij}) \log \mathbb{P}(\tau_j \succ \tau_i).$$

The learned reward is then used to optimize a policy

$$\pi_\theta^* \in \arg\max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_\theta(s_t, a_t) \right].$$

6.2.2.2 Theoretical Perspectives and Guarantees Despite its intuitive appeal, PbRL presents substantial theoretical challenges. Unlike standard RL, where rewards define a clear optimization objective, preference-based settings require careful definitions of optimality and regret. The absence of absolute rewards complicates performance guarantees and raises questions about identifiability of optimal policies.

Several works have studied PbRL from a theoretical standpoint, analyzing convergence, sample complexity, and regret under various preference models. Bayesian approaches, such as dueling posterior sampling (Novoseller et al., 2019), provide principled ways to handle uncertainty over latent utilities while guiding exploration in preference space.

Nevertheless, much of the early theory focused on bandit or simplified sequential settings. Extending these insights to general Markov decision processes has remained an open challenge, motivating recent research on the theoretical foundations of preference-based sequential decision making.

6.2.2.3 Sequential Decision Making with Preference Feedback A significant step toward a unified theoretical understanding of PbRL in sequential environments is provided by Drago et al. (2025). This work formalizes sequential decision making with preference feedback and studies the relationship between preferences, utilities, and policies in Markov decision processes.

The authors clarify how preference signals can be interpreted consistently across time, addressing ambiguities that arise when preferences are expressed over partial trajectories or outcomes. By grounding preference feedback within a rigorous decision-theoretic framework, this work connects PbRL more tightly to reinforcement learning theory.

Importantly, the paper highlights conditions under which learning from preferences is well-posed and when meaningful performance guarantees can be established. This perspective helps bridge the gap between practical PbRL algorithms and their theoretical justification, offering guidance for algorithm design and evaluation.

6.2.2.4 Discussion and Outlook Preference-Based Reinforcement Learning represents a compelling alternative to reward-centric learning paradigms, particularly in domains where human judgment plays a central role. By relying on relative feedback, PbRL aligns more naturally with how preferences are expressed in practice, but this flexibility comes at the cost of additional ambiguity and theoretical complexity.

Recent progress has strengthened the theoretical foundations of PbRL, especially in sequential settings, and clarified its connections to inverse reinforcement learning and utility theory. As demonstrated by Drago et al. (2025), rigorous analysis of preference feedback can yield principled learning objectives and performance guarantees.

Future research directions include improving scalability to high-dimensional environments, integrating richer forms of human feedback, and developing unified frameworks that encompass rewards, preferences, and risk-sensitive objectives. As these challenges are addressed, PbRL is poised to play an increasingly important role in the design of interactive and human-aligned learning systems.

6.2.2.5 AI4REALNET-Relevant Applications PbRL naturally complements AI4REALNET's human-in-the-loop perspective by enabling structured incorporation of expert judgment into the control of real-world networks. In domains such as traffic management, energy dispatch, or emergency response coordination, stakeholders may disagree on precise reward definitions but can reliably express comparative assessments (e.g., preferring one congestion pattern or load-balancing strategy over another). PbRL allows these qualitative preferences to shape policy learning without requiring explicit scalarization of multi-objective criteria such as sustainability, equity, resilience, and cost. Furthermore, active preference elicitation can be integrated with simulation-based digital twins of networked systems, enabling iterative refinement of policies through expert feedback before deployment. In distributed or multi-agent network settings, PbRL also offers a mechanism to reconcile heterogeneous stakeholder utilities, aligning learned policies with socially acceptable trade-offs. As such, PbRL supports AI4REALNET's broader objective of developing adaptive, transparent, and human-aligned AI systems for complex real-world networks.

6.2.3. SHAPING AI BEHAVIOR TO OPERATIONAL "BEST PRACTICES"

Achieving a level of human alignment must begin at the training stage. AI systems must be trained on data and evaluation criteria that reflect real operational practices, expert judgment, and the contextual nuances of decision-making in safety-critical domains. Incorporating human feedback and/or expertise, operational constraints, and safety-focused performance metrics during development ensures that the AI model learns not only what is technically possible, but also what is operationally acceptable. Without such deliberate training and validation, even highly capable systems may produce recommendations that diverge from human expectations, undermining trust and limiting their adoption in environments where human-centered safety remains paramount. In the previous sections on IRL and PbRL, the focus was on inferring the reward structure from human demonstrations and using this structure to guide RL behavior toward human-preferred directions. In this section, alternative mechanisms for guiding RL behavior under a fixed, designed reward structure are investigated.

In the context of AI4REALNET, reinforcement *learning* can be substantially accelerated and steered toward operationally acceptable behavior by incorporating additional sources of structure beyond sparse trial-and-error rewards and reward shaping. One such mechanism is **action shielding**, which restricts the agent's action space online by filtering out commands that are known *a priori* to violate hard safety constraints stemming from operational knowledge (e.g., predicted loss of separation in aircraft conflict resolution), see Alshiekh et al. (2018). By ensuring that exploration remains within a safe and feasible region of the state–action space, shielding reduces the risk of catastrophic outcomes while preserving meaningful learning signals. **Human feedback**, provided either as corrective guidance or as preferences over actions or trajectories, can further shape the learning process by biasing value estimates or policy updates toward behaviors that align with domain expertise, particularly in situations where the reward function alone is ambiguous or underspecified (Knox and Stone, 2009). **Expert demonstrations** (Piot et al., 2014), derived from rule-based heuristics or human-designed solutions in the form of 'best practices', offer another complementary source of guidance by seeding learning with human strategies and tacit operational norms that are known to be effective, while still allowing the AI agent to refine or improve upon them through continued interaction with the environment.

Together, these mechanisms reduce the effective exploration burden, improve sample efficiency, and encourage convergence toward solutions that are both performant and consistent with established operational practices. In this section, these ideas are instantiated using a Q-learning agent for a representative two-dimensional aircraft conflict-resolution task (see Figure 14), which serves as a concrete example to illustrate how shielding, human feedback, and expert demonstrations can be integrated to shape RL behavior in a safety-critical domain. These possibilities are integrated into a single RL architecture (Figure 15), enabling the evaluation of each technique both individually and in combination.

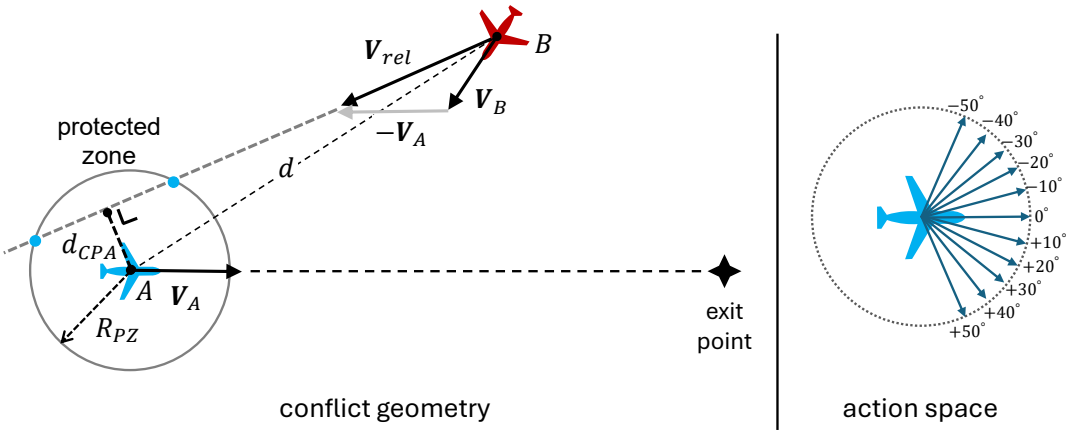


FIGURE 14 - AIRCRAFT CONFLICT GEOMETRY AND ACTION SPACE.

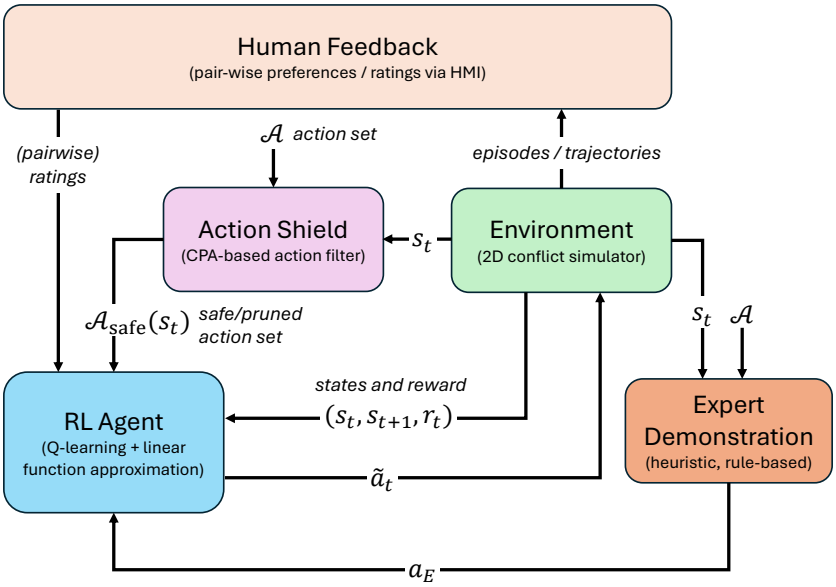


FIGURE 15 - SINGLE RL ARCHITECTURE WITH PRE-SELECTION (PRE-SHIELDING) ACTION FILTER, HUMAN FEEDBACK AND LEARNING FROM EXPERT DEMONSTRATIONS. EACH SHAPING TECHNIQUE CAN BE ACTIVATED BOTH INDIVIDUALLY AND IN COMBINATION.

For this example, a linear Q-learning with function approximation is employed (Melo and Ribeiro, 2007), where the action-value function is parameterized as:

$$Q(s, a) = \mathbf{w}_a^\top \boldsymbol{\phi}(s), \tag{14}$$

with feature vector $\boldsymbol{\phi}(s) \in \mathbb{R}^F$ and action-specific weight vectors $\mathbf{w}_a$. The standard Q-learning update for a transition (s_t, a, r, s_{t+1}) is:

$$\delta = r + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t), \tag{15}$$

$$\mathbf{w}_a \leftarrow \mathbf{w}_a + \alpha \left(\delta \, \boldsymbol{\phi}(s) - \lambda \mathbf{w}_a\right), \tag{16}$$

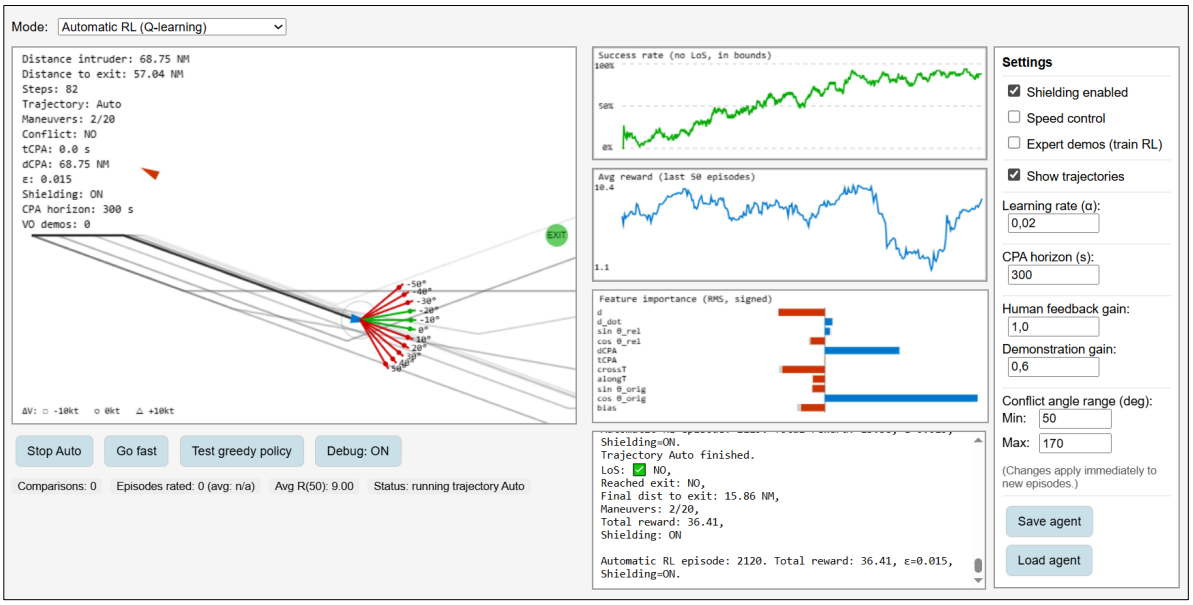


FIGURE 16 - HTML/JAVASCRIPT DEMONSTRATION APPLICATION.

where r is the reward per step, α is the learning rate, γ the discount factor, $\lambda > 0$ the L2 regularization coefficient. Exploration is performed with ε-greedy over the action set.

The proposed framework is implemented as an interactive JavaScript/HTML application that enables real-time visualization and analysis of learning-based aircraft conflict detection and resolution in a two-dimensional airspace (see Figure 16). The application displays the simulated environment, including ownship and intruder trajectories within a 150×100 Nautical Miles (NM) sector, along with configurable control options for training and evaluation (e.g., exploration rate, shielding, and expert demonstrations). During execution, the interface provides live plots of key learning metrics such as episode reward, success and failure rates, maneuver counts, and feature importance derived from the learned Q-function, allowing the evolution of the policy to be monitored over time. A detailed, step-by-step event log further supports traceability and debugging by recording actions, rewards, and termination causes. Training scenarios are generated by sampling randomized conflict geometries, including relative positions, headings, and speeds, ensuring exposure to a diverse set of encounter configurations while maintaining a controlled and reproducible experimental setup. Table 1 summarizes the hyperparameters set in the demonstration application. These parameters have been tuned based on several trial and error iterations.

When trained using plain Q-learning with linear function approximation and the proposed feature set and reward function, the agent typically requires on the order of **3,000–4,000 episodes** to converge to a reasonably stable policy. During early training, learning is dominated by the bias feature, indicating that the agent primarily exploits global reward structure before discovering meaningful state–action correlations. As training progresses, useful conflict resolution behavior emerges gradually, but con-

TABLE 1 - HYPERPARAMETER SETTINGS

Parameter	Value
Learning rate α	0.02
Discount factor γ	0.99
Initial exploration rate ε_0	0.40
Exploration decay (per episode)	0.995
Minimum exploration rate ε_{min}	0.015
Maximum episode length	200 steps
Maneuver budget N_{event}	20 events
L2 regularization coefficient λ	10^{-4}

vergence remains sensitive to reward scaling and exploration scheduling. After having learned conflict resolution, a reward gating mechanism shifts the focus more towards operational performance to restore the trajectory close to its initial route and exit the airspace from the appropriate side.

6.2.3.1 Action Shielding To speed up learning and to prevent the RL agent from taking unsafe or obviously suboptimal actions, a *pre-selection action shielding* mechanism is employed (Könighofer et al., 2020), see Figure 15. This means that action filtering occurs *before* the policy selects an action. Thus, the agent samples from a reduced action set

$$\mathcal{A}_{\text{safe}}(s) \subseteq \mathcal{A},$$

and the unsafe actions are never passed to the Q-learning update. Action shielding is most common in the field of *Safe Reinforcement Learning*.

Here, an action a is considered safe if and only if all of the following geometric constraints hold:

1. **Immediate separation constraint:** Simulate the next-step relative positions; if the action would result in immediate loss of separation ($d_{t+1} < 5$ NM), then a is rejected.

2. **Closest-Point-of-Approach (CPA) monotonicity constraint:** Let d_{CPA}^{current} denote the predicted minimum separation if maintaining the current heading, and let d_{CPA}^{next} be the predicted CPA after applying action a. If

   $$d_{CPA}^{\text{next}} < d_{CPA}^{\text{current}} - \eta,$$

   with η a small marging factor, then the action is rejected. Thus, actions that worsen predicted separation are blocked.

3. **Conflict introduction constraint:** If no predicted conflict is present in the current state, an action may not introduce a predicted conflict.

4. **Exit-heading constraint (when safe):** If no predicted conflict exists, an action that increases the exit-heading error beyond a small tolerance is rejected. Note: this shielding factor is not a safety-related constraint, but a performance-related constraint, yielding more desired behavior by preventing the aircraft from flying in the opposite direction of the exit waypoint.

The shield shown in Figure 15 is a *pre-shielding* mechanism that guarantees the agent only explores safe or geometrically meaningful actions, see Könighofer et al. (2020). Q-learning itself never sees unsafe actions, which accelerates training and stabilizes convergence.

Pre-action vs. post-action shielding. Safety constraints in shielded RL can be enforced either before action selection (pre-action shielding) or after an action has been proposed (post-action shielding). Pre-action shielding like in Figure 15 restricts the agent to selecting actions only from a state-dependent safe set, ensuring safety by construction, see Könighofer et al. (2020). This leads to more stable learning dynamics and stronger compatibility with formal verification and constrained control frameworks. Notably, pre-shielding has a faster convergence than post-shielding, as it maintains a consistency between the agent's decisions and the executed behavior, although overly conservative constraints may reduce exploration and slow convergence (Könighofer et al., 2020, 2025).
In contrast, post-action shielding allows the agent to propose arbitrary actions that are subsequently filtered or corrected by a safety mechanism (Könighofer et al., 2020). While this approach is easier to integrate with existing agents and can support broader exploration, it introduces a mismatch between the agent's internal policy updates (i.e., what the agent assumes it has executed) and the actions actually executed in the environment—commonly described as *action aliasing*—which can degrade learning stability and complicate analysis (Könighofer et al., 2020). Pre-action shielding is generally preferred when feasible, whereas post-action shielding remains a practical alternative when architectural constraints limit tighter integration.

Shielding vs. reward shaping. In safety-critical reinforcement learning, there is a fundamental design choice between encoding constraints and desired behavior in (i) an explicit *action shielding* mechanism, or (ii) the *reward function*. In the aircraft conflict-resolution setting, both approaches can accelerate learning, but they have different implications for convergence speed, policy flexibility, and human-feedback integration.
Reward shaping provides a principled way to encode behavioral preferences (e.g. minimizing extra track miles, limiting the number of maneuvers, returning to a nominal heading). However, reward-only approaches can converge slowly in this domain for three reasons:

1. **Delayed credit assignment:** separation-loss penalties and track-mile penalties may accumulate over multiple steps, requiring TD learning to propagate information backward in time.

2. **Small reward differences between actions:** many heading-change actions lead to similar short-term outcomes, creating a low signal-to-noise ratio for Q-learning.

3. **Exploration burden:** without constraints, the agent must explore many suboptimal but feasible behaviors before learning to avoid them, especially with function approximation.

These effects often produce long transients before the policy becomes useful.
Pre-selection action shielding reduces the effective action space from $\mathcal{A}$ to a state-dependent subset $\mathcal{A}_{\text{safe}}$, dramatically increasing sample efficiency. In particular, shielding i) blocks actions that violate hard safety constraints (immediate separation loss), ii) removes actions that degrade predicted safety margins (e.g. lower predicted d_{CPA}), iii) prevents exploration of actions that introduce conflicts from otherwise safe states. By filtering clearly undesirable actions, the agent focuses learning on the remaining meaningful decisions and typically achieves safe behavior with fewer episodes.

Initial findings. Enabling action shielding—by pruning actions that violate kinematic or separation constraints—leads to almost immediate emergence of desired conflict resolution behavior. Since unsafe or clearly suboptimal actions are never explored, the agent effectively operates within a reduced and well-structured action space. While this produces near-operationally acceptable trajectories from the start, it also strongly limits exploration, preventing the agent from discovering alternative avoidance strategies or learning fine-grained trade-offs between maneuver magnitude, timing, and recovery. As a result, shielding is highly effective for deployment and demonstration, but less suitable as a standalone training mechanism.

6.2.3.2 Human Feedback as Policy Shaping After (or during) autonomous Q-learning, human-feedback mechanisms can also shape the policy without discarding the learned Q-function. Here, human feedback can be given after each episode and not after each individual action, see Figure 17.

Pairwise preference learning. Given two trajectories A and B, the human selects the preferred one. If A is preferred over B, the Q-function is updated as:

$$w \leftarrow w + \alpha_h \big(\nabla_w Q(A) - \nabla_w Q(B)\big),$$

while the update is reversed if B is preferred. The update will push up the value of preferred actions and decreasing the value of dispreferred ones (Jain et al., 2013, 2015; Wirth et al., 2017). Here, α_h is the feedback gain. To ensure a fair comparison, both trajectories are initialized with identical encounter geometry, including ownship and intruder positions, headings, and speeds.
To avoid trivial duplication of behavior, the second trajectory B must be explicitly forced to diverge from the earliest non-zero maneuver decision by excluding only the first action taken in trajectory A

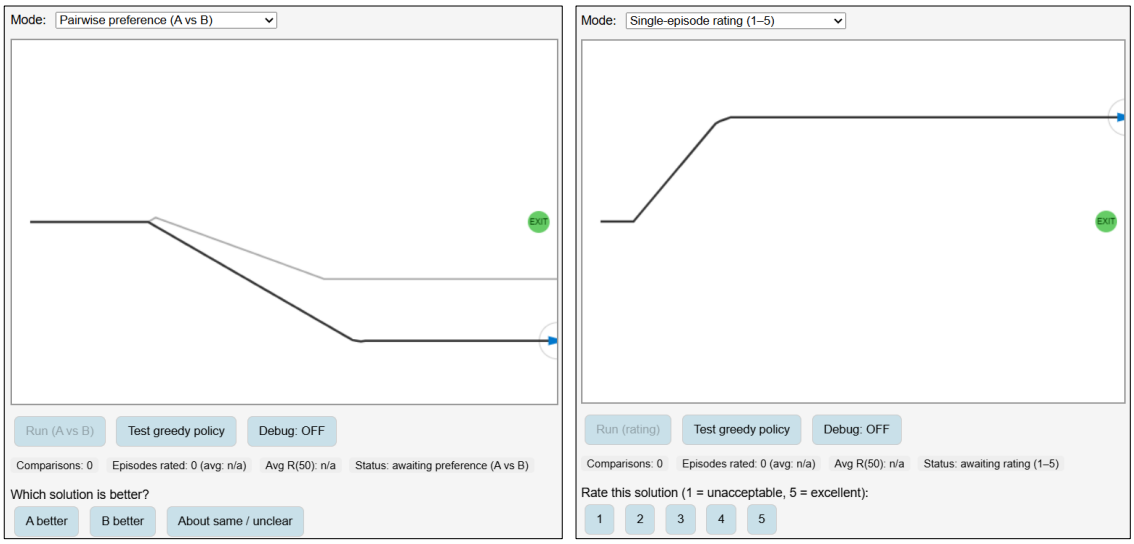


FIGURE 17 - HUMAN FEEDBACK MODES: PAIRWISE COMPARISON BETWEEN TWO TRAJECTORIES (LEFT) VS. SINGLE TRAJECTORY RATING (RIGHT).

at that decision step. Apart from this enforced divergence, both trajectories are generated using the same policy, optional action shielding, and constraints.

After execution, the user selects the more acceptable trajectory (or indicates no clear preference), and the agent's action-value estimates are updated by reinforcing actions along the preferred trajectory while penalizing those of the non-preferred one. This mechanism allows human judgment to directly shape the policy while preserving safety and consistency across comparisons.

With linear function approximation $Q(s,a) = \mathbf{w}^\top \boldsymbol{\phi}(s,a)$, the Q-function update simplifies to adding a scaled difference of feature vectors, $\alpha_h(\boldsymbol{\phi}(s,A) - \boldsymbol{\phi}(s,B))$, to the weights. Geometrically, this moves the parameter vector in a direction that increases the margin $Q(s,A) - Q(s,B)$, thereby reshaping the value function so that the preferred action becomes more attractive in the current state and in similar states that share feature structure. Repeated updates of this kind gradually enforce consistency with human preferences by increasing the separation between favored and disfavored actions in value space.

This update can be interpreted as stochastic gradient ascent on an implicit objective that maximizes the difference between the Q-values of preferred and rejected actions, or equivalently minimizes a ranking loss that penalizes violations of the desired ordering. Unlike standard temporal-difference learning, which adjusts values to match observed rewards and bootstrapped returns, pairwise feedback imposes inequality constraints of the form $Q(s,A) > Q(s,B)$, making it a form of preference-based reinforcement learning. The resulting mechanism resembles large-margin ranking updates, in which each comparison incrementally pushes the value function toward a configuration that respects accumulated human judgments. When integrated with ordinary Q-learning updates, these preference gradients act as an *auxiliary* shaping signal that biases the learned policy toward actions that align

with human expectations, while still allowing environmental rewards to determine long-term optimal behavior.

Rating-based feedback. A trajectory receives a human rating $r \in \{1, 2, 3, 4, 5\}$, mapped to a scalar signal $R \in [-1, 1]$ (Jain et al., 2013, 2015; Wirth et al., 2017). The weights are updated:

$$w_a \leftarrow w_a + \alpha_h (R - Q(s, a; w))\, \phi(s).$$

This update adjusts the value estimate toward the human-provided signal: if the predicted value $Q(s, a; w)$ is lower than the rating-derived signal R, the estimate is increased, whereas if it is higher, the estimate is reduced. In this way, human ratings act as a target signal that gradually shifts the value function toward trajectories that humans judge as desirable. Highly rated trajectories increase the value of their action sequences; low-rated ones decrease it.

Initial findings. Incorporating human feedback, for example by penalizing or discouraging undesirable actions during training, proves ineffective when applied from scratch. Without a baseline policy that already achieves conflict avoidance, the feedback signal is overwhelmed by random exploration and sparse terminal rewards. Empirically, human feedback becomes useful only once the agent has learned a *minimum viable policy*, after which feedback can bias learning toward smoother trajectories, fewer maneuvers, or earlier recovery. This observation aligns with the interpretation of human feedback as a *policy shaping* mechanism rather than a replacement for reinforcement *learning*.

6.2.3.3 Learning from Expert Demonstrations In addition to learning-based conflict resolution, pure heuristic algorithms that reflect established air traffic control (ATC) best practices can provide structured guidance for decision-making. Such algorithms can also be used as supervisory signals for RL, effectively serving as algorithmic priors or heuristic teachers when historical expert demonstration data is limited or unavailable. Here, a rule-based Velocity Obstacle (VO) formulation is used to generate control actions that place the relative velocity of the controlled aircraft outside the collision cone (i.e., the velocity obstacle) induced by the intruder's protected zone, thereby providing guidance signals for the learning agent (Fiorini, 1998; Mercado Velasco et al., 2015), see Figure 18. In the proposed rule-based variant, candidate maneuvers are further constrained to ensure a fixed *pass-behind* geometry: among all safe actions that position the relative velocity vector outside the velocity obstacle, the algorithm selects the smallest available heading and/or speed deviation for which the intruder lies ahead of the controlled aircraft at the predicted closest point of approach. This criterion directly encodes the operational preference for predictable behind-passing maneuvers, which are commonly issued by *expert* human controllers (Regtuit et al., 2018).

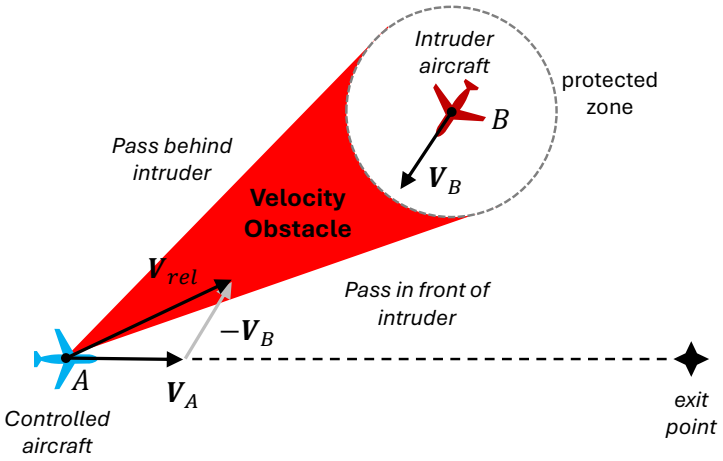


FIGURE 18 - VO GEOMETRY AS SEEN FROM THE CONTROLLED AIRCRAFT. HERE, ANY ACTION ON V_A THAT PUTS V_{rel} TO THE LEFT OF THE VO RESULTS IN PASSING THE INTRUDER FROM BEHIND.

Once the intruder has passed the controlled aircraft along the original route direction and a safety margin is guaranteed, the rule-based logic transitions to a recovery phase. In this phase, the controlled aircraft applies the minimal corrective action that reduces deviation from the original heading (and speed, if enabled), while ensuring that no new predicted conflict is introduced. By defining "passed" relative to the initial route direction rather than the instantaneous heading, the algorithm robustly initiates recovery even after large avoidance maneuvers.

Incorporating expert demonstrations into Q-learning. To accelerate learning and bias the policy toward operationally acceptable behavior, expert demonstrations are incorporated into the Q-learning updates. Let $\pi_E(s)$ denote an expert policy, here realized by a rule-based VO conflict resolution controller that selects safe and structured maneuvers (i.e., pass-behind strategies) (Fiorini, 1998; Mercado Velasco et al., 2015). Given a state s_t and the expert-selected action $a_t^E = \pi_E(s_t)$, the agent receives an additional learning signal beyond the standard temporal-difference (TD) update.

When an expert action a_t^E is available, an additional supervised-style update is applied to encourage the agent to assign higher value to the demonstrated action (Piot et al., 2014). Specifically, the agent performs an imitation update of the form

$$\mathbf{w}_{a_t^E} \leftarrow \mathbf{w}_{a_t^E} + \alpha_E \left(Q_E - Q(s_t, a_t^E)\right) \boldsymbol{\phi}(s_t), \tag{17}$$

where α_E is a demonstration gain and Q_E is a target value that biases the demonstrated action upward relative to competing actions. In practice, Q_E can be chosen implicitly (e.g., by applying a fixed positive margin) or implemented as a small additive boost to the demonstrated action's Q-value.

In the margin-based formulation of learning from expert demonstrations, the demonstrated action is encouraged not merely to have a high value in absolute terms, but to be decisively better than *all*

other competing actions (i.e., $a \neq a^E$) in the same state. Concretely, a target value is constructed as $Q_E = \max_a Q(s,a) + m$, where $m > 0$ is a fixed margin that specifies how much better the expert action should be compared to the agent's current best alternative, see Piot et al. (2014). The Q-update can then be interpreted as stochastic gradient descent on a regression objective that drives the predicted value of the demonstrated action toward this margin-based target. If the demonstrated action already exceeds all other actions by at least the margin, the update vanishes or becomes negative, preventing unnecessary growth; otherwise, the update increases its value until the inequality $Q(s,a^E) \geq \max_a Q(s,a) + m$ is satisfied. This creates a clear separation between the expert action and its competitors, improving policy robustness by ensuring that small value estimation errors or noise do not easily cause the policy to deviate from demonstrated behavior.

This update does not replace the TD update, but rather complements it, resulting in a combined learning signal:

$$\Delta \mathbf{w} = \Delta \mathbf{w}_{\text{TD}} + \Delta \mathbf{w}_{\text{demo}}. \tag{18}$$

To avoid over-constraining the policy, expert imitation is applied selectively. In particular, demonstration updates are gated to states in which expert knowledge is most relevant, such as when a conflict is predicted or imminent. Formally, the imitation update is applied only if

$$\mathbb{I}_{\text{demo}}(s_t) = \begin{cases} 1, & \text{if predictedConflict}(s_t) = \text{true}, \\ 0, & \text{otherwise}, \end{cases} \tag{19}$$

This ensures that the agent remains free to explore during benign flight phases while receiving strong guidance during safety-critical situations.

This approach is closely related to demonstration-augmented reinforcement learning methods such as Deep Q-learning from Demonstrations (DQfD), see Hester et al. (2018), but differs in two key aspects. First, demonstrations are incorporated online rather than through a fixed replay buffer, allowing the agent to continuously receive an additional learning signal in conflict states. Second, no hard constraint is imposed to force imitation; instead, demonstrations softly bias the value function while preserving ϵ-greedy exploration. As a result, the agent can initially imitate expert behavior to achieve safe and efficient conflict resolution, and subsequently refine or simplify these strategies as dictated by the learned reward structure.

By injecting expert demonstrations into the Q-updates, the agent receives informative gradients even in regions of the state space where reward signals are sparse or delayed. This significantly reduces the number of episodes required to discover effective avoidance maneuvers, stabilizes early learning, and promotes convergence toward policies that align with established operational practices while retaining the flexibility of reinforcement learning.

Initial findings. Expert demonstrations generated by a rule-based Velocity Obstacle (VO) pass-behind controller significantly reduce training time. Compared to baseline Q-learning, the agent typically reaches stable conflict resolution behavior within **500–1000 episodes**. By biasing exploratory actions toward expert-selected maneuvers, the agent quickly learns to associate key geometric features (e.g., relative bearing and d_{CPA} margin) with effective avoidance actions. However, because the expert follows a fixed steer-behind heuristic, the learned policy may inherit a strong inductive bias, defaulting to this behavior even in situations where alternative maneuvers would be equally valid or more efficient. Consequently, while expert demonstrations accelerate convergence, they may also limit policy diversity (impacted by the margin size m) and adaptability if not combined with continued exploration or regularization.

6.2.3.4 Stability considerations with multiple learning signals When off-policy Q-learning with linear function approximation is combined with human pairwise feedback and expert demonstrations, multiple weight updates influence the value function simultaneously. Temporal-difference (TD) learning adjusts Q-values according to environmental rewards, while demonstration and human feedback updates reshape them according to external guidance. If applied with large learning rates or inconsistent signals, these updates may compete and lead to oscillations or overestimation, particularly because off-policy Q-learning with function approximation is already sensitive to instability.

In the present setting, pre-shielding is used to prevent unsafe actions before they are executed. This means the agent only ever observes transitions resulting from admissible actions, and unsafe actions are removed from the decision set in advance. For consistency, TD targets should therefore be computed only over actions that remain available after pre-shielding, ensuring that value estimates reflect the true set of executable actions. A stable integration strategy when safety is critical is to treat TD learning as the primary learning signal with pre-shielding *always* activated, use expert demonstrations mainly during *early* learning to bootstrap reasonable value estimates, and *occasionally* apply human pairwise (or rating) feedback as a weaker auxiliary signal for refining action preferences. Gradually reducing the influence of demonstration and human feedback updates over time may help prevent conflicting gradients and supports stable convergence of the optimal value function.

6.2.3.5 Future directions Although the presented framework demonstrates a practical combination of Q-learning, action shielding, human-feedback shaping, and “learning from expert demonstrations” to shape RL behavior towards operationally accepted policies in line with human air traffic controllers’ expectations, several methodological and conceptual limitations remain that will need to be addressed in a next iteration.

Hierarchical or multi-phase reasoning. Currently, the agent implicitly learns an "avoid then return" pattern via a gated reward function and shaping, but no structural bias exists to enforce such a two-phase policy. This may result in occasional unnecessary maneuvers or delayed return-to-exit actions. A hierarchical RL architecture or an explicit high-level phase-switch policy could better reflect the decomposed structure of human conflict-resolution strategies.

Scalability limitations for multi-aircraft scenarios. While the proposed approach demonstrates the feasibility of shielded off-policy Q-learning for pairwise aircraft conflict resolution, its current formulation exhibits several limitations when scaling to scenarios involving multiple intruding aircraft. The present feature representation encodes the relative geometry between the ownship and a single intruder. In multi-aircraft encounters, a naive extension would require concatenating feature vectors for each intruder, leading to a state dimensionality that grows linearly with the number of aircraft. This rapidly increases the complexity of the value-function approximation, degrades generalization, and exacerbates sample inefficiency, particularly for linear function approximators.

The linear Q-function approximation employed in the current implementation assumes that the effect of features on action values is approximately additive (Melo and Ribeiro, 2007). In multi-aircraft encounters, however, optimal decisions often depend on nonlinear interactions between intruders (e.g., prioritizing the most critical conflict while temporarily accepting reduced margins elsewhere). Capturing such interactions typically requires more expressive function approximators or explicit prioritization mechanisms.

The formulation treats conflict resolution as a single-agent problem in which only the ownship is controlled and all intruders follow fixed trajectories. In multi-aircraft scenarios, especially those involving other adaptive or autonomous agents, the environment becomes non-stationary from the perspective of any single learner. Standard off-policy Q-learning does not directly address this non-stationarity and may exhibit instability or slow convergence.

Scaling the approach to multi-aircraft scenarios will likely require a combination of architectural and algorithmic changes, such as attention-based or set-invariant state representations, hierarchical or receding-horizon decision structures, prioritized conflict selection, and more expressive value or policy models. Additionally, extending the shielding mechanism to reason over multiple intruders jointly, rather than independently, will be essential to preserve safety guarantees without overly restricting the action space.

Human feedback for more complex agents. In contrast to linear function approximation, where human feedback can directly modify individual Q-weights, incorporating human preferences into more complicated model structure, like Deep Q-Networks (DQN), requires indirect mechanisms that shape the learned action-value function through additional training signals. Humans typically do not give

feedback on gradients, network weights, and TD targets. Several indirect approaches are commonly considered, each offering different trade-offs between simplicity, scalability, and stability.

A straightforward option is *direct reward shaping*, where human feedback is converted into an auxiliary reward signal and added to the environment reward. If a human provides scalar feedback r_t^{h} (e.g. an episodic rating mapped to a per-step bonus), the environment reward can be augmented as

$$\tilde{r}_t = r_t + \lambda r_t^{\mathrm{h}},$$

While simple to implement, this approach suffers from poor credit assignment when feedback is sparse or delayed, and may destabilize learning if human signals are noisy or inconsistent. As a result, direct reward shaping is typically best suited for prototyping rather than long-term deployment.

A more principled approach is to train a separate *reward model* from human feedback, following the reinforcement learning from human feedback (RLHF) paradigm. In this setting, human preferences are used to learn a parametric reward function, which then replaces or augments the hand-crafted reward during DQN training. This decouples preference learning from policy optimization, improves scalability, and provides a clean interface for integrating feedback from multiple users. However, it introduces additional model complexity and training overhead.

Beyond reward shaping and preference-based learning, human influence can be incorporated into DQN training by steering the agent's behavior without modifying the underlying temporal-difference objective. For example, a technique that includes *residual* reinforcement learning, in which the agent learns small corrections on top of a baseline heuristic controller, and demonstration-biased replay, where human or scripted trajectories are oversampled during training without introducing imitation or preference losses.

Limitations of expert demonstrations. The current expert demonstrations are based on a deterministic Velocity Obstacle (VO) *pass-behind* rule, which is operationally sound and effective at preventing conflicts but introduces several limitations when used for reinforcement learning. Because the rule represents a single fixed avoidance strategy, it induces a strong bias toward repeated pass-behind maneuvers, reducing policy diversity and limiting generalization to alternative safe solutions. Moreover, the expert policy is myopic with respect to longer-term objectives such as minimizing cross-track deviation, maneuver count, and smooth recovery to the original flight path, leaving these aspects to be learned solely from the reward signal. Finally, demonstrations are provided at the single-action level and do not encode temporal structure, such as the desired two-phase behavior of one avoidance maneuver followed by one recovery maneuver, which can lead to oscillatory actions and maneuver budget depletion.

These limitations could be mitigated by incorporating multi-modal expert strategies, context-

dependent weighting of demonstrations across flight phases, and short expert trajectories instead of isolated actions. While the VO rule provides an effective bootstrap, richer and more diverse expert guidance is necessary to achieve flexible and operationally robust learned policies.

6.2.4. REAL-TIME INTERACTIVE PREFERENCE LEARNING

Interactive learning refers to learning processes that occur *during system use* rather than in a separate offline training stage. It allows AI models to incorporate contextual or situational knowledge not available during initial model development.

A key motivation for adopting interactive learning in real time is the fundamental scarcity of labeled human data. High-quality annotations from domain experts—whether they represent operational preferences, safety constraints, or nuanced judgments—are inherently limited. Experts are available only intermittently, and the cognitive cost of producing comprehensive labeled datasets is prohibitive. As a result, classical offline supervised learning pipelines, which rely on large, static datasets, are often impractical or incapable of capturing the full richness of human reasoning in operational environments such as Air Traffic Management (ATM).

Within AI4REALNET, this challenge becomes very concrete in the context of dynamic airspace sectorization. The sectorization problem is a high-dimensional, multi-objective optimization task that must balance competing goals (workload distribution, geometry clarity, coordination costs, traffic flow predictability, etc.) (Delahaye et al., 1998; Schultz et al., 2019). No large labeled dataset exists for "good" vs. "bad" sectorizations—not only because expert judgment is too expensive to collect at scale, but also because optimal sector shapes depend heavily on context, weather, traffic patterns, operator preferences, and local operational culture. In such a setting, fully supervised learning is infeasible.

To overcome this, the AI4REALNET framework employs a stochastic multi-objective Interactive Evolutionary Computation (IEC) approach using Covariance Matrix Adaptation-Evolutionary Strategies (CMA-ES) (Hansen, 2006), coined here as iCMA-ES, to generate a diverse set of candidate sectorizations on the fly. Instead of needing pre-labeled examples, the evolutionary algorithm explores the solution space and presents a rich collection of alternatives every run. These solutions are displayed to the human operator through an interactive Pareto frontier, where the user can visually inspect trade-offs across objectives and indicate preferred or rejected solutions with minimal cognitive effort. Each feedback action—selecting a promising configuration, rejecting a fragmented sector, or choosing a geometry that better aligns with expected traffic flows—provides a high-value learning signal.

These sparse human signals feed into an adaptive kernel density reward model, which acts as a surrogate preference learner. The model incorporates each preference sample into a smooth reward landscape that guides future CMA-ES generations. Crucially, the model dynamically adjusts its kernel size based on the consistency of human feedback:

- When the operator's preferences are stable and coherent, the kernel narrows, creating sharper gradients that drive CMA-ES rapidly toward high-quality regions.
- When feedback is noisy, exploratory, or contradictory, the kernel broadens to avoid overfitting and maintain robust, diverse search behavior.

This adaptive mechanism ensures that the AI learns at the right level of generality: neither too rigidly (which risks lock-in to a misleading early preference), nor too diffusely (which would dilute the impact of human judgment).

Through this interactive, co-adaptive loop, the system effectively constructs its training data *during* operation—leveraging human input only when it is most informative and using evolutionary strategies to fill the gaps where no labeled data exists. In this way, real-time interactive learning provides a practical alternative to traditional supervised learning, enabling advanced AI capabilities in complex ATM applications where human judgment is essential and large annotated datasets simply cannot be obtained. This concept has been operationalized in an extension to the ATM Sectorization application and HMI developed under WP1 (T1.3) and WP2 (T2.3), see Figure 19.

6.2.4.1 Background Dynamic airspace sectorization can be fundamentally framed as an optimization problem driven by the strategic placement of Voronoi centers that define sector boundaries in response to evolving traffic patterns (Xue, 2009). By positioning these centers appropriately, the re-

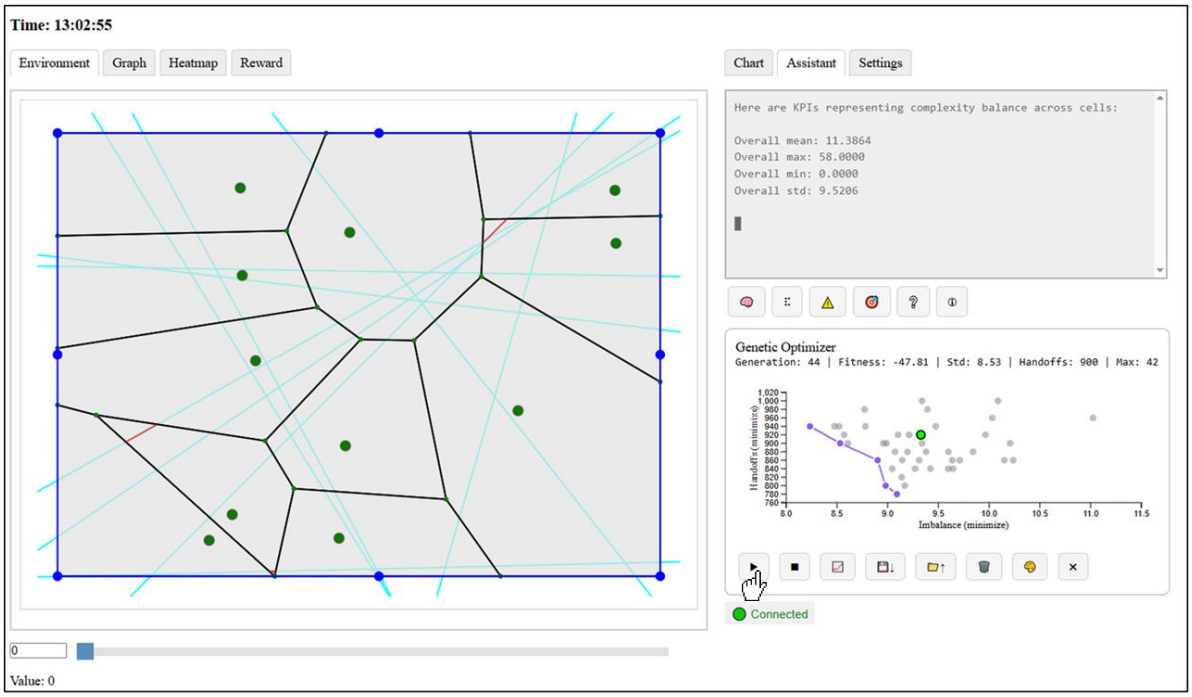


FIGURE 19 - ATM DYNAMIC SECTORIZATION ASSISTANT.

sulting Voronoi tessellation generates convex sectors that are geometrically simple, operationally intuitive, and well suited for controller management. Within this framework, two competing objectives must be jointly optimized. First, sectors should be configured so that the predicted air traffic control officer (ATCO) taskload is evenly distributed, typically expressed by minimizing the standard deviation of workload across sectors to ensure balanced cognitive demand and sustainable operational performance. Second, the number of inter-sector hand-offs must be minimized, as each transfer of aircraft between sectors introduces coordination workload and potential complexity; reducing hand-offs promotes more continuous, unfragmented airspace structures. Effective dynamic sectorization therefore requires carefully balancing workload equity and coordination efficiency, using Voronoi center placement as a flexible mechanism to adapt sector geometry while maintaining operationally acceptable and human-centered airspace designs.

6.2.4.2 Multi-objective CMA-ES Dynamic airspace sectorization can be formulated as a continuous optimization problem in which the placement of Voronoi centers determines sector boundaries and directly influences controller workload distribution and coordination complexity across the airspace (Xue, 2009). Let K denote the number of sectors and d the spatial dimension (typically $d = 2$ for horizontal airspace). The optimization variable is the stacked vector of Voronoi center coordinates

$$x \in \mathbb{R}^n, \qquad n = Kd, \qquad x = [c_1^\top, c_2^\top, \ldots, c_K^\top]^\top, \tag{20}$$

where each $c_k \in \mathbb{R}^d$ represents the spatial position of Voronoi center k, and n is the total number of decision variables describing all sector centers. The Voronoi tessellation induced by these centers defines sector geometries such that each point in space is assigned to its nearest center:

$$V_k = \left\{ p \in \mathbb{R}^d : \|p - c_k\| \leq \|p - c_j\|, \ \forall j \neq k \right\}, \tag{21}$$

where p denotes any point in the airspace, $\|\cdot\|$ denotes Euclidean distance, and V_k is the Voronoi region associated with center c_k. Each operational sector is defined as $S_k = V_k \cap \Omega$, where Ω represents the bounded airspace region under consideration. This construction ensures convex, geometrically simple sectors that remain interpretable for human operators and suitable for dynamic adaptation.

Two competing operational objectives must be optimized. The first objective balances controller workload across sectors by minimizing the standard deviation of predicted ATCO taskload:

$$f_1(x) = \sigma_L(x) = \sqrt{\frac{1}{K} \sum_{k=1}^{K} \left(\widehat{L}_k(x) - \overline{L}(x) \right)^2}, \qquad \overline{L}(x) = \frac{1}{K} \sum_{k=1}^{K} \widehat{L}_k(x), \tag{22}$$

where $\widehat{L}_k(x)$ denotes the predicted taskload for sector k given the Voronoi configuration defined by x, $\overline{L}(x)$ is the mean taskload across all sectors, and $\sigma_L(x)$ represents the dispersion of workload.

Minimizing $f_1(x)$ promotes equitable distribution of cognitive demand across controllers. The second objective reduces coordination complexity by minimizing inter-sector hand-offs:

$$f_2(x) = H(x), \tag{23}$$

where $H(x)$ represents the number of aircraft trajectory crossings between sectors, and thus quantifies coordination workload between controllers. Lower values of $f_2(x)$ correspond to more continuous and less fragmented airspace structures.

Because the two objectives are conflicting, scalarization is used to convert the multi-objective problem into a sequence of single-objective problems compatible with CMA-ES (Paria et al., 2018). At each *optimization run*, a random scalarization weight is sampled:

$$u \sim \mathcal{U}(0,1), \qquad w^{(t)} = \begin{bmatrix} u \\ \\ 1-u \end{bmatrix}, \tag{24}$$

where u is a uniform random variable and $w^{(t)}$ is a weight vector assigning relative importance to workload balance and coordination minimization. The scalarized cost function becomes

$$J^{(t)}(x) = w_1^{(t)} \frac{f_1(x)}{s_1} + w_2^{(t)} \frac{f_2(x)}{s_2}, \tag{25}$$

where $w_1^{(t)}$ and $w_2^{(t)}$ are the scalarization weights, and s_1 and s_2 are optional normalization constants ensuring comparable magnitudes between objectives. Varying $w^{(t)}$ across iterations allows exploration of different trade-offs between balanced taskload and reduced coordination workload.

The CMA-ES is used to automatically compute optimal Voronoi center placements. CMA-ES is a stochastic search method well suited for high-dimensional continuous problems because it adapts the entire covariance structure of the sampling distribution (Hansen, 2006). At iteration t within a single optimization run, CMA-ES maintains a multivariate normal sampling distribution characterized by mean vector $m_t \in \mathbb{R}^n$ (representing the current best estimate of center placement), step-size $\sigma_t > 0$ (controlling global search scale), and covariance matrix $C_t \in \mathbb{R}^{n \times n}$ (describing the shape and orientation of the search distribution). Candidate solutions are generated as

$$z_{t,i} \sim \mathcal{N}(0, I_n), \qquad i = 1, \ldots, \lambda, \tag{26}$$

$$x_{t,i} = m_t + \sigma_t A_t z_{t,i}, \qquad A_t A_t^\top = C_t, \tag{27}$$

where $z_{t,i}$ is a standard normal random vector, λ is the number of sampled candidate solutions per iteration, and A_t is a matrix square root of C_t used to shape the sampling distribution. Each $x_{t,i}$ corresponds to a complete set of Voronoi center positions and therefore a full sectorization. Candidates are evaluated using $J^{(t)}(x_{t,i})$ and ranked from best to worst. The top μ candidates are recombined

using positive weights w_i satisfying $\sum_{i=1}^{\mu} w_i = 1$, producing normalized search steps

$$y_{t,i} = \frac{x_{t,i:\lambda} - m_t}{\sigma_t}, \qquad y_w = \sum_{i=1}^{\mu} w_i y_{t,i}, \tag{28}$$

where $y_{t,i}$ represents the normalized displacement of candidate i from the current mean and y_w is the weighted average step direction. The notation $x_{t,i:\lambda}$ denotes the i-th best candidate after ranking all λ samples. The mean update becomes

$$m_{t+1} = m_t + \sigma_t y_w, \tag{29}$$

shifting the search distribution toward better-performing Voronoi configurations. The effective selection mass

$$\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} w_i^2 \right)^{-1} \tag{30}$$

quantifies the variance-effective number of selected candidates.

To adapt search dynamics, evolution paths accumulate information across iterations. The step-size evolution path

$$p_{\sigma,t+1} = (1 - c_\sigma) p_{\sigma,t} + \sqrt{c_\sigma (2 - c_\sigma) \mu_{\text{eff}}}\, C_t^{-1/2} y_w \tag{31}$$

tracks successive movement directions, where c_σ is a learning rate and $C_t^{-1/2}$ whitens the step using the current covariance. The covariance evolution path

$$p_{c,t+1} = (1 - c_c) p_{c,t} + h_{\sigma,t+1} \sqrt{c_c (2 - c_c) \mu_{\text{eff}}}\, y_w \tag{32}$$

captures long-term directional trends, where c_c is a learning parameter and $h_{\sigma,t+1}$ is an indicator determining whether the step-size path length is within expected bounds. The covariance matrix update

$$C_{t+1} = (1 - c_1 - c_\mu) C_t + c_1 p_{c,t+1} p_{c,t+1}^\top + c_\mu \sum_{i=1}^{\mu} w_i y_{t,i} y_{t,i}^\top \tag{33}$$

combines rank-one adaptation from the evolution path and rank-μ adaptation from selected candidates, where c_1 and c_μ are covariance learning rates. Finally, the global step-size evolves as

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\|p_{\sigma,t+1}\|}{\chi_n} - 1 \right) \right), \qquad \chi_n \approx \sqrt{n} \left(1 - \frac{1}{4n} + \frac{1}{21 n^2} \right), \tag{34}$$

where d_σ is a damping parameter and χ_n is the expected norm of a standard normal vector in n dimensions. Through repeated sampling, evaluation, and adaptation of m_t, C_t, and σ_t, CMA-ES automatically discovers Voronoi center placements that produce convex sectors with balanced predicted controller taskload while minimizing coordination workload from inter-sector hand-offs, and the use of random scalarization ensures exploration of operationally meaningful trade-offs between these competing objectives over several optimization runs.

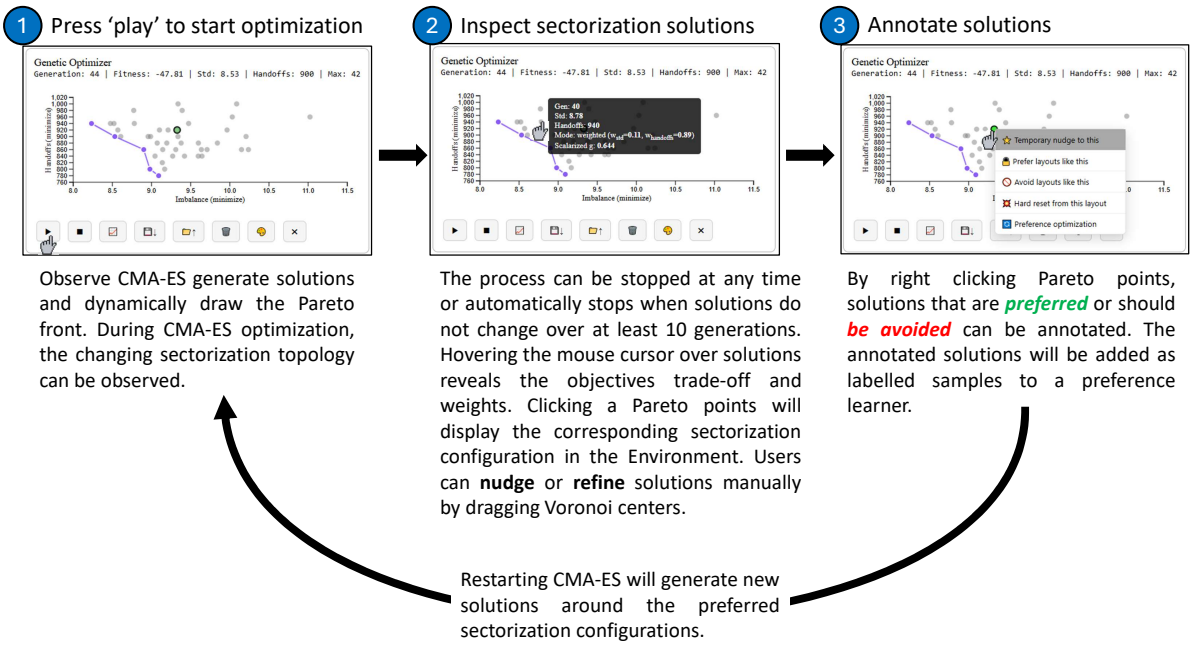


FIGURE 20 - HUMAN-IN-THE-LOOP INTERACTION AND SOLUTION STEERING IN ATM SECTORIZATION.

6.2.4.3 Human-in-the-loop interaction and solution steering To ensure operational acceptability and human-centered control, the CMA-ES optimization process can be interactively steered by a human operator during runtime, see Figure 20. At any iteration, the optimization can be temporarily frozen to allow inspection of the current population of candidate sectorizations and the corresponding trade-offs between workload balance and coordination complexity. Each evaluated candidate solution corresponds to a specific Voronoi center configuration and therefore to a concrete airspace sectorization that can be visualized and assessed by the user. The operator may then select a preferred intermediate optimization point representing a compromise between balanced ATCO taskload and minimized inter-sector hand-offs.

Once selected, the optimization can be restarted from this configuration by replacing the CMA-ES mean vector m_t with the chosen Voronoi center vector, thereby centering subsequent search around the user-approved sectorization. Alternatively, user-provided sector centers can be injected directly to override or bias the search distribution, either by fully redefining m_t or by blending the current mean with human-proposed centers using a convex combination. This interactive mechanism allows the algorithm to retain its adaptive exploration capabilities while incorporating expert judgment, enabling iterative refinement of sectorization solutions that remain both computationally efficient and operationally aligned with human expectations.

6.2.4.4 Human Preference Meta-Learner Beyond direct human interaction mechanisms such as freezing the optimization, revising sectorizations, selecting (intermediate) optimizer solutions, or

restarting the search from user-defined configurations, the human operator can also provide explicit qualitative feedback by labeling generated sectorizations as *preferred* or *to be avoided*. These binary annotations constitute an additional optimization signal that complements the intrinsic performance objectives (e.g. workload balance or hand-off minimization) and can be incorporated as a learned preference-based target shaping the search process. Rather than replacing the base objective, such feedback augments it, allowing the optimizer to progressively align with human operational intent while preserving its ability to explore novel configurations.

Let $\mathbf{x} \in \mathbb{R}^{2n}$ denote the vectorized coordinates of n Voronoi centers defining a candidate sectorization layout, where each pair of entries represents the spatial coordinates of one center and therefore the full vector uniquely specifies a complete airspace partition. At generation t, the optimizer evaluates an intrinsic task-specific cost $J(\mathbf{x})$ representing operational metrics such as workload imbalance, coordination complexity, or predicted controller taskload dispersion. In parallel, the human feedback module computes a real-time preference reward $R(\mathbf{x}, t)$ derived from labeled sectorizations. The combined scalarized objective guiding optimization is

$$F(\mathbf{x}, t) = J(\mathbf{x}) - \alpha(t)\, R(\mathbf{x}, t), \tag{35}$$

where $F(\mathbf{x}, t)$ is the augmented objective minimized by CMA-ES, $R(\mathbf{x}, t)$ is the learned preference reward representing the degree of human approval, and $\alpha(t) \geq 0$ is a time-dependent weighting factor controlling how strongly human guidance influences the search. In CMA-ES this objective is minimized directly, while in genetic algorithms the same expression may be interpreted as a maximized fitness. The preference reward combines continuous kernel-based influence from labeled samples and optional relational preference comparisons.

Each labeled configuration $\mathbf{x}_i$ is associated with a human label $y_i \in \{+1, -1\}$, where $+1$ indicates a preferred configuration and -1 indicates a configuration to be avoided. Every labeled sample induces a Gaussian kernel in the configuration space, producing a smooth reward field:

$$R_{\text{kernel}}(\mathbf{x}) = \frac{1}{Z} \sum_i y_i \, \exp\left[-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2} \right], \tag{36}$$

where $\|\mathbf{x} - \mathbf{x}_i\|$ denotes Euclidean distance between the current configuration and labeled sample i, σ is the kernel width controlling spatial influence of each sample, and Z is a normalization constant ensuring bounded reward magnitude. The exponential kernel produces positive reward near preferred samples and higher reward in areas with a greater concentration of positive samples. Conversely, it assigns negative reward near rejected samples. The resulting reward landscape is smooth and continuous, biasing exploration toward human-aligned regions while preserving continuity. Optionally, each contribution may be multiplied by a temporal decay factor $e^{-(t-t_i)/\tau}$, where t_i is the labeling time and τ controls how quickly older feedback becomes less influential.

The influence of human feedback can be gradually increased through a ramp-up schedule:

$$\alpha(t) = \alpha_{\text{min}} + (\alpha_{\text{max}} - \alpha_{\text{min}}) \left(\frac{t}{T_{\text{max}}} \right)^{\gamma}, \tag{37}$$

where α_{min} and α_{max} define minimum and maximum preference influence, T_{max} is the total number of generations, and $\gamma > 1$ produces a nonlinear ease-in schedule. Early iterations emphasize exploration of the intrinsic objective J, whereas later iterations increasingly exploit regions aligned with human preferences.

To further enhance adaptability, the meta-learner adjusts its spatial sensitivity through online meta-learning of kernel parameters. For a candidate configuration represented by Voronoi centers $\mathbf{C}$ and a labeled sample $(\mathbf{C}_i, y_i)$, the contribution of that sample is

$$r_i(\mathbf{C}) = y_i \, \exp\left(-\frac{D^2(\mathbf{C}, \mathbf{C}_i)}{2\sigma^2} \right), \tag{38}$$

where $D(\mathbf{C}, \mathbf{C}_i)$ denotes the average Euclidean distance between corresponding centers and σ is the kernel width controlling spatial generalization. The normalized kernel reward becomes

$$R_{\text{kernel}}(\mathbf{C}) = \frac{\sum_i r_i(\mathbf{C})}{\sum_i \exp\left(-\frac{D^2(\mathbf{C}, \mathbf{C}_i)}{2\sigma^2} \right)}, \tag{39}$$

ensuring bounded and scale-consistent reward magnitude. To adapt σ automatically, the system evaluates consistency of recent feedback. Let the most recent k labeled samples be $\{(\mathbf{C}_t, y_t)\}_{t=T-k}^{T}$ with labels $y_t \in \{+1, -1\}$. The consistency metric is

$$\kappa = \frac{1}{k-1} \sum_{i=T-k+1}^{T} y_i y_{i-1} \exp\left(-\frac{D^2(\mathbf{C}_i, \mathbf{C}_{i-1})}{2\sigma^2} \right), \tag{40}$$

where $\kappa > 0$ indicates consistent preferences in nearby regions and $\kappa < 0$ indicates contradictory feedback. The kernel width adapts according to

$$\sigma \leftarrow \begin{cases} \sigma(1 - \zeta), & \text{if } \kappa > 0, \\ \sigma(1 + 0.5\zeta), & \text{if } \kappa \leq 0, \end{cases} \tag{41}$$

where ζ is a small meta-learning rate. Consistent feedback narrows the spatial generalization to focus on local refinement (exploitation), whereas inconsistent feedback broadens spatial generalization (exploration). The overall preference influence weight may also adapt:

$$\omega \leftarrow \omega(1 + \beta \max(0, \kappa)), \tag{42}$$

where β controls how strongly consistent feedback increases global human influence.

To enable long-term adaptation, the meta-learner maintains a persistent memory across sessions. At

the end of each session, the internal state

$$\mathcal{M}_t = \left\{ \{(\mathbf{C}_i, y_i)\}_{i=1}^{N_t}, \sigma_t, \omega_t \right\} \tag{43}$$

is stored, where $\{(\mathbf{C}_i, y_i)\}$ is the set of labeled configurations, σ_t is the kernel width, and ω_t is the global preference weight. At the next session start, this memory is restored:

$$\mathcal{M}_{t+1}^{(0)} \leftarrow \mathcal{M}_t, \tag{44}$$

providing continuity of learned preferences. In practice, this persistence is implemented through serialization:

$$\texttt{saveToLocalStorage()} : \mathcal{M}_t \mapsto \texttt{JSON}, \tag{45}$$

$$\texttt{loadFromLocalStorage()} : \texttt{JSON} \mapsto \mathcal{M}_t. \tag{46}$$

Through the integration of explicit human labeling, real-time preference shaping, meta-learning of kernel sensitivity, and continual learning across sessions, the human preference meta-learner enables progressive alignment of the optimization process with human operational intent while preserving robust and adaptive exploration of the sectorization design space.

6.2.4.5 Concluding remarks and future directions In AI4REALNET's sectorization framework a stochastic multi-objective CMA-ES generates diverse sector configurations over different optimization runs, exposing the human to alternative solutions and, as such, helping humans refine their operational understanding.

Instead of approximating the Pareto front within a single optimization run, the proposed framework constructs the Pareto frontier progressively through a sequence of scalarized optimization runs. Each run optimizes a specific trade-off between the objectives, determined by a fixed scalarization weight vector. By randomly varying these weights across runs, the optimizer explores different regions of the objective trade-off space. As it requires multiple runs to approximate the Pareto front, exploration will become slower compared to other multi-objective evolutionary optimization strategies that can explicitly approximate Pareto fronts. However, such methods increase algorithmic complexity and generally reduce interpretability (for human operators) compared to scalarization.

A natural extension of this framework is to allow human operators to directly select or adjust the scalarization weights that define the objective trade-offs. Rather than predefining the weight vectors across optimization runs, the system could provide an input dialog through which users specify their preferred balance between objectives. This would enable operators to steer the optimization process toward regions of the trade-off space that better reflect current operational priorities. By making the objective weighting explicit and user-configurable, the framework would further strengthen human–AI

collaboration, allowing users to explore alternative sectorization strategies interactively while maintaining transparency regarding how optimization objectives influence the resulting solutions.

6.3. HUMANS LEARNING FROM AI

AI systems can reveal structure in data that may be difficult for humans to identify unaided. Hidden correlations, emerging patterns, regions of high uncertainty or operational sensitivity, feasible alternatives and their implications as well as multi-objective trade-offs can be made tangible to humans through visualization, summarization, explanation or interactive AI surfaces. This helps to contextualize decisions. By continuously presenting candidate solutions, trade-offs, and sensitivity information, users gain a deeper understanding of the decision space. This supports the development of mental models that are better aligned with the operational behavior of the system. Additionally, operators are given the option to actively formulate their own solutions and compare the trade-offs to AI-generated solutions, allowing for expansion of system understanding as well as that of the AI support system. The full co-learning concept is presented in section 6.3.6.

Both the co-learning AI and full autonomous AI concepts (described in Section 7) are based on a case study conducted to inform the design of human-AI collaboration that supports learning while preserving critical expertise for safe and resilient operations (T3.3), and to inform the design of fully autonomous AI (T3.4). The study was conducted in a traffic management control room of the participating railway company with three objectives:

1. understanding dispatchers' decision-making and learning processes as well as how dispatchers detect and interpret early indicators of disruption ("weak signals") (Sections 6.3.1-6.3.5)
2. developing a co-learning concept (T3.3; Section 6.3.6))
3. deriving requirements for the design of autonomous AI under human supervision (T3.4; Section 7.1.1))

The sample consisted of:

- Observations: n = 8 railway dispatcher, n = 1 assistant railway dispatcher, and n = 1 railway controller (for T3.3 and T3.4)
- Interviews: n = 5 railway dispatchers (for T3.3) and n = 5 (for T3.4)

All participants had more than 2 years of post training experience.

The following section provides a brief overview of the methodological approach of the case study (6.3.1), followed by a presentation of the results (6.3.2-6.3.5). Section 6.3.6 describes conclusions regarding the co-learning concept (T3.3), while requirements for the design of autonomous AI under human supervision (T3.4) are presented in section 7.1.1.

6.3.1. METHODOLOGICAL APPROACH OF THE CASE STUDY

Data collection comprised approximately 160 hours of non-participant full-shift observations and 20 hours of semi-structured interviews (10 hours T3.3; 10 hours T3.4). For T3.3, observation material was analyzed to identify recurring behavioral patterns and human-environment interactions, building a comprehensive representation of dispatchers' tasks and the contextual factors shaping performance. Interviews were used to deepen and validate these findings and focused in particular on weak signals, guided by the Structured Exploration of Complex Adaptations (SECA) method (Patriarca et al. (2022)). In order to describe the dispatching context systematically, the GOMS method was applied, in order to describe **G**oals, **O**perations, **M**ethods, and **S**election Rules (Card et al. (2008)). This approach facilitated task decomposition, the identification of decision points and relevant knowledge, and a clear characterization of roles and interactions between people, tasks, and organizational structures. In parallel, key cognitive demands were identified. To identify the knowledge and information sources necessary for successful task execution, WIL-FRAM was developed, a method based on the Functional Resonance Analysis Method (FRAM) (Hollnagel (2017)), which was adapted to represent the relevant knowledge elements and their relationships. This informed AI design requirements while reducing the risk of deskilling human experts with their critical expertise (see Section 4.4.5). The remaining elements were documented using a structured template, which included a description of the subtask, the roles of humans and technology in executing the task, and the external factors influencing execution. This formed the basis for GOMS. The cognitive demands were of particular interest as they highlighted difficulties and therefore informed AI design requirements directly. The results of the case study were used in workshops and other discussions together with our technical partners developing the co-learning AI for the railway use case, including the conceptual developments of D2.3 (e.g., Hamouche et al., in press).

6.3.2. GOMS RESULTS – A BRIEF INSIGHT

The case study produced a hierarchical GOMS model. It connects the main overall goal "running operations safely, efficiently, and on time (G0)" with smaller goals across three levels (G1-G3) and the respective operators that realize to achieve them. At the first level (G1), dispatchers' work can be grouped into five main aims:

1. Maintaining timetables and network stability,
2. ensure operational safety,
3. efficient use of infrastructure and resources,
4. establishing situational awareness and controllability, and

5. build and share own expertise and knowledge within and across teams.

To show what the deeper levels (G2/G3) look like, we use the G1 goal "building situation awareness and controllability" as an example. At this level, the goal refers to maintaining a clear and shared understanding of the current situation and retaining the ability to steer developments in a desired direction.

The G2 level translates this overarching goal into a small number of guiding principles that structure dispatchers' thinking and decision-making. These include continuously checking system forecasts against the evolving reality, actively closing information gaps across sectors, anticipating risks and downstream effects, and preserving decision sovereignty. Rather than reacting to events as they occur, dispatchers aim to influence the situation early so that later decisions remain manageable and undesirable options are avoided.

The G3 level then specifies how these principles are realized in everyday work. It consists of recurring operational activities such as analyzing train, conflict, and infrastructure data, anticipating possible future scenarios, coordinating with neighboring sectors and other actors, and preparing decisions in advance. At this level, the focus shifts from why control is needed to how it is practically achieved during the shift. An illustrative example highlights this logic across levels.

Illustrative example (keeping decision control across different sectors G2/G3). During a shift, the dispatcher deliberately kept an express train behind a freight train. Although this decision appeared counterintuitive at first, it was taken to prevent a likely downstream problem in the next sector. Had the express train been moved ahead, the neighboring sector might have delayed a punctual suburban train by several minutes to let the already heavily delayed express pass. By shaping the initial conditions early, the dispatcher made this downstream option infeasible, thereby preserving controllability and protecting punctual service across sectors.

6.3.3. DECISION-MAKING PATTERNS AND COGNITIVE DEMANDS

Observations show that dispatchers' decisions usually follow a repeating four-step control loop (Figure 21). First, they gather and interpret information from different sources. Second, they set priorities for the current situation (what matters most right now). Third, they develop and compare possible solutions. And fourth, they monitor what they implemented and make corrections if the situation changes. This loop repeats whenever new information comes in, constraints shift, or feedback from coordination with others makes a new check necessary. It can start proactively (when dispatchers act early to prevent problems) or reactively (when they respond to disruptions as they occur).

To summarize what makes planning difficult across these four phases, Figure 22 groups the main cognitive demands into several interacting drivers: infrastructure constraints, environmental factors, pri-

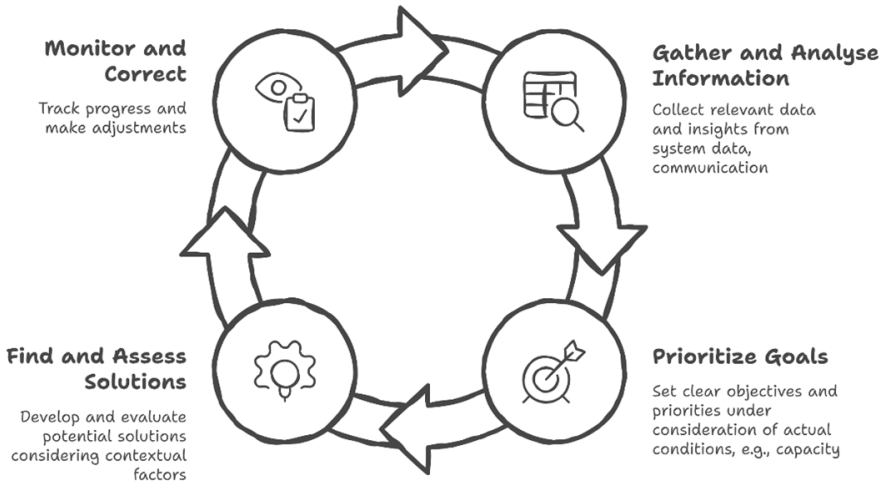


FIGURE 21 - DECISION MAKING CONTROL LOOP OF DISPATCHERS IN THE RAILWAY CONTEXT.

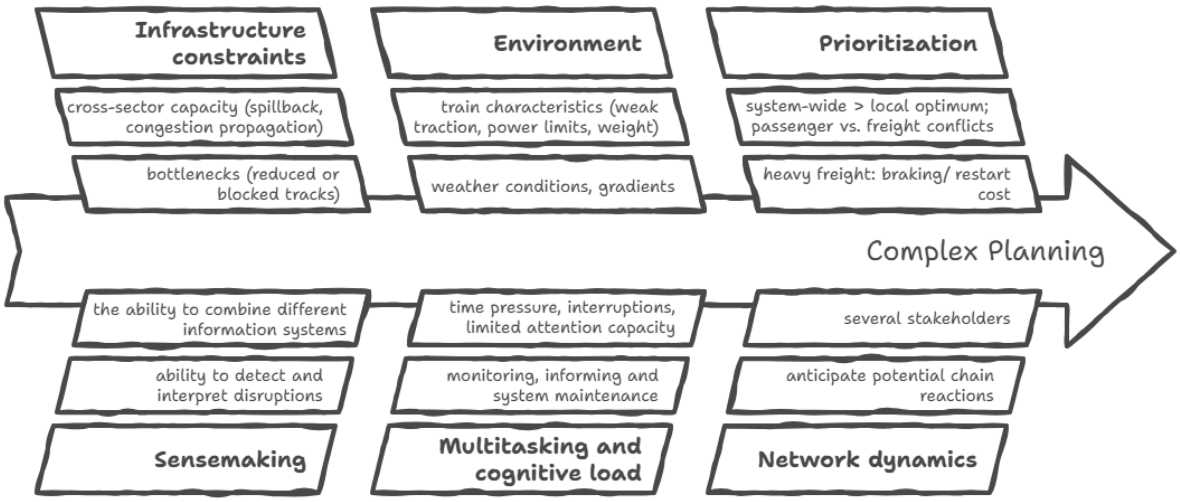


FIGURE 22 - KEY DRIVERS OF COMPLEX PLANNING IN RAIL DISPATCHING (SYNTHESIS OF OBSERVED COGNITIVE DEMANDS - LIST NOT COMPLETE)

oritization, sensemaking, multitasking/cognitive load, and network dynamics.

These demands become especially visible during disruptions. Under fault conditions, attention becomes a scarce resource, because detailed troubleshooting competes with the need to keep an overall live overview. As one participant described it: “The moment I start troubleshooting, I can no longer monitor the system”. Sensemaking demands also intensify in fault mode, where expertise becomes apparent through communication and the ability to translate signals into operational meaning: “[...] Especially in fault mode. That’s where you notice expertise in communication... The sector must know what to do; fault detection; interpretation of the fault (what does the fault mean?); knowing the appropriate solution”. The same account highlights that early detection is difficult for novices and attributes this to limited system knowledge, particularly the ability to read multiple systems in combination. Pri-

oritization is also shaped by physical and system-wide constraints. For example, heavy freight trains are costly to stop and restart, so it can be operationally better to keep them moving and accept small delays elsewhere to protect overall stability. Finally, controllability is not only local: even if there is still capacity in one sector, an overloaded neighboring sector can block movements and cause congestion to spread backwards ("spillbacks"). This makes early sequencing and cross-sector coordination important to prevent wider network effects: "[...] However, the actual bottleneck is not in [my] sector itself, but in the adjacent sector. Although there is still capacity available in the sector, trains cannot continue there because the neighboring section is overloaded. This also causes traffic congestion in the sector.".

6.3.4. WEAK SIGNALS

Continuous detection of weak signals (early indicators of potential future interruptions) is a key ongoing task for dispatchers. It happens during routine monitoring and coordination and relies on multiple information channels. Dispatchers become aware of weak signals across data through:

- operational system cues (e.g. conflict indications or malfunction messages),
- auditory cues (e.g. tones or alarms), and
- human communication, especially frequent calls or direct reports from all kinds of relevant people (dispatchers, shift leaders, other stakeholders)

Detection involves noticing when something deviates from expected patterns. Examples include unusually long block occupancy, atypical routing or track usage, or inconsistencies between system status and the unfolding operational situation. Interviews with experienced dispatchers revealed additional indicators that can trigger concern – or a "gut feeling" – even when no clear alarm is present. These include unclear or incomplete information that allows multiple interpretations, uncertainty or lack of communication from train drivers, ambiguous or atypical wording in communication systems, team composition (e.g., staffing and working experience mix), a perceived loss of control due to external factors, and unusually high traffic density of unusual timing and sequencing of messages and calls on a route. Primarily based on the observational data, we identified five recurring patterns of weak signals:

1. **Infrastructure anomalies**, such as switch, contact or track irregularities. These often signal emerging constraints before they affect the wider network.
2. **Train-related constraints**, including limited traction, readiness problems, or crew issues. These issues limit operational flexibility and may create downstream conflicts if they are not addressed early.

3. **Traffic irregularities**, such as unexpected early or delayed trains, atypical dwell or occupancy patterns, and emerging conflicts in train paths.

4. **Mismatches between plan and realty**, where system states or optimization outputs appear inconsistent with current conditions – for example, conflicts that persist despite changes, trains that remain marked as "active" even through they should no longer be operationally relevant, or atypical planned paths.

5. **Communication coherence problems**, where the content, timing, or framing of reports does not align with the situation, prompting dispatchers to question whether the overall picture is reliable.

An incident involving traction power illustrates the fifth pattern. A train driver reported lowering the pantograph and continuing to operate with a second pantograph, whereas the power specialist indicated that the voltage had returned to normal. However, despite these seemingly reassuring cues, the dispatcher became concerned because the timing and framing of the reports did not align with the situation. This included the absence of the expected event keyword that would have triggered the standard escalation process. This prompted the dispatcher to request that the line be secured and operations be constrained. A subsequent inspection confirmed extensive catenary damage (Observation protocol OBS-V1A8, Pos. 59–61). These results highlight the ongoing importance of human expertise in dispatching. Weak-signal detection depends not only on system alerts, but also on contextual judgment, communication cues, and coherence checks, which are difficult to fully formalize or automate. To ensure maintaining and developing that kind of expertise, we identified relevant knowledge and its learning sources to prevent deskilling, which will be the topic of section 4.4.5.

6.3.5. LEARNING: WIL-FRAM

The Functional Resonance Analysis Method (FRAM) makes it possible to describe sociotechnical systems as a set of functions and their interrelations (Hollnagel (2017)). In the case study, this method was adapted to understand learning processes by visualizing knowledge flows and information requirements. This serves two primary purposes: first, mapping the essential functions and cognitive demands of specific tasks; second, documenting how learning occurs within the system and identifying which information and knowledge are critical for skill development. This enhanced transparency supports the identification of functions that should remain with human operators to prevent expertise degradation (see D2.3 chapter 3.17.2 on transparency design requirements). The findings presented here emerge from completed analyses using the WIL-FRAM approach. Further methodological refinements and additional applications will be presented at the 17th International Conference on Applied Human Factors and Ergonomics (AHFE 2026) in July 2026.

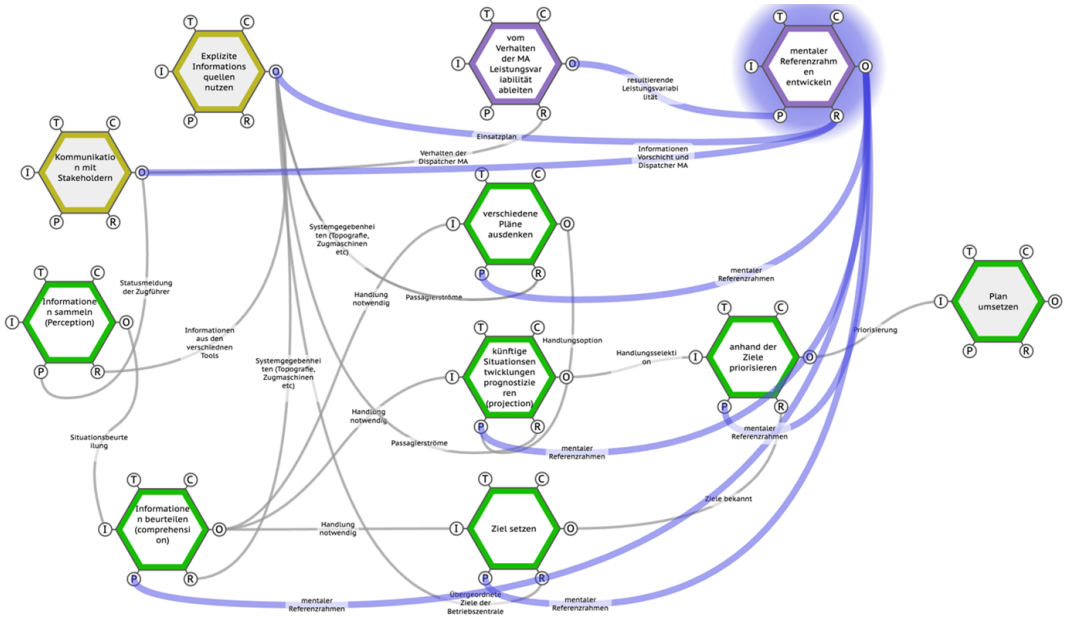


FIGURE 23 - SIMPLIFIED INSTANTIATION EXTRACTED FROM THE COMPLETE WIL-FRAM. GREEN FUNCTIONS REPRESENT THE DECISION-MAKING PROCESS, YELLOW FUNCTIONS DENOTE INFORMATION SOURCES, AND PURPLE REPRESENT EXPERIENTIAL LEARNING PROCESSES.

FRAM was selected for its capacity to model sociotechnical systems as networks of interacting functions rather than linear task sequences. By characterizing functions through their conditions, constraints, and interdependencies, FRAM enables nuanced representation of how system interactions shape overall performance. We modified the standard six aspects to better capture our target information: 'preconditions' now represent required working knowledge, whilst 'resources' denote necessary information to build situation awareness. Within our working-knowledge, information, learning FRAM (WIL-FRAM), learning operates at a conceptual level, emphasizing inter-functional relationships.

Figure 23 presents a simplified instantiation from the complete WIL-FRAM, streamlined for illustrative purposes to enhance comprehension. Interview data were analyzed deductively through the theoretical lenses of situation awareness (Endsley (1995)) and experiential learning (Kolb and Kolb (2009)). The analysis reveals that dispatchers construct what we term a "mental frame of reference" – a dynamic body of working knowledge concerning the state of the entire socio-technical system. This cognitive framework underpins numerous decisions, represented as functions within the WIL-FRAM model.

One dispatcher articulated this process, explaining how decisions incorporate employee experience levels: "Once you have that in mind, you can start building on your foundation or the basis of this layer, where someone might then have to say, yes, if an employee has a problem and no experience, then perhaps I need to take the scheduling area away from the other side, because they don't have the capacity to do anything with him" (interview protocol INT-7K3M, pos. 382). The WIL-FRAM demonstrates that this working knowledge develops through accumulated experience and experiential learn-

ing. This experiential foundation draws upon prior operational exposure and integrates information from multiple sources: handover communication with preceding shifts, staffing schedules, and ongoing dialogue with colleagues. These ongoing dialogues serve an additional critical function beyond immediate information exchange: they enable dispatchers to develop knowledge about performance variability among team members. This understanding of individual capabilities and working styles becomes integrated into the mental frame of reference, influencing strategic decision-making across diverse scenarios, including contingency planning, task prioritization and situation projection. These findings carry implications for AI implementation. For instance, dispatchers should maintain active roles that enable continued development of their mental frames of reference. Additionally, system design should preserve dispatcher access to the information sources essential for constructing and updating this working knowledge. Further design implications are reported in the case study report. Together with the results of the other analyses, these implications constitute the foundation of the Co-Learning concept, which is described in detail in the following chapter.

6.3.6. CO-LEARNING CONCEPT & HMI

Due to the complexity of the operational context and practical constraints, the co-learning concept could not incorporate all conceptual aspects in full depth. Instead, scope and feature priorities were guided through 1) field evidence, 2) theory (conceptual requirements and design principles from D2.3), and 3) feasibility. The final design therefore builds up on these three corners. In Table 2, the phases of human learning and connected to conceptual and planned HMI functions. An exemplary HMI has been developed within Interactive AI for the flatland simulator, the main screen of which is shown in Figure 24.

Phase	Conceptual Functions	Planned Function [Function ID]
Concrete Experience	Real-time effects of decisions on the networks KPIs [TP; EX] and Impact transparency [TP]	1. Module with risk assessment and impact of a specified action [**1.1**] • Calculating time buffer • Show number and what kind of required actions • Identifying affected sectors • Calculating impact of action on affected trains 2. Module with alternatives [**1.2**] • Alternatives of actions based on AI calculations
Reflection	Decision rationale replay [MR]	Reflection module [**2.3**] • Prompting reflection questions • Clustering of situations • Automatically summarize situations and related answers • Anonymized knowledge sharing
Abstract conceptualization	Include decision principles from past cases [TP; EX]	No module → Human formulates the rule; automated rule generation was out of scope, so reflection questions were implemented to support abstraction.
Active experimentation	1) Impact feedback on KPIs and on sectors around [TP] 2) Sandbox mode "what-if" [EX]	1. Summary over all incidents in a shift [**2.1**] • Showing all actions made in a shift and impact on KPIs • Showing all affected trains by actions 2. Potential event simulation [**2.4**] • Simulating experiences incidents and allow humans to experiment with different actions and see impact • Simulation non-experienced incidents of high-risk cases and allow humans to solve the problem and experiment with different actions

TABLE 2 - SUMMARY OF HUMAN LEARNING SUPPORT FUNCTIONS. SUPPORT FUNCTIONS ARE ALIGNED WITH THE CONCEPTUAL WORK OF D2.3 (Hamouche et al.). TP = TRANSPARENCY, EX = EXPERIMENTATION, MR = MIRRORING

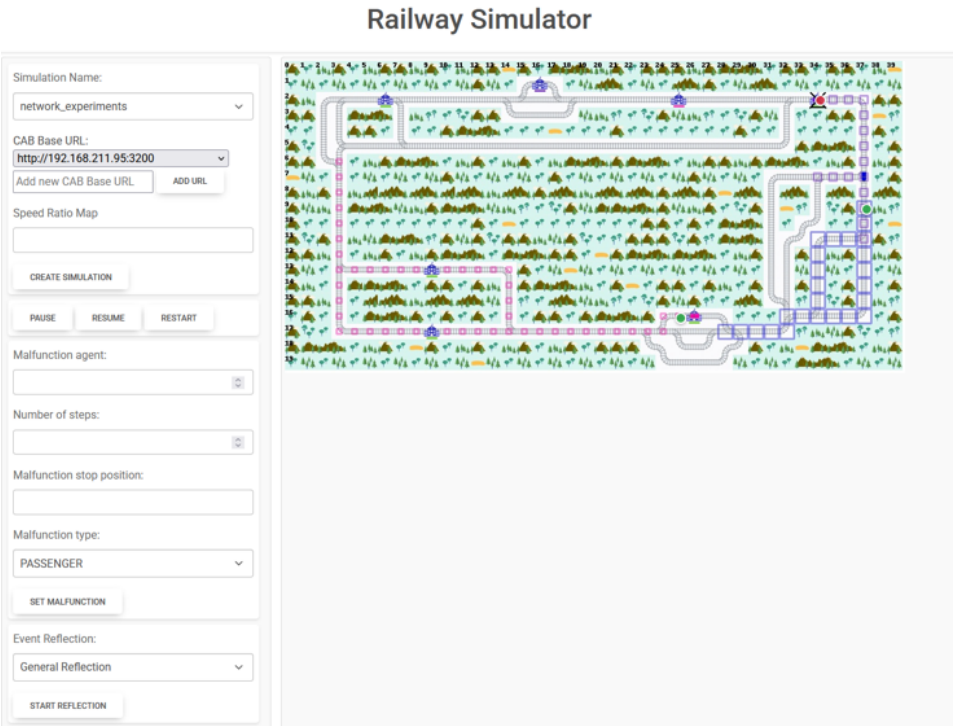


FIGURE 24 - MAIN SCREEN OF THE SYSTEMX FLATLAND SIMULATOR

The functions described in Table 2 are exemplified in the following using the aforementioned railway use-case, for which they have been implemented within the web-app flatland simulator, a sub-app of the Interactive-AI framework. The flatland simulator allows for the running of pre-specified scenarios and can be extended to allow for randomly generated environments.

These environments can then be loaded into the flatland simulator co-learning HMI and simulated. The simulator itself implements the functions listed in Table 2, with some additional data visualization and processing steps. Firstly, all train information is made visible to the operator via a train information window (Figure 25), providing the following data points:

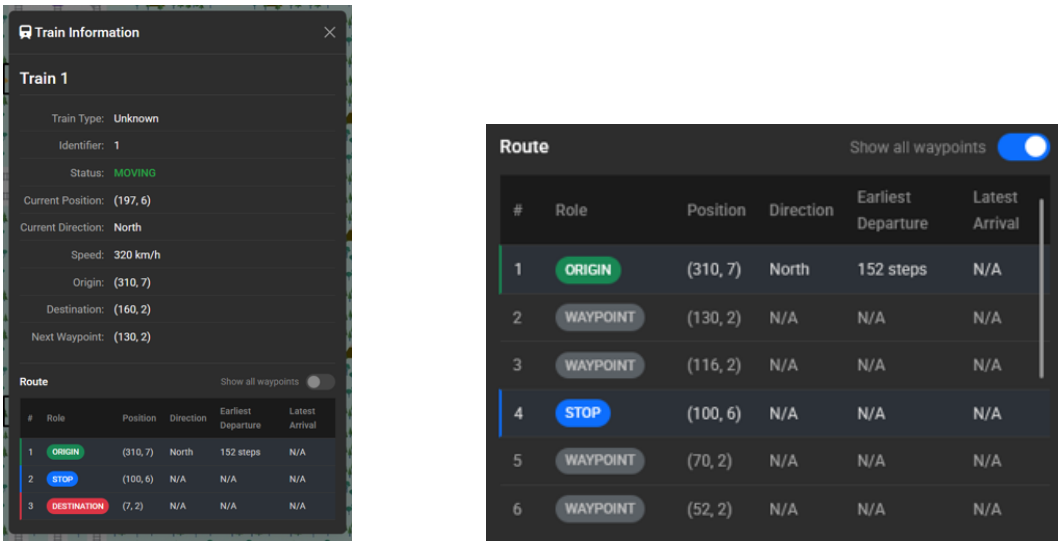
- **Train Type:** indicates the category of train. While these can be arbitrarily chosen in the simulator, standard types are cargo and passenger, which can be further divided into regional (slower) and intercity (faster) trains
- **Identifier:** the ID given to the train in the system, essentially its index
- **Status:** indicates whether the train is moving, waiting at a stop or malfunctioned
- **Current Position:** provides the current position of the train in the grid system, given as $(\texttt{rows}, \texttt{cols})$
- **Current Direction:** the current bearing of the train in the grid system, which is always one of the major cardinal directions

- **Current Speed:** the current speed of the train in km/h

Additionally to the listed information, a route overview is provided which shows the origin and target stations, as well as all planned stops (Figure 25(a)). Optionally, all route waypoints can be toggled via the “show all waypoints” button (Figure 25(b)).
When malfunctions occur, the operator can open a malfunction screen (Figure 26), which provides an overview of the malfunction, including the following information:

- Train ID of the malfunctioning train
- Malfunction type
- Criticality of the malfunction
- Duration since beginning of the malfunction
- Prediction of the malfunction duration
- Prediction of the delay the train will experience as a result of this malfunction

The operator is given the option to either have a solution generated by the system or formulate a solution themselves in natural language (Figure 27(a) and (b), respectively). This is then translated to a directive and context tokens, which are readable for the system. Solution formulation and generation correspond to the functions 1.1 and 1.2 given in Table 2, respectively.



(a) TRAIN INFORMATION OVERVIEW **(b) TRAIN WAYPOINTS TOGGLED**

FIGURE 25 - TRAIN INFORMATION VIEW WITH WAYPOINT TOGGLING.

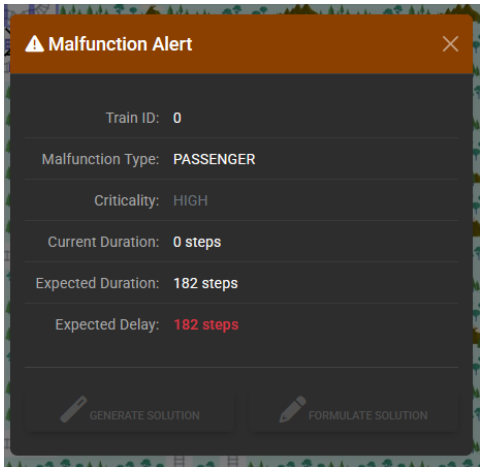
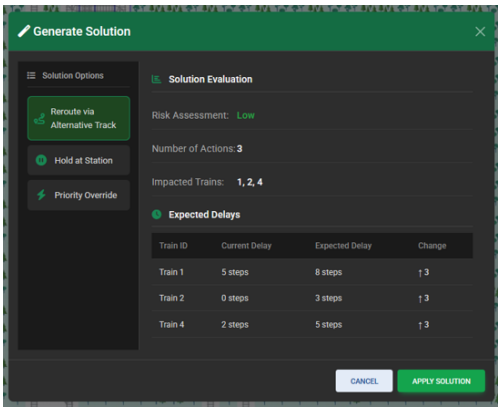
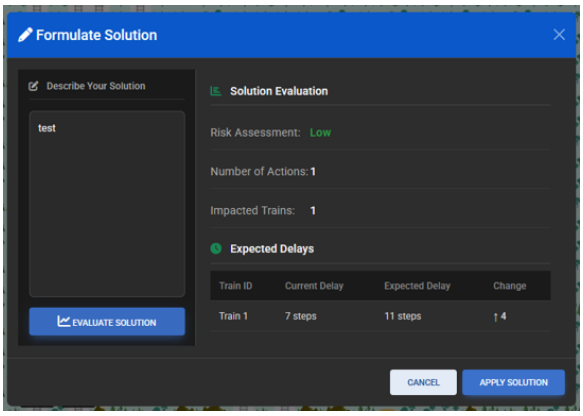


FIGURE 26 - MALFUNCTION WARNING WINDOW WITH EVENT INFORMATION



(a) SOLUTION GENERATION WINDOW



(b) SOLUTION FORMULATION WINDOW

FIGURE 27 - IMPLEMENTATIONS OF HUMAN LEARNING SUPPORT FUNCTIONS 1.1 (SOLUTION EVALUATION) AND 1.2 (ALTERNATIVE SOLUTION GENERATION)

In both cases, the solutions are evaluated and the results thereof presented to the operator, who can either formulate / generate another solution or choose to apply the selected solution. The evaluation of the solutions includes:

- Risk assessment of the solution
- Number of actions required for the solution
- Trains impacted by the solution
- Overview of the current and expected delays of all impacted trains

Finally, the operator is given the opportunity to reflect on their experiences. This can occur either during or after the simulation, and can be a general reflection on the whole experienced simulation or one specific event, as visualized in Figure 28.

When the reflection is executed, the operator is asked the following questions:

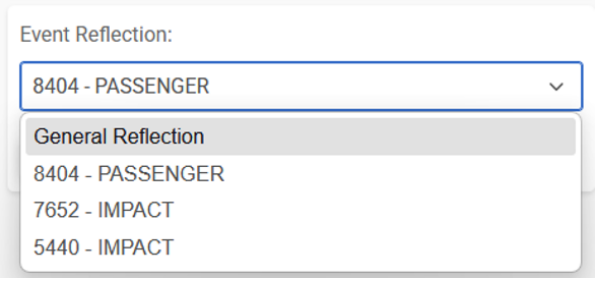


FIGURE 28 - DIALOG FOR REFLECTING ON EVENTS.

1. Did your solution lead to the expected result? [Yes/No]
2. Describe what happened. [Text answer]
3. What would you change next time? [Text answer]
4. What conclusions do you draw from this experience? [Text answer]
5. How do you measure success in this case? [Text answer]

The answers to these questions is logged anonymously together with the simulation and can be used to gain valuable insights into operator's decision making processes. Using a clustering model, knowledge transfer between operators and from more experienced to less experienced operators can be achieved.

7. AUTONOMOUS AI-DRIVEN DECISION SYSTEMS

7.1. INTRODUCTION

This chapter details the system architecture developed for autonomous AI-driven decision systems. Despite the AI system being autonomous, the nature of critical systems requires a certain level of human oversight, which further requires the operator supervising the system to understand both the system and the AI agents operating within the system. While the integration of automation can significantly improve system performance, it is also likely to infringe upon operator autonomy, lead to the loss of important skills and knowledge, and lack of experienced meaningfulness and motivation - entirely dependent on the interaction design. The integration of knowledge from work psychology is therefore essential for the successful alignment of the autonomous system with human cognitive requirements. The following chapter describes the autonomous system developed that maintains human autonomy without losing the benefits of autonomous systems.

The system developed is referred to as the “Director System”, given that the human role in the system is inspired by orchestra directors - the operator orchestrates the system by issuing high-level “directives” that are then autonomously executed by the individual system components. While the director does not need to be able to execute the individual functions, they understand the process and what influence they have over it. Referring back to the orchestra director - while they can likely not play every instrument themselves, they know what each instrument should sound like and what result they wish from the musician playing it. This proposed director system possesses significant benefits over standard “human-on-the-loop” monitoring systems due to its more active integration of the human supervisor and alignment.

7.1.1. REQUIREMENTS TO THE AUTONOMOUS SYSTEM

In safety-critical control settings, introducing higher levels of automation often changes the human role. When more decision logic is handled by automation, people often shift from doing the work to watching it and stepping in only when something goes wrong. This shift creates well-known human-factors risks: sustained monitoring is hard to perform reliably, and humans can become “out of the loop,” with reduced engagement and weaker situation awareness (Bainbridge (1983);Endsley (2023)). One common way to deal with this issue is to add explanations (e.g., “why the AI recommends X”). The idea is: if the systems becomes a black box, explanations should help people validate outputs. But explanations alone do not necessarily fix the underlying work problem: they often do not change the

human role form passive monitoring back to active control. Buçinca et al. (2025) make this point more concrete: they found that adding explanations does not automatically lead to better human judgment or appropriate reliance. This is because humans do normally not cognitively engage with explanations when their role is to passively monitor AI. For this reason, we consider the black-box problem of autonomous AI a work-design problem: integrating humans is not only a question of "keeping a human in the loop by providing explanations," but of shaping tasks and function allocation so that humans retain a meaningful and active role (e.g., setting priorities, constraints, and strategies) and remain prepared to intervene effectively.

To address this limitation, we build on the concept of interpretable primitives (Wahde and Virgolin (2021). Interpretable primitives are small, well-defined operations (e.g., select, compare, filter) that can be chained into a transparent pipeline with explicit inputs and outputs at each step. Instead of giving users a single recommendation to accept or reject, the system exposes the intermediate steps. The key point is that this supports a different work role: humans are not just validators of a final recommendation, but conductors of the procedure. They can decide which primitives should be executed and in which order. The AI then carries out these steps and returns traceable intermediate results. This keeps the work cognitively engaging and intervention ready.

Hierarchical Task Analysis (HTA) and Primitive Derivation. Hierarchical Task Analysis (HTA) is a method for describing work by breaking down an overarching goal into sub-goals and the tasks/subtasks that contribute to them. The result is a structured hierarchy that makes both what people do and why they do it explicit (Stanton et al. (2017)). The HTA structure is useful for primitive derivation for different reasons. First, its hierarchical structure makes potential primitive-sized work units visible: subtasks often describe a coherent function with a limited scope, which aligns with the idea of primitives as small, autonomous building blocks (Wahde and Virgolin (2021)). Second, HTA links actions to the purpose they serve (goal/sub-goal context). This supports formulating primitives with an explicit purpose instead of a list of isolated actions. HTA is also a good fit because it does not only link actions to goals but often makes input-action dependencies explicit: higher level units imply the input which is required (e.g. "Check feasibility of the rerouting"), while lower-level tasks describe the operations required to produce that input. (e.g. "Check whether the train type fits the rerouting route in general", "Check that the train can physically operate on that route" etc.). This aligns with the pipeline-oriented view of Wahde and Virgolin (2021), where complex behavior is built from interpretable primitives that transforms explicit inputs into outputs in a traceable sequence.

We derive the HTA from domain input by eliciting what railway traffic controllers do during disruption management and why they do it, primarily through interviews. The material is then structured by coding goals and the corresponding tasks, so that actions are not listed as isolated steps but linked the

intention they serve. Based on this coding, the HTA hierarchy is built by organizing the overall goal, subgoals, and tasks/subtask in a consistent goal-to-task structure. We then use the HTA as the starting point to derive primitive candidates. Starting at the lowest level, we screen tasks/subtasks bottom-up against three criteria: ”clarity of purpose” (immediate understanding of what the primitive is for—the goal it addresses), ”process transparency” (traceability of how the primitive works—i.e., the steps or rules leading to the result), and ”granularity” (level of action/abstraction: bounded scope and suitable as a composable unit). If a unit scores poorly, we move one level up in the HTA hierarchy and re-evaluate the higher-level unit, repeating this until the candidates can be rated well on the criteria. For detailed criteria, indicators and the process, see Dettling et al. (2026). Figure 29 provides a small HTA example from disruption management. It shows how the overarching goal (stabilize operations and maintain passenger journey chains) is decomposed into a subgoal (“passengers reach their destination via authorized rerouting routes”) and then into a concrete dispatcher intent such as “Search suitable trains for rerouting”. In our approach, this intent can be expressed as a command input and is executed by composing smaller, interpretable primitives that score well on our criteria. For example, the system can (1) identify affected trains and routes, (2) filter by train type / capacity (prioritizing high transport capacity because these trains can take additional passengers, e.g., after extraordinary stops), and (3) filter for direct-to-destination services (to avoid connection losses). The output then can be ranked as a shortlist of candidate trains.

7.2. AUTONOMOUS SYSTEM ARCHITECTURE

The architecture for the proposed director system is visualized for the railway use-case in Figure 30, as the railway use-case is the only one which foresees the implementation of such autonomous systems. The proposed director system architecture is illustrated in Figure 30 using the railway use case, as this is currently the only domain expecting autonomous deployment. However, it is designed to be transferable to air traffic and power grid systems by leveraging their common graph-based structure. Given any environment that can be represented as a graph, the same methods can be applied. In the railway context, the architecture includes: a simulation environment with a graph model, a negotiation proxy, a human director, reinforcement learning agents, a learning algorithm, and an experience clustering module. In the case of the railway domain, the simulation environment is flatland, of which a graph representation is generated. In this graph representation, switches and stations are represented as nodes. For each track connecting these nodes, one edge exists. These edges can be given domain-specific attributes, for example the maximum possible speed for railway applications. The graph thereby encodes all topological context and information given by the grid system of flatland and allows for direct manipulation by the human agent to add context information (for example, removing edges if the system does not recognize infrastructure malfunction).

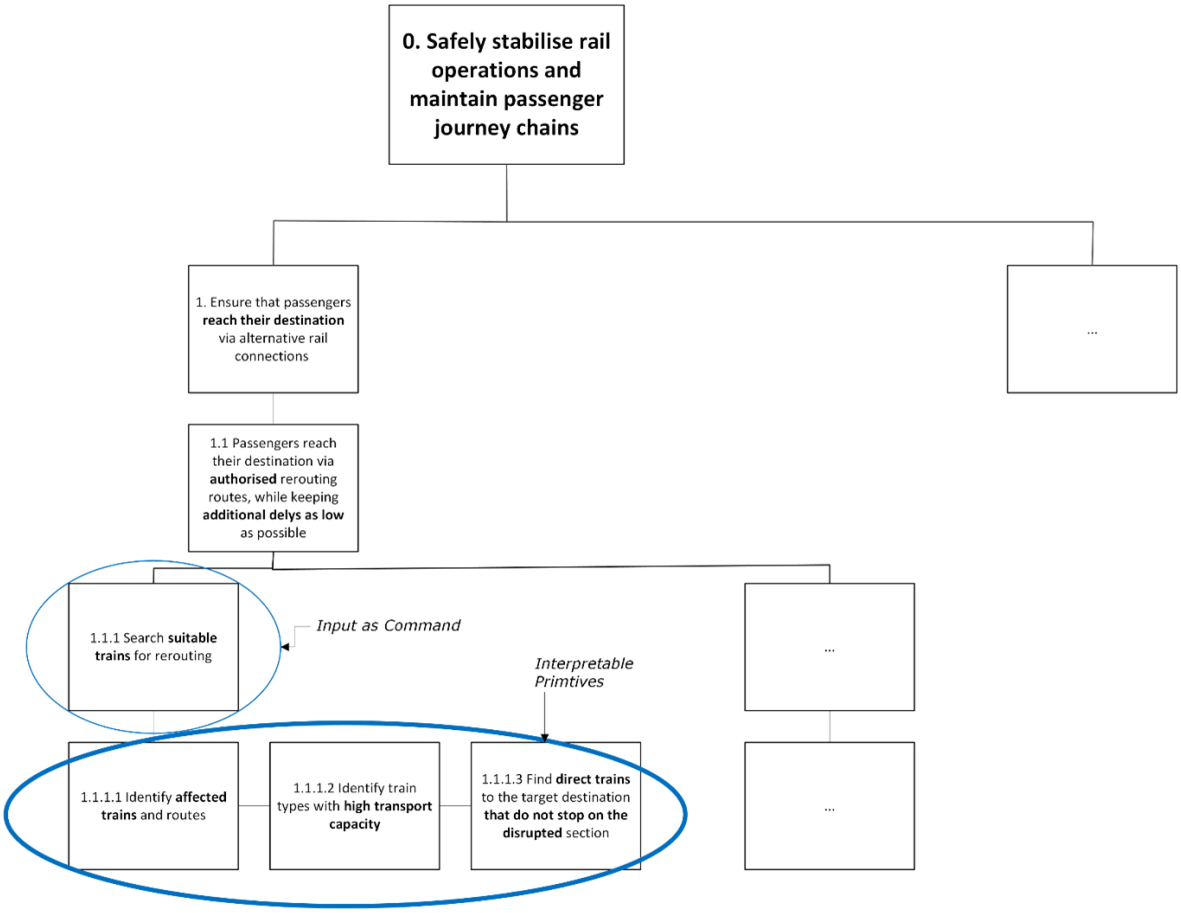


FIGURE 29 - HTA SNIPPET SHOWING THE TASK DECOMPOSITION FOR DISRUPTION MANAGEMENT.

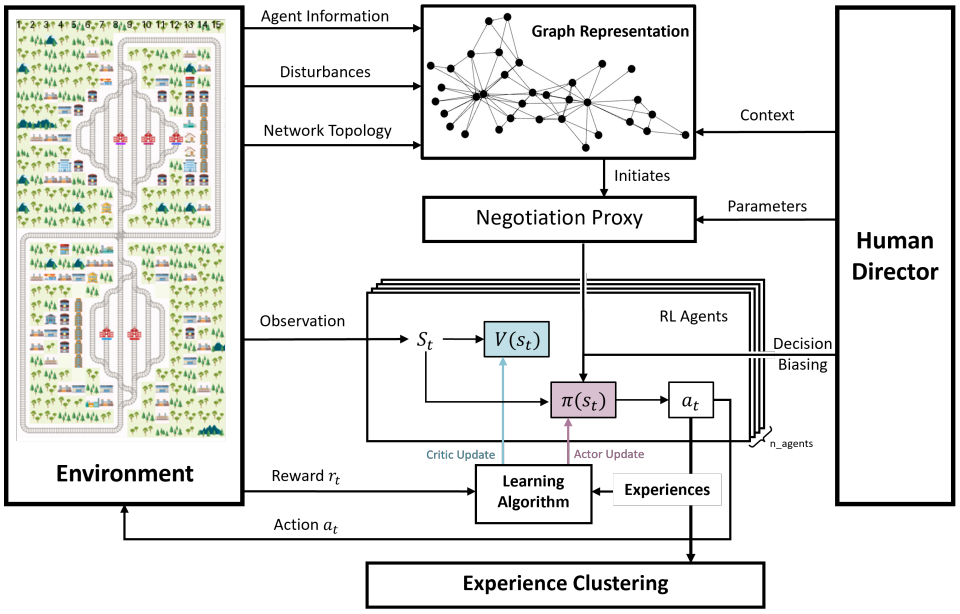


FIGURE 30 - AUTONOMOUS ARCHITECTURE FOR RAILWAY USE-CASE

Within the environment, each train is an individual agent, which learns to navigate the environment and handle disruptions. When conflicts are predicted between agents in the environment, a negotiation proxy is instantiated, enabling communication between the conflicting agents. This communication is planned to take the form of a bidding process, wherein the agents negotiate the local best course of action. The result of this negotiation then overrides the predicted best course of action, the "loser" of the negotiation must recalculate their path. In the following sections, we will explain and motivate the implementation in detail.

7.3. GRAPH-BASED REPRESENTATION OF THE FLATLAND ENVIRONMENT

A Flatland simulation environment can be represented using different graph abstractions, depending on the intended application. At the most detailed level, each grid cell may be mapped directly to a graph node. However, more compact and computationally efficient representations can be obtained by focusing only on decision-relevant cells, such as switches or crossings.

For our railway applications, it is not sufficient to represent only the physical rail layout. Instead, the feasible movement options of an agent must also be captured, taking into account the agent's incoming direction. In railway systems, the set of possible successor cells depends on the direction from which a train enters a track segment. To model this directional dependency, each grid cell is expanded into multiple graph nodes, one for each possible entry direction.

7.3.1. GRAPH CONSTRUCTION

The implemented solutions based on the `FlatlandGraphBuilder` (Flatland ExtensionTools) converts Flatland's grid-based topology into a directed graph $G = (V, E)$. Each node $v \in V$ represents a unique combination of position and entry direction, defined as

$$v = (x, y, d),$$

where x and y denote the horizontal and vertical grid coordinates of the underlying Flatland cell, and d denotes the direction from which the agent enters the cell. Directed edges represent feasible transitions between nodes. An edge $e = (u, v) \in E$ exists if and only if a valid Flatland transition allows an agent located at node u to move to node v. The feasibility of transitions is determined using Flatland's native navigation rules. Since not all neighboring cells are reachable from every entry direction, the directional component d is essential for correctly constructing the graph.

Each edge is associated with several attributes:

- **Length**: By default, the edge length is one grid unit. If infrastructure elements are defined, the

edge length corresponds to the infrastructure-specific length assigned to the underlying cell.

- **Resource**: The resource associated with an edge corresponds to the Flatland cell underlying the source node u. This enables modeling of mutually exclusive resource usage.
- **Action**: The Flatland action required to move from node u to node v.

7.3.2. GRAPH SIMPLIFICATION AND DECISION NODES

The graph representation enables efficient computation of shortest paths and other routing tasks using standard graph algorithms. The implemented graph allows these computations to be performed efficiently. For optimization purposes, the graph can be further simplified by retaining only decision-relevant nodes and merging intermediate nodes along linear track segments. A decision node corresponds to a cell (or track segment) where an agent can meaningfully choose between alternative actions, such as stopping, continuing, or branching onto another route. Examples include switches, crossings, or merging points.
In contrast, waiting on straight track segments between decision points is typically undesirable. An agent that stops before reaching a decision node blocks all intermediate resources, thereby unnecessarily restricting other agents. From an optimization perspective, it is therefore advantageous to collapse such linear segments into single edges, reducing graph size and improving computational efficiency.

7.3.3. SHORTEST PATHS AND DECISION-NODE GRAPH SIMPLIFICATION

Once the Flatland topology has been translated into a directed graph, many navigation and planning problems become standard graph problems. In particular, finding a shortest connection between two states, for example from a start node A to a target node B, can be solved with classical shortest-path algorithms (e.g., Dijkstra or A*, MultiAgents RL-Solver). Instead of reasoning directly on the grid with domain-specific rules, the planner operates on nodes and edges with well-defined costs (such as distance or traversal time). The developed agent system supports these computations efficiently by exposing direct access to the graph structure and its edge attributes, enabling fast route queries even for large environments. This includes the routing conflict solving parts as well as the communication with the dispatcher.

Although the full graph representation is accurate, it may contain many nodes that do not contribute meaningful decisions. Along long straight track segments, an agent often has no real choice: it can only continue forward until it reaches a switch, a crossing, or another operationally relevant location. Keeping every intermediate cell as a separate node increases the problem size without adding

decision-making power. Therefore, the graph has been simplified by removing such non-decision nodes and merging them into longer edges. We therefore implemented the so called *decision nodes*:

A *decision node* is defined as a node (or the underlying Flatland cell) where an agent can reasonably choose among distinct operational actions, such as:

- **Stop or proceed:** a location where waiting is meaningful because it does not unnecessarily occupy long track sections.
- **Branching:** a switch or fork where multiple successor routes are available.
- **Conflict-prone infrastructure:** crossings, merges, or other locations where train interactions must be coordinated.

In other words, decision nodes are the points where routing and scheduling decisions actually matter. Allowing an agent to wait at arbitrary points on a straight segment is typically disadvantageous. If a train stops in the middle of a corridor, it occupies (and therefore blocks) a sequence of resources behind and in front of it, depending on the resource model. Operationally, this is undesirable because it prevents other trains from using the same track section, even though the stopped train has not yet reached the location where conflicts occur.
A typical example is a crossing ahead. If the agent (train) stops far before the crossing, it blocks all intermediate cells between its current position and the crossing. From a scheduling and optimization perspective, this rarely makes sense: it is usually preferable for the agent to move forward to a designated decision point (e.g., the cell immediately before the crossing) and wait there. This reduces unnecessary resource occupation, keeps more track space available for other agents, and concentrates decisions at locations where they influence interactions.
The simplification process therefore collapses linear sequences of non-decision nodes into a single edge between two decision nodes. The resulting reduced graph preserves the important operational structure, while significantly lowering the number of nodes and edges.

7.3.4. IMPLEMENTATION

The graph construction and manipulation are implemented using the `networkX` library. This provides access to a wide range of graph algorithms and utilities, facilitating routing, analysis, and further extensions of the model.
In the existing representation (direction-expanded model), graph nodes are distinguished not only by their grid location but also by the cardinal direction. This modeling choice is useful to encode direction-dependent feasibility constraints, but it has an important side effect: the graph becomes fragmented

with respect to physical proximity. In particular, locations that are geographically close in the grid (e.g., adjacent track elements around a switch or crossing) may appear far apart in the graph because they are connected only through sequences of direction-specific nodes. As a result, the graph distance between two physically close agents can be large even though they operate within the same local infrastructure.

This fragmentation becomes problematic for coordination and communication (inefficient information propagation). If two agents are geographically close and should quickly exchange information (for example, to detect an imminent conflict), messages often cannot be routed through a small local neighborhood in the graph. Instead, they must propagate through a chain of nodes that are graph-reachable but geographically irrelevant. In practice, this increases overhead and delays local conflict awareness, because the message traversal is driven by graph topology rather than by physical locality.

7.3.5. THE MULTIDIGRAPH SWITCH COMPRESSION

As a core idea, the proposed solution replaces the direction-expanded representation with a *MultiDiGraph* abstraction. In this new graph:

- each switch is represented by **exactly one node**, and
- alternative tracks, routes, or directional movement options are represented as **multiple parallel directed edges** between nodes.

Hence, instead of duplicating nodes to encode directionality, the model keeps the node set compact and shifts the variability into the edge set. The simplified graph is collapsing up to four direction-specific nodes into a single switch node immediately reduces the total number of nodes. At the same time, using multiple edges preserves the existence of distinct track options without inflating the vertex set. This yields a substantially smaller and structurally clearer graph, while still allowing the system to distinguish different movement possibilities through edge attributes (e.g., track identifier, traversal cost, resource set, or required action).

7.3.6. ALGORITHMIC BENEFITS: FASTER CONFLICT RECOGNITION AND NEGOTIATION

The MultiDiGraph representation supports faster and more efficient conflict recognition, because local interactions are evaluated within a smaller, more geographically coherent subgraph. When a potential conflict is detected, the system can trigger conflict negotiation via the negotiation proxy earlier and with less computational overhead. In short, the new representation improves both *efficiency* (smaller graph, fewer irrelevant traversals) and *responsiveness* (local conflicts become visible sooner), which is essential for scalable multi-agent railway coordination.

A key advantage of the new representation is that geographically meaningful structures remain grouped together. Since the graph no longer separates a single physical element into multiple direction-dependent vertices, local infrastructure (e.g., the set of tracks meeting at a switch) remains local in the graph. Therefore, graph neighborhoods align better with physical neighborhoods: agents that are close on the grid tend to also be close in the graph.

7.4. NEGOTIATION

A particularly relevant collaboration pattern for critical network operations is negotiation-based mixed-initiative control, where AI supports coordination through structured negotiation, while the human operator retains governance intent and can adjust policy-level settings when required. In large-scale infrastructures (e.g., railway traffic management under major disruptions), many operational conflicts are inherently local and resource-based: multiple actors compete for limited capacity at junctions, shared single-track segments, and short-horizon edge-time slots. Rather than relying on a single global plan that may be brittle under uncertainty, negotiation provides a modular mechanism for resolving conflicts as they emerge, while preserving a clear division of responsibility between high-level governance intent and algorithmic execution.

In this prototype, per-agent action proposals are generated first, then a centralized dispatcher resolves contested resource-time requests via deterministic auction-style allocation with reservations. When several trains request the same resource in the same time window, each requester is scored using explicit components (priority proxy, stuck steps, fairness loss streak, delay proxy), and the allocation is decided by deterministic ranking with anti-starvation and tie-break rules. The resulting process is interpretable rather than hidden: each contested allocation produces a structured trace of requests, bids, active constraints, winner identity, and reason codes.

The operator role shifts from manual micromanagement of individual conflicts to rule-level steering and exception policy. Rather than selecting every train action, the operator configures high-level priorities and safeguards that shape negotiation outcomes—for example, fairness limits, anti-starvation thresholds, and weight settings in the bid function. In this MVP, this governance is primarily realized through configuration and scripted policy patches, with optional predefined resource overrides for targeted situations. This preserves meaningful human authority at the policy layer while allowing AI to handle high-frequency coordination work during disturbances.

A key element of the approach is the Guardian supervision layer: a lightweight monitoring component that continuously tracks negotiation dynamics and system evolution, detecting emerging failure modes such as prolonged no-move streaks, congestion buildup, and repeated unfair outcomes. The Guardian does not replace negotiation; it evaluates whether the negotiation process remains stable and policy-compliant. When risk indicators exceed thresholds, it applies bounded governance

patches (e.g., fairness-weight increases, priority-weight reductions, anti-starvation tightening, and limited override activation) to recover flow while maintaining deterministic behavior.

Negotiation-based mixed-initiative collaboration also imposes concrete transparency requirements. In safety-critical operations, it is typically insufficient for an AI system to output an action recommendation without context. The operator needs operationally meaningful explanations: (i) active constraints and priorities that shaped the negotiated outcome, and (ii) a concise rationale for each allocation decision (e.g., fairness guard, junction safety, or override). In this implementation, explainability is provided through step-wise JSON negotiation transcripts, compact reason codes, and governance snapshots, which align with operational mental models and support calibrated trust without exposing internal model complexity.

Overall, negotiation-based mixed-initiative control provides a practical way to combine scalable AI coordination with human governance intent. It supports robust operation under uncertainty by resolving conflicts locally, enables policy steering through adjustable constraints and thresholds, and preserves the ability to apply targeted interventions when required—forming a concrete foundation for interactive, human-centered AI in real-world network management.

7.4.1. DIRECTIVES

To allow humans influence over the autonomous AI system, the human is given the option to provide high-level **directives** to the system, which are autonomously executed. An example for such a high-level directive could be:

- “Prioritize regional trains over long-distance connections”
- “Reduce traffic load in sector A”
- “Track 5 in Bern is malfunctioning, reroute traffic”
- “Reroute Train 274 via Bern”

The listed directives provide one of three types of information: context information, a parameter for solution generation or an explicit command. In the case of context information, the human provides information that the AI system is either not capable of receiving or has not received due to an error. In Figure 30, context information flows directly, as described previously. Directives which inform solution generation and negotiation are considered parameter directives, and flow into the negotiation proxy. For example, the directive to prioritize long-distance connections informs negotiation by giving long-distance trains a higher weighting. Finally, explicit commands flow directly into the policy via decision biasing.

FIGURE 31 - GENERALIZED STRUCTURE OF COMMUNICATION USING TOKENS.

(a) TRAIN PRIORITIZATION (b) STOP CANCELLATION

FIGURE 32 - TWO EXAMPLES OF PRIMITIVE DESIGN FOR THE RAILWAY RESCHEDULING USE-CASE.

Three key considerations influenced the design of the communication protocols for the above mentioned interaction modes. The first is that the architecture must allow for the addition and adaptation of the interaction modes. The second is that it allow for further refinement and development, without the development of an initial proof of concept be too complex. Finally, the developed architecture must be in line with existing human strategies of communication, making it intuitive for non-AI experts. The solution developed to encode the directive is a token-based architecture, using tokens to formulate ”prompts” which are discrete, and can be passed directly to the AI agent.

The architecture considers two types of tokens: action tokens and context tokens. The **Action tokens** $t_a \in T_A$ identifies the type of input the human wishes to give to the AI system, which then requires one or more environment-grounded **context tokens** $t_c \in T_C$ to provide situational details. A series of tokens is referred to as a **primitive**. For each action token t_a, a subset of context tokens are available.

$$T_{C_{t_a}} \subseteq T_C \qquad \forall\, t_a \in T_A \tag{47}$$

Figure 31 depicts the general structure of a primitive with a variable length of context tokens. Importantly, the order of these context tokens can influence the meaning of the primitive, which is referred to as the **syntax**.

Additionally, each action token requires a specific syntax, meaning that the order of the context tokens influences the meaning of the primitive. These primitives can then be processed by the AI agent.

7.5. THE AUTONOMOUS AI FORMULATED WITHIN A MARL FRAMEWORK

Railway traffic management is formulated as a multi-agent reinforcement learning (MARL) problem within an existing autonomous decision-making framework, where each train is modeled as an independent decision-making entity. These agents interact within a shared infrastructure, compete for limited resources, and must coordinate implicitly to avoid deadlocks, minimize delays, and resolve

conflicts.

The Flatland simulator acting as environment, provides the simulation backbone for this setup as explained above. It models railway topology, movement constraints, and resource conflicts, while enabling the integration of realistic operational extensions such as multiple resource allocation and vehicle dynamics. Within this environment, each agent learns a policy that maps observations (e.g., local track structure, occupancy, and routing information) to control actions (e.g., move forward, stop, or choose a branch). The learning objective is to optimize long-term performance under dynamic and highly interactive conditions. This is done over a token communication protocol explained above.

Dedicated RL Implementations for Flatland. To support Autonomous AI in this setting, we developed dedicated Flatland-compatible implementations of state-of-the-art reinforcement learning algorithms. The implementations are modular and customizable, allowing flexible experimentation. Furthermore, they are designed to support scalable parallelization, which is essential for handling the computational complexity of multi-agent railway scenarios.

Our work focuses on variants of Proximal Policy Optimization (PPO), a policy-gradient method known for stable and reliable learning behavior in high-dimensional and stochastic environments. As a reference configuration, we employed a standard, non-parallelized PPO setup. In this approach, a single learner interacts with one instance of the Flatland environment, collects rollouts (state-action-reward trajectories), and periodically updates the policy network. While conceptually simple and stable, this setup results in comparatively long training times due to sequential data generation and learning. To enable scalable Autonomous AI training in Flatland, we developed two parallelized PPO pipelines that distribute rollout generation across multiple workers, each interacting with its own environment instance.

In the synchronous configuration, all workers generate rollouts in parallel using a shared policy. A network update is performed only after all workers have completed their assigned experience collection. The aggregated rollouts are used to compute gradients, and the updated parameters are then distributed back to the workers.

This approach ensures consistency, as all workers operate on the same policy version during data collection. However, it introduces synchronization overhead: the update step must wait for the slowest worker to finish, which can limit overall training speed.

To further accelerate training, we implemented an asynchronous PPO variant inspired by the IMPALA architecture. In this setup, workers continuously generate rollouts and send them to a central learner. The learner updates the network as soon as a sufficient number of rollouts has been received, without waiting for all workers to synchronize.

This continuous update mechanism significantly improves hardware utilization and reduces idle time.

It enables faster convergence and better scalability, particularly in large and computationally demanding multi-agent Flatland environments. Although asynchronous learning introduces additional architectural complexity, it is well suited for dynamic railway scenarios with many interacting agents.

8. GENERALIZATION OF RESULTS

The developments achieved within AI4REALNET have been designed with generalization in mind; however, their transferability is not uniform across all components. The project's contributions range from high-level methodological principles to domain-specific model implementations. While the overarching framework—such as the human–AI collaboration paradigm, co-learning support concepts, and interaction structures—can be readily adapted to other network-structured environments, certain technical models remain inherently tied to the operational characteristics of the domains in which they were developed. In particular, AI models that rely on domain-specific data, constraints, and operational rules cannot be transferred directly without retraining or substantial adaptation. Table 3 on page 119 therefore distinguishes between elements that are broadly reusable and those that depend on context-specific assumptions or infrastructure characteristics. This differentiation underscores that AI4REALNET's primary contribution lies in its transferable design principles and architectural patterns, whereas the instantiated models themselves may require careful reconfiguration when applied to new environments.

8.1. DOMAIN INDEPENDENCE OF THE METHODOLOGICAL FRAMEWORK

Although this work is illustrated through applications in power grid control, air traffic management, and railway operations, the proposed methods are intended to generalize to a broad class of network-structured problems. Key components—such as the co-learning human support functions and the "director mode" interaction paradigm—are largely domain-independent, as they are grounded in fundamental principles of human learning and human–AI collaboration. These elements can therefore be adapted to other operational contexts with relatively limited structural changes. At the same time, the underlying AI models are inherently tied to domain-specific data, constraints, and operational objectives, and thus require retraining or adaptation when deployed in new environments. In this way, AI4REALNET contributes not only domain-validated implementations, but more importantly a transferable human-centered AI framework that moves beyond domain-specific automation toward reusable principles of collaborative autonomy.

8.2. ARCHITECTURAL AND ALGORITHMIC ABSTRACTION ACROSS NETWORK-STRUCTURED SYSTEMS

The AI4REALNET solutions are built around human-centered control modes, modular service-oriented architectures, and advanced learning and optimization methods, which are inherently applicable to any complex network-structured system. Railway networks, power grids, and air traffic systems share structural properties such as distributed interdependencies, cascading effects, multi-objective trade-offs, and safety-critical constraints. The algorithms and architectural principles developed—including uncertainty modeling, multi-objective optimization and reinforcement learning, conformal risk assessment, and agent-as-a-service encapsulation—are therefore formulated at a level of abstraction that transcends these three domains. In this sense, the AI4REALNET use cases serve as concrete validation environments rather than limiting application contexts.

8.3. TRANSFERABILITY OF CO-LEARNING AND HUMAN-CENTERED SUPPORT MECHANISMS

The co-learning and human learning support functions, in particular, are designed around general cognitive and organizational principles rather than domain-specific heuristics. As outlined in the interactive AI and co-learning sections, these systems are designed to support human decision-making processes, including hypothesis evaluation, alternative generation, reflection, trust calibration, and mental model development. These cognitive processes are not unique to railway dispatchers or air controllers; they characterize expert decision-making across safety-critical domains. Consequently, the mechanisms for interactive feedback, preference learning, explanation, and structured self-reflection generalize well to any environment in which human operators must reason under uncertainty, manage trade-offs, and maintain situation awareness. While the domain interface, for example, visualization of trains versus power lines or airspace sectors, must be adapted, the underlying human learning dynamics and the AI modules that support them remain structurally consistent.

8.4. SUPERVISORY AUTONOMY AND POLICY TRANSFER LIMITATIONS

Similarly, the proposed “director mode” or supervisory autonomy paradigm, corresponding to trustworthy full-AI control under human oversight, follows a generalizable architectural principle. As described in the sections on autonomous AI-driven decision systems and the Agent-as-a-Service (A3S) framework, autonomous agents are encapsulated within transparent, inspectable, and uncertainty-

aware service layers. This separation between control policy, uncertainty estimation, explanation generation, and supervisory interface is domain-agnostic by design. However, while the architectural scaffold generalizes, and the underlying learning and optimization algorithms can in principle be applied across domains, the learned policies themselves do not automatically transfer across environments. Reinforcement learning agents trained on a railway topology cannot be expected to operate effectively in a power grid or air traffic environment without retraining, as their learned representations encode environment-specific dynamics, constraints, and reward structures. In this sense, autonomy architectures and algorithmic methods are structurally generalizable, but policy competence remains environment-dependent.

8.5. FUTURE DIRECTIONS TOWARD TRANSFERABLE PRIMITIVES

Although beyond the immediate scope of AI4REALNET, this line of work opens a highly relevant research direction: the development of transferable primitives and higher-level abstractions that significantly reduce the need for task-specific retraining. Instead of relying exclusively on large-scale foundation models, an alternative is to advocate for modular, reusable building blocks–domain-informed representations, structured embeddings, physics-aware features, and composable learning components–that can generalize across related tasks and system configurations. For instance, the document already points toward mechanisms such as inverse reinforcement learning, preference-based reinforcement learning, and modular multi-objective optimization. These approaches suggest the possibility of learning reusable behavioral building blocks, such as conflict resolution, congestion mitigation, or load balancing strategies, that can be adapted to new network contexts through observation of human operators and interactive refinement rather than full retraining from scratch. Future work could therefore explore meta-learning, domain-general representation learning, and human-guided adaptation as pathways toward more scalable generalization.

8.6. SUMMARY

In summary, while the empirical demonstrations are grounded in the three projects' domains, the conceptual contributions and algorithms underlying human-centered control modes, uncertainty-aware decision support, co-learning architectures, multi-objective trade-offs, and modular supervisory autonomy are explicitly designed to generalize to any complex, safety-critical, network-structured system. Domain adaptation primarily affects learned policies and interface representations, whereas the structural principles of collaboration, uncertainty communication, and human oversight remain broadly transferable across infrastructures.

TABLE 3 - APPLICABILITY AND TRANSFERABILITY OF CONTRIBUTIONS

Result	Knowledge representation type	Potential to transfer knowledge to other domains	Examples
Uncertainty modeling & failure forecasting	Evidential neural networks, rule-based and natural language explanations from a LLM, and gradient boosting trees (or any other supervised classification model)	The uncertainty estimation algorithms are domain-agnostic and can be applied across different sectors. Failure forecasting, however, requires experts to define the critical event to be prevented–such as a blackout in power systems–against which risk levels and early-warning indicators are assessed.	Any situation in which an AI-based assistant may propose actions that fail to adequately mitigate critical risks, such as water supply disruption, cascading failures in telecommunication networks, or violations of minimum quality-of-supply standards.
Real-time interactive & preference-guided optimization	Adapted CMA-ES evolutionary algorithm with multi-objective formalization, sampling injection, and convergence to human-preference rewards.	Although applied in the ATM Sectorization use case, the formulation can easily be adapted to other contexts by replacing the optimization target(s) and output representation to the context under consideration.	Any domain that requires spatial partitioning under workload, coverage, or coordination constraints, such as: urban service and emergency response zoning, telecommunications planning, and logistics & distribution.
RL behavior-shaping architecture	Q-learning agent with linear function approximation, utilizing action shielding, human feedback, and learning from expert demonstrations in a unified feedback architecture.	Although the proposed RL architecture is domain-agnostic, the linear Q-learning agent has been specifically tailored and tuned to ATM conflict resolution. Although the same principle can be applied to other transportation domains focused on conflict resolution involving moving vehicles, linear rewards may limit transferability.	Any transportation domain requiring RL behavior to demonstrate more safe and human-like behavior aligned with operational best practices, such as maritime shipping.
Inverse Reinforcement Learning	Knowledge is represented through a reward function or a set of reward functions that make the demonstrated behavior optimal. The reward function constitutes the most "succinct" representation of the task.	The result of IRL being a reward function can be easily transferred to different environments since, differently from behaviors, it is not affected by dynamic features of the specific environment.	Any domain in which demonstrations of expert agents, e.g., humans, are available. This includes transportation and energy networks. Although the IRL methodology is generic, the applicability of a learned reward function is domain specific.

TABLE 3 - (CONTINUED)

Result	Knowledge representation type	Potential to transfer knowledge	Examples
Preference-based Reinforcement Learning	Knowledge resides in the preferences that are provided by human experts. In the simplest case, this is represented by a rationality model that connects the preference generation process with an underlying reward function.	The presented approaches are by definition general, since they are not tailored for a specific environment. Since they do not require an explicit specification of the reward function, they are more easily applicable to systems where only human feedback is available.	Any domain in which there is the availability of experts (e.g., humans) that are able to provide comparison feedback, which may include direct preferences or indifference. This includes transportation and energy networks.
Agent-as-a-Service (*A3S*)	Service-oriented wrapper that couples an existing AI agent with a simulator/digital twin and exposes endpoints (e.g., *restore*, *simulate*, *get_action_space*).	Domain-agnostic and transferable because *A3S* sits in the *roll-out/usage layer* rather than training: it can wrap different agent types and simulators while preserving the same human-in-the-loop workflow (recommendation + uncertainty/KPIs + optional override + short-horizon "what-if" simulation). Porting mainly requires adapting the simulator interface and KPI definitions.	Critical-infrastructure decision support (rail, power grid, ATM): short-horizon look-ahead with KPI alarms, operator "what-if" checks by fixing the first action and letting the agent continue, and post-hoc counterfactual exploration of recorded trajectories (optionally via GUI extensions such as *TraceRL*).

TABLE 3 - (CONTINUED)

Result	Knowledge representation type	Potential to transfer knowledge	Examples
Method for AI-assisted human learning	A design framework for supporting human learning in human–AI collaboration. It supports the definition of transparency requirements for AI functionalities and the design of collaboration between humans and AI based on a descriptive cognitive task analysis and normative criteria for the design of human tasks. The framework is grounded in psychological concepts, particularly with regard to the cognitive processes underlying human learning and motivation.	The method can be applied independently of knowledge content and can therefore be used for any form of knowledge-intensive work to derive requirements for the design of AI-assisted human learning. In particular, it also supports learning processes relating to implicit experiential knowledge (tacit knowledge), the content of which is difficult to explain explicitly.	Any domain in which human expertise is critical and based on explicit and implicit, experience-based knowledge, and in which corresponding learning processes are to be supported by AI. Examples include healthcare and clinical decision support, industrial process control, cybersecurity operations, emergency and disaster management, financial risk analysis, and military command and control.
Method for human supervision over autonomous AI	An analysis framework supporting the derivation of requirements for the design of a psychologically sound human role in supervision over autonomous AI. It avoids well known problems of supervisory control tasks out-of-the-loop and gives the human an in-the-loop role as director of autonomous AI-agents that are intuitively understandable (i.e. primitives).	The method can be applied independently of task content and can therefore be used for any form of task where the human is expected to supervise autonomous AI agents.	Any area where problems are dynamic and therefore require human expertise in adjusting objectives and problem-solving strategies, and where execution can be delegated to autonomous AI. Examples include autonomous driving and logistics, robotic manufacturing and assembly, autonomous drone operations, and maritime navigation.

TABLE 3 - (CONTINUED)

Result	Knowledge representation type	Potential to transfer knowledge	Examples
Directive-driven Autonomous System Architecture	Graph-based representations of system structure and agent interactions, a token-based negotiation proxy for coordination between agents, directives expressed as primitives over multi-agent reinforcement learning (MARL) agents, and an experience clustering module that organizes previously observed system behaviors.	The architectural framework is task-agnostic and applicable to a wide range of multi-agent reinforcement learning settings in which multiple autonomous agents interact within a shared environment. While the architecture and coordination mechanisms generalize across domains, learned experience models and domain-specific directives or primitives must typically be adapted to the dynamics, constraints, and objectives of the target application.	Applicable to dynamic multi-agent environments where humans and AI co-learn: humans adapt goals and strategies based on expertise, while execution and local decision-making are delegated to autonomous AI agents. Example include autonomous drone fleet coordination, robotic warehouse logistics, traffic signal control in urban mobility networks, and distributed control in industrial production systems.

9. CONCLUSIONS

This deliverable presented the AI4REALNET approach to augmenting human decision-making in the operation of critical network infrastructures. The report adopted a human-centered AI perspective, focusing not on replacing operators with automation, but on enabling effective and trustworthy human-AI collaboration. Across the presented contributions, the key objective has been to translate operational requirements into algorithmic and architectural solutions that support decision-making under uncertainty, enable interactive and co-learning processes, and allow autonomous decision-making where appropriate while maintaining meaningful human supervision and direction.

The following sections summarize the main conclusions of each chapter and give a final conclusion. First, we introduced the overall AI4REALNET framework, centered on optimizing the degree and form of AI-driven decision support such that the human-AI team becomes more effective than either party alone. AI4REALNET structured this perspective into three complementary control modes: (i) (AI-assisted) full-human control, (ii) interactive and human-AI co-learning, and (iii) trustworthy full-AI-based control under human supervision. The main conclusion is that these control modes provide a coherent structure for embedding AI into operational workflows across safety-critical infrastructures. In particular, it established that interpretability, uncertainty communication, and preservation of operator authority are fundamental requirements for trustworthy deployment.

9.1. FROM AUTONOMOUS TO HUMAN-CENTERED AI

An important topic was how we can transition from autonomous to Human-Centered AI. In this chapter, we discussed challenges and opportunities for this transition in decision-making settings. The analysis highlighted that autonomous AI agents, even when technically capable, do not directly translate into effective decision-support tools unless they are adapted to the cognitive, motivational, and alignment requirements of human operators.

The topic identified key challenges related to differing solution styles, inappropriate trust calibration, cognitive workload and situational awareness, and human-AI objective misalignment. A primary conclusion is that human-centered decision support requires AI systems that go beyond generating recommendations and instead support joint decision-making through interaction, evaluation, and adaptive assistance. This motivates the uncertainty-aware decision-support functions (Chapter 4), the explicit representation of trade-offs through multi-objective methods (Chapter 5), and the interactive and co-learning mechanisms presented in Chapter 6.

9.2. AGENT-AS-A-SERVICE

The Agent-As-A-Service (A3S) concept was introduced as a practical integration framework for repurposing autonomous agents for human-centered operation. The main conclusion is that encapsulating AI agents within a modular service architecture enables operators to access not only recommended actions but also the associated uncertainty, risk indicators, and relevant contextual information.
A3S supports the tuning of autonomy levels and enables AI-driven processes to remain supervised and accountable. By exposing uncertainty estimates and enabling inspection of model behavior, the framework provides an operational pathway for turning autonomous decision-making into a supervised process in which humans remain the informed authority. This architecture provides a foundation for integrating the algorithmic contributions described in later chapters.

9.3. RISK AND UNCERTAINTY IN DECISIONS

Risk and uncertainty quantification were identified as central requirements for AI-assisted operations in critical infrastructure. The analysis demonstrates that uncertainty modelling is essential for achieving calibrated trust, particularly in human-in-the-loop decision-making settings where operators must determine whether to rely on, override, or further scrutinize AI-generated recommendations.
The distinction between epistemic uncertainty (reflecting model limitations and out-of-distribution states) and aleatoric uncertainty (reflecting inherent stochasticity) was shown to be critical. The chapter presented methods for forecasting the probability of RL agent failure and for constructing uncertainty intervals for power line loadings using conformal theory. The main conclusion is that explicit uncertainty quantification provides operational self-awareness of AI-based decision systems and strengthens decision support by making AI limitations visible and actionable to the human operator.

9.4. MULTI-OBJECTIVE DECISION-MAKING WITH AI

Multi-objective reinforcement learning was presented as a method for addressing the need to balance multiple competing objectives in operational decision-making. The discussion highlights the need for an explicit representation of trade-offs to ensure alignment with operator intent and broader stakeholder priorities.
By developing approaches for finding convex coverage sets and implementing the AI4REALNET multi-objective package, the project enables the generation of approximately Pareto-optimal solutions that expose the structure of objective conflicts. A key conclusion is that multi-objective methods naturally support mixed-initiative decision-making, as they allow operators to select or steer solutions based on situational priorities rather than relying on fixed objective weights.

9.5. INTERACTIVE AI TO AUGMENT DECISION-MAKING

Interactive AI was introduced as a core paradigm for augmenting decision-making, with particular focus on interactive human-AI learning and human-AI co-learning. The analysis indicates that effective decision support in complex operational contexts requires continuous interaction, enabling both the AI system and the operator to adapt over time.

The chapter emphasized that interactive learning enables AI systems to incorporate explicit and implicit human feedback, particularly in settings with limited data or rare critical events. Co-learning extends this by supporting bidirectional adaptation, in which AI systems learn operator preferences and heuristics while operators learn from AI-generated alternatives, explanations, and scenario exploration. The architectural discussion further concluded that interactive AI requires modular integration of human feedback processing, optimization engines, explanation and visualization components, and state persistence layers. These elements provide the infrastructure for long-term learning, trust calibration, and the development of operator competence.

9.6. AUTONOMOUS AI-DRIVEN DECISION SYSTEMS

Autonomous AI-driven decision systems were examined in which AI agents assume primary responsibility for action execution while remaining under human supervision and direction. The analysis concludes that autonomy in safety-critical infrastructures must be realized as trustworthy autonomy, ensuring that human oversight remains meaningful and that AI behavior stays aligned with human directives.

The chapter's director-mode approach provides a structured mechanism for guiding autonomous agents through high-level directives, ensuring that autonomy remains constrained by operational intent. However, the chapter also concluded that while the director-mode interaction structure generalizes, the underlying AI policies do not directly transfer between environments and must be retrained for new system dynamics and directive sets. This reinforces the importance of maintaining supervision, transparency, and auditing mechanisms for autonomous operation.

9.7. GENERALIZATION OF RESULTS

The extent to which the AI4REALNET solutions generalize beyond the demonstrated use cases was evaluated. The analysis concludes that the architectural principles and algorithmic mechanisms developed in this deliverable generalize well to network-structured problems, as they are formulated around fundamental decision-making challenges such as uncertainty, cascading effects, and multi-objective trade-offs.

In particular, the co-learning support functions generalize well, as they are grounded in general human learning processes rather than domain-specific procedures. Similarly, the A3S architecture provides a domain-agnostic method for integrating autonomous agents into human-centered workflows. However, the chapter concluded that trained AI policies remain environment-dependent, and therefore retraining or adaptation is required when transferring to new operational contexts.

OVERALL PERSPECTIVE

The results presented in this deliverable indicate that effective augmentation of human decision-making in critical network infrastructures cannot be achieved by autonomous optimization alone. Instead, it requires a deliberate shift toward human-centered AI, in which algorithmic capability is embedded into operational workflows through structures that preserve human authority, enable oversight, and support the cognitive processes by which operators build situational awareness, generate hypotheses, evaluate alternatives, and manage risk. Within AI4REALNET, this shift is operationalized through a coherent set of control modes, namely AI-assisted full-human control, interactive and co-learning decision support, and trustworthy full-AI-based control under supervision. These modes are treated not as competing paradigms, but as complementary configurations that can be matched to task criticality, uncertainty, and operational context.

A central insight across these modes is that trustworthiness is not a property of an autonomous policy in isolation, but of the wider socio-technical system in which it is deployed. In practice, this means that AI outputs must be coupled to information that supports human judgment, including interpretable rationales, traceable decision pathways, and most critically explicit representations of risk and uncertainty. The deliverable shows that uncertainty estimation and communication are not optional additions but enabling mechanisms for calibrated trust, because they allow operators to differentiate between recommendations that are robust and those that are fragile due to unfamiliar states, model limitations, or stochastic dynamics. By distinguishing epistemic uncertainty, representing knowledge boundaries, from aleatoric uncertainty, representing irreducible variability, and by linking these to operationally meaningful outputs such as reliability indicators, early warnings, and uncertainty intervals, the presented approach supports the operator's core task, namely deciding when to rely on AI, when to probe further, and when to override or intervene.

From an integration standpoint, the deliverable further demonstrates that technical performance alone is insufficient without a deployment architecture that makes AI systems inspectable, steerable, and accountable. The Agent-As-A-Service (A3S) abstraction exemplifies this by wrapping agents in a human-centered service layer that exposes uncertainty and contextual information, supports what-if exploration, and enables adjustable autonomy levels without collapsing into opaque automation. This modularity is especially important in safety-critical settings, where operational acceptance depends

on the ability to audit behavior, understand failure modes, and maintain meaningful human supervision. In parallel, the emphasis on multi-objective decision-making reinforces the idea that many infrastructure problems are defined not by a single optimum but by structured trade-offs. Explicitly representing these trade-offs, rather than hiding them inside a fixed scalar objective, creates a natural interface for mixed-initiative operation, in which operators can steer solutions based on situational priorities and stakeholder constraints.

Finally, the interactive and co-learning perspective strengthens the deliverable's broader claim that decision support is most valuable when it supports learning and adaptation over time. Human–AI collaboration in operational environments is not static, as operator intent evolves, preferences shift with context, and rare events reveal gaps in both human mental models and AI training distributions. The interactive and co-learning architecture described in the deliverable, therefore, functions as more than a user interface pattern. It is a mechanism for sustained alignment, enabling feedback, preference signals, and operator interventions to be incorporated in a controlled way, while simultaneously allowing operators to learn from AI-generated alternatives, explanations, and scenario exploration. In this sense, the AI system becomes not only an optimizer, but a structured partner in reasoning, exploration, and reflection, capabilities that are particularly relevant in safety-critical domains where competence development and knowledge sharing are long-term organizational objectives.

View into the future. The deliverable outlines a clear research and deployment trajectory. While architectural principles and interaction structures generalize well, learned control policies—particularly in reinforcement learning—remain environment-dependent and typically require retraining when applied to new networks, dynamics, or directive sets. Future research should therefore focus on bridging this gap by developing more transferable abstractions, such as reusable primitives, cross-domain representations, and human-guided adaptation strategies that reduce the need for full retraining.

In parallel, uncertainty-aware operation is expected to evolve from isolated confidence estimates toward integrated assurance frameworks that combine predictive risk models, runtime monitoring, auditability, and interaction-aware explainability. As human-centered AI systems approach deployment, evaluation must also extend beyond algorithmic performance to address calibrated trust, cognitive workload, and long-term learning effects—specifically, whether systems enhance both immediate decision quality and operators' ability to manage rare, high-impact events. In this context, AI4REALNET's integrated approach—linking uncertainty-aware decision support, multi-objective reasoning, interactive learning, and supervised autonomy—offers a solid foundation for the next generation of trustworthy human–AI systems in critical infrastructure.

REFERENCES

Thomas Lautenbacher, Ali Rajaei, Davide Barbieri, Jan Viebahn, and Jochen L. Cremer. Multi-objective reinforcement learning for power grid topology control, 2025. URL https://arxiv.org/abs/2502.00040.

Marco Mussi, Alberto Maria Metelli, Marcello Restelli, Gianvito Losapio, Ricardo J. Bessa, Daniel Boos, Clark Borst, Giulia Leto, Alberto Castagna, Ricardo Chavarriaga, Duarte Dias, Adrian Egli, Andrina Eisenegger, Yassine El Manyari, Anton Fuxjäger, Joaquim Geraldes, Samira Hamouche, Mohamed Hassouna, Bruno Lemetayer, Milad Leyli-Abadi, Roman Liessner, Jonas Lundberg, Antoine Marot, Maroua Meddeb, Viola Schiaffonati, Manuel Schneider, Thilo Stadelmann, Julia Usher, Herke Van Hoof, Jan Viebahn, Toni Waefler, and Giacomo Zanotti. Human-ai interaction in safety-critical network infrastructures. *iScience*, 28, 9 2025. ISSN 25890042. doi: 10.1016/j.isci.2025.113400.

Milad Leyli-abadi, Ricardo J. Bessa, Jan Viebahn, Daniel Boos, Clark Borst, Alberto Castagna, Ricardo Chavarriaga, Mohamed Hassouna, Bruno Lemetayer, Giulia Leto, Antoine Marot, Maroua Meddeb, Manuel Meyer, Viola Schiaffonati, Manuel Schneider, and Toni Waefler. A conceptual framework for ai-based decision systems in critical infrastructures, 2025. URL https://arxiv.org/abs/2504.16133.

Omar Khan, Pascal Poupart, and James Black. Minimal sufficient explanations for factored markov decision processes. In *Proceedings of the international conference on automated planning and scheduling*, volume 19, pages 194–200, 2009.

Herman Yau, Chris Russell, and Simon Hadfield. What did you think would happen? explaining agent behaviour through intended outcomes. *Advances in Neural Information Processing Systems*, 33: 18375–18386, 2020.

Ricardo Bessa, Mouadh Yagoubi, and Milad Leyliabadi. Ai4realnet framework and use cases. Technical Report D1.1, AI4REALNET, 2024.

Tim Miller. Explainable AI is Dead, Long Live Explainable AI! Hypothesis-driven Decision Support using Evaluative AI. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, page 333–342, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701924. doi: 10.1145/3593013.3594001. URL https://doi.org/10.1145/3593013.3594001.

M. R. Endsley. Ironies of artificial intelligence. *Ergonomics*, 66(11):1656–1668, 2023.

Conor F Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M Zintgraf, Richard Dazeley, Fredrik Heintz, et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36:26, 2022.

Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, 2000.

European Union. Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence and amending regulations (ec) no 300/2008, (eu) no 167/2013, (eu) no 168/2013, (eu) 2018/858, (eu) 2018/1139 and (eu) 2019/2144 and directives 2014/90/eu, (eu) 2016/797 and (eu) 2020/1828 (artificial intelligence act), 2024. URL `https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng`. Official Journal of the European Union L 2024/1689, 12 July 2024, pp. 1-144.

J. R. Andrade, C. Rocha, R. Silva, J. P. Viana, R. J. Bessa, C. Gouveia, B. Almeida, R. J. Santos, M. Louro, P. M. Santos, and A. F. Ribeiro. Data-driven anomaly detection and event log profiling of SCADA alarms. *IEEE Access*, pages 73758–73773, 2022.

Jianlong Zhou, Syed Z. Arshad, Simon Luo, and Fang Chen. Effects of uncertainty and cognitive load on user trust in predictive decision making. In Regina Bernhaupt, Girish Dalvi, Anirudha Joshi, Devanuj K. Balkrishan, Jacki O'Neill, and Marco Winckler, editors, *Human-Computer Interaction – INTERACT 2017*, pages 23–39, Cham, 2017. Springer International Publishing.

Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, et al. Forecasting: theory and practice. *International Journal of Forecasting*, 38(3):705–871, 2022.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110:457–506, 2021.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *International Conference on Machine Learning (ICML)*, pages 1613–1622. PMLR, 2015.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1050–1059. PMLR, 2016.

Aslak Djupskas, Signe Riemer-Sorensen, and Alexander Johannes Stasik. Unreliable monte carlo dropout uncertainty estimation. In *Northern Lights Deep Learning Conference (NLDL)*, 2026.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Tony Duan, Anand Avati, Daisy Yi Ding, Sanjay Khan, Andrew Y Ng, and Pranav Basu. Ngboost: Natural gradient boosting for probabilistic prediction. In *International Conference on Machine Learning (ICML)*, pages 2690–2700. PMLR, 2020.

Tao Hong and Shu Fan. Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3):914–938, 2016.

Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of Economic Perspectives*, 15(4): 143–156, 2001.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2nd edition, 2009.

Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(3), 2008.

Anastasios N Angelopoulos, Stephen Bates, et al. Conformal prediction: A gentle introduction. *Foundations and Trends® in Machine Learning*, 16(4):494–591, 2023.

Lars Lindemann, Yiqi Zhao, Xinyi Yu, George J. Pappas, and Jyotirmoy V. Deshmukh. Formal verification and control with conformal prediction: Practical safety guarantees for autonomous systems. *IEEE Control Systems*, 45(6):72–122, December 2025.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Yvet Renkema, Nico Brinkel, and Tarek Alskaif. Conformal prediction for stochastic decision-making of PV power in electricity markets. *Electric Power Systems Research*, 234:110750, 2024.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019.

Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. *Advances in Neural Information Processing Systems*, 34:1660–1672, 2021.

Kegan J Strawn, Nora Ayanian, and Lars Lindemann. Conformal predictive safety filter for RL controllers in dynamic environments. *IEEE Robotics and Automation Letters*, 8(11):7833–7840, 2023.

Felipe Leno Da Silva, Pablo Hernandez-Leal, Bilal Kartal, and Matthew E. Taylor. Uncertainty-aware action advising for deep reinforcement learning agents. In *Thirty-Fourth AAAI Conference on artificial Intelligence (AAAI-20)*, volume 34, pages 5792–5799, New York, USA, February 2020.

Mohamed-Harith Ibrahim, Stephane Lecoeuche, Jacques Boonaert, and Mireille Batton-Hubert. Uncertainty quantification for efficient andrisk-sensitive reinforcement learning. In *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, Mexico City, Mexico, December 2023.

Antoine Marot, Adrian Kelly, Matija Naglic, Vincent Barbesant, Jochen Cremer, Alexandru Stefanov, and Jan Viebahn. Perspectives on future power system control centers for energy transition. *Journal of Modern Power Systems and Clean Energy*, 10(2):328–344, 2022.

Adrien Pavão, Antoine Marot, et al. AI challenge for safe and low carbon power grid operation. *Energy and AI*, 22:100564, December 2025.

Malte Lehna, Mohamed Hassouna, Dmitry Degtyar, Sven Tomforde, and Christoph Scholz. Fault detection for agents on power grid topology optimization: A comprehensive analysis. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML 2024). Workshop Machine Learning for Sustainable Power Systems (ML4SPS)*, Vilnius, Lithuania, September 2024.

Tong Su, Tong Wu, Junbo Zhao, Anna Scaglione, and Le Xie. A review of safe reinforcement learning methods for modern power systems. *Proceedings of the IEEE*, 113(3):213–255, March 2025.

A. Pepiciello and J.L. Domínguez-García. Small-signal stability analysis of uncertain power systems: A comprehensive survey. *Renewable and Sustainable Energy Reviews*, 200:114576, 2024.

Olivier Sprangers, Sebastian Schelter, and Maarten de Rijke. Probabilistic gradient boosting machines for large-scale probabilistic regression. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1531–1541, 2021.

Rina Foygel Barber, Emmanuel J Candes, Aaditya Ramdas, and Ryan J Tibshirani. The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, 10(2):455–482, 2021.

Tiago Bessa. Explaining the optimization of dynamic tariffs for electric vehicles. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2025. Chapters 3 and 4.

Hossam Mossalam, Yannis M. Assael, Diederik M. Roijers, and Shimon Whiteson. Multi-objective deep reinforcement learning, 2016. URL https://arxiv.org/abs/1610.02707.

Diederik Roijers. *Multi-objective decision-theoretic planning*. Phd thesis, Universiteit van Amsterdam, Amsgterdam, Netherlands, 2016. URL https://hdl.handle.net/11245/1.532759.

Florian Felten, Lucas N. Alegre, Ann Nowé, Ana L. C. Bazzan, El Ghazali Talbi, Grégoire Danoy, and Bruno C. da Silva. A toolkit for reliable benchmarking and research in multi-objective reinforcement learning. In *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023.

A. Y. Kolb and D. A. Kolb. The Learning Way: Meta-cognitive Aspects of Experiential Learning. *Simulation & Gaming*, 40(3):297–327, 2009. doi: 10.1177/1046878108325713.

Julian Abich and Eric Sikorski. Taking a constructivist approach to human-AI co-learning design. In *Make it Happen*, 2023.

Roger C. Mayer, James H. Davis, and F. David Schoorman. An integrative model of organizational trust. *The Academy of Management Review*, 20(3):709–734, 1995. ISSN 03637425. URL http://www.jstor.org/stable/258792.

Karel van den Bosch, Tjeerd A.J. Schoonderwoerd, Romy Blankendaal, and Mark A. Neerincx. Six challenges for human-AI co-learning. In *Adaptive Instructional Systems: First International Conference, AIS 2019, Held as Part of the 21st HCI International Conference, HCII 2019, Orlando, FL, USA, July 26–31, 2019, Proceedings*, pages 572–589. Springer-Verlag, 2019. ISBN 978-3-030-22340-3. doi: 10.1007/978-3-030-22341-0_45. URL https://doi.org/10.1007/978-3-030-22341-0_45.

Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, 2008.

Alberto Maria Metelli, Filippo Lazzati, and Marcello Restelli. Towards theoretical understanding of inverse reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pages 24555–24591. PMLR, 2023.

Filippo Lazzati, Mirco Mutti, and Alberto Maria Metelli. Offline inverse reinforcement learning: New solution concepts and provably efficient algorithms. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*. PMLR, 2024a.

Filippo Lazzati, Mirco Mutti, and Alberto Maria Metelli. How does inverse reinforcement learning scale to large state spaces? a provably efficient approach. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b.

Filippo Lazzati and Alberto Maria Metelli. Learning utilities from demonstrations in markov decision processes. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*. PMLR, 2025.

Riccardo Poiani, Gabriele Curti, Alberto Maria Metelli, and Marcello Restelli. Sub-optimal experts mitigate ambiguity in inverse reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136):1–46, 2017.

Ellen R. Novoseller, Yibing Wei, Yanan Sui, Yisong Yue, and Joel W. Burdick. Dueling posterior sampling for preference-based reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Simone Drago, Marco Mussi, and Alberto Maria Metelli. Towards theoretical understanding of sequential decision making with preference feedback. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*. PMLR, 2025.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

W. Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The TAMER framework. In *Proceedings of the Fifth International Conference on Knowledge Capture*, pages 9–16, Redondo Beach California USA, September 2009. ACM. doi: 10.1145/1597735.1597738.

Bilal Piot, Matthieu Geist, and Olivier Pietquin. Boosted bellman residual minimization handling expert demonstrations. In *Machine Learning and Knowledge Discovery in Databases*, pages 549–564. Springer, 2014. doi: 10.1007/978-3-662-44851-9_35.

Francisco S. Melo and M. Isabel Ribeiro. Q-Learning with Linear Function Approximation. In *Learning Theory*, volume 4539, pages 308–322. Springer Berlin Heidelberg, 2007. doi: 10.1007/978-3-540-72927-3_23.

Bettina Könighofer, Florian Lorber, Nils Jansen, and Roderick Bloem. Shield Synthesis for Reinforcement Learning. In *Leveraging Applications of Formal Methods, Verification and Validation: Verification*

Principles: 9th International Symposium on Leveraging Applications of Formal Methods, ISoLA 2020, Rhodes, Greece, October 20–30, 2020, Proceedings, Part I, pages 290–306. Springer-Verlag, 2020. doi: 10.1007/978-3-030-61362-4_16.

Bettina Könighofer, Roderick Bloem, Nils Jansen, Sebastian Junges, and Stefan Pranger. Shields for Safe Reinforcement Learning. *Communications of the ACM*, 68(11):80–90, 2025. ISSN 0001-0782, 1557-7317. doi: 10.1145/3715958.

Ashesh Jain, Brian Wojcik, Thorsten Joachims, and Ashutosh Saxena. Learning trajectory preferences for manipulators via iterative improvement. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, volume 1 of *NIPS'13*, pages 575–583. Curran Associates Inc., 2013.

Ashesh Jain, Shikhar Sharma, Thorsten Joachims, and Ashutosh Saxena. Learning preferences for manipulation tasks from online coactive feedback. *The International Journal of Robotics Research*, 34 (10):1296–1313, 2015. ISSN 0278-3649, 1741-3176. doi: 10.1177/0278364915581193.

Paolo Fiorini. Motion Planning in Dynamic Environments Using Velocity Obstacles. *The International Journal of Robotics Research*, 17:760, 1998. doi: 10.1177/027836499801700706.

Gustavo Adrián Mercado Velasco, Clark Borst, Joost Ellerbroek, M. (René) M. van Paassen, and Max Mulder. The Use of Intent Information in Conflict Detection and Resolution Models Based on Dynamic Velocity Obstacles. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):2297–2302, 2015. ISSN 1558-0016. doi: 10.1109/TITS.2014.2376031.

Robert Regtuit, Clark Borst, and Erik-Jan Van Kampen. Building Strategic Conformal Automation for Air Traffic Control Using Machine Learning. In *2018 AIAA Information Systems-AIAA Infotech @ Aerospace*. American Institute of Aeronautics and Astronautics, 2018. doi: 10.2514/6.2018-0074.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John Agapiou, Joel Leibo, and Audrunas Gruslys. Deep Q-learning From Demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32 of *AAAI'18/IAAI'18/EAAI'18*, pages 3223–3230. AAAI Press, 2018. doi: 10.1609/aaai.v32i1.11757.

Daniel Delahaye, Marc Schoenauer, and Jean-Marc Alliot. Airspace Sectoring by Evolutionary Computation. In *1998 IEEE International Conference on Evolutionary Computation Proceedings*, pages 218–223, 06 1998. ISBN 0-7803-4869-9. doi: 10.1109/ICEC.1998.699504.

Michael Schultz, Ingrid Gerdes, Thomas Standfuß, and Annette Temme. Future Airspace Design by Dynamic Sectorization. In *Air Traffic Management and Systems III*, pages 19–34, Singapore, 2019.

Springer Singapore. doi: 10.1007/978-981-13-7086-1_2. URL https://elib.dlr.de/127136/. Lecture Notes in Electrical Engineering.

Nikolaus Hansen. The CMA Evolution Strategy: A Comparing Review. *Towards a New Evolutionary Computation*, pages 75–102, 2006. doi: 10.1007/3-540-32494-1_4.

Min Xue. Airspace Sector Redesign Based on Voronoi Diagrams. *Journal of Aerospace Computing, Information, and Communication*, 6(12):624–634, 2009. doi: 10.2514/1.41159.

Biswajit Paria, Kirthevasan Kandasamy, and Barnabas Poczos. A Flexible Multi-Objective Bayesian Optimization Approach using Random Scalarizations. 05 2018. https://arxiv.org/abs/1805.12168.

Riccardo Patriarca, Jörg Leonhardt, and Antonio Licu. Unearthing Weak Signals for safer and more efficient socio-technical systems. The Structured Exploration of Complex Adaptations (SECA) method. Information, EUROCONTROL, 2022.

Stuart K. Card, Thomas P. Moran, and Allen Newell. *The psychology of human-computer interaction*. Erlbaum, Mahwah, NJ, repr edition, 2008. ISBN 978-0-89859-243-6 978-0-89859-859-9.

Erik Hollnagel. *FRAM: The Functional Resonance Analysis Method: Modelling Complex Socio-technical Systems*. CRC Press, 1 edition, November 2017. ISBN 978-1-315-25507-1. doi: 10.1201/9781315255071. URL https://www.taylorfrancis.com/books/9781351935968.

Samira Hamouche, Nerissa Dettling, Julia Usher, Manuel Renold, and Toni Waefler. A methodical approach to AI-supported human learning in complex task environments. *The Applied Human Factors and Ergonomics (AFHE) Open Access (in press)*.

M. R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

Lisanne Bainbridge. Ironies of automation. *Automatica*, 19(6):775–779, 1983. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(83)90046-8.

Zana Buçinca, Siddharth Swaroop, Amanda E. Paluch, Finale Doshi-Velez, and Krzysztof Z. Gajos. Contrastive Explanations That Anticipate Human Misconceptions Can Improve Human Decision-Making Skills. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400713941. doi: 10.1145/3706598.3713229. URL https://doi.org/10.1145/3706598.3713229.

Mattias Wahde and Marco Virgolin. The five Is: Key principles for interpretable and safe conversational AI. In *2021 The 4th International Conference on Computational Intelligence and Intelligent Systems*,

pages 50–54, Tokyo Japan, November 2021. ACM. ISBN 978-1-4503-8593-0. doi: 10.1145/3507623.3507632. URL https://dl.acm.org/doi/10.1145/3507623.3507632.

Neville A. Stanton, Paul M. Salmon, Laura A. Rafferty, Guy H. Walker, Chris Baber, and Daniel P. Jenkins. *Human Factors Methods: A Practical Guide for Engineering and Design*. CRC Press, 1 edition, September 2017. ISBN 978-1-315-58739-4. doi: 10.1201/9781315587394. URL https://www.taylorfrancis.com/books/9781317120162.

Nerissa Dettling, Samira Hamouche, and Toni Waefler. Avoiding black box problems by assigning an active role to humans in the control of autonomous AI: A methodological approach. In *press*, 2026.