Human Factors in Robots, Drones and Unmanned Systems, Vol. 218, 2026, 84–94
https://doi.org/10.54941/ahfe1007681

Rule-Based Interpretable AI for Concurrent Collision Detection in Industrial Robot Manipulators

Hesam Jafarian[1], Marzieh Zare[2], Ayanoglu Uras[1], Juha Kalliovaara[1], and Jarkko Paavola[1]

[1]Turku University of Applied Sciences, Dept. ICT, Turku, Finland
[2]Tampere University, Dept. Computer Science, Tampere, Finland

ABSTRACT

Safe human-robot collaboration in industrial environments demands collision detection systems that are both computationally efficient and interpretable. Existing approaches based on geometric modeling, bounding volume hierarchies, or physics engines impose significant computational overhead that limits real-time performance, particularly for high-degree-of-freedom manipulators operating in complex workspaces. This reliance on expensive computation also perpetuates manual path teaching and validation practices that reduce deployment efficiency and increase operator workload. This paper proposes a rule-based artificial intelligence framework that replaces iterative geometric calculations with a learned, symbolic representation of the collision function. Joint configurations are sampled across the robot's operational space within a simulation environment and labeled according to their collision state. An ensemble learning method is trained on this dataset to approximate the collision boundary directly from joint space, bypassing the need for explicit kinematic or geometric modeling at query time. The central contribution of this work is the systematic extraction of decision rules from the trained model. These rules are compiled into a structured knowledge base, in which an inference engine queries to evaluate collision states in constant time independent of scene complexity or robot configuration. This architecture offers two critical advantages over classical methods: a substantial reduction in computational cost during operation, and a transparent, inspectable representation of system behavior that supports validation and human oversight. The proposed method is evaluated on a six-degree-of-freedom industrial manipulator in a controlled simulation environment. Results demonstrate a significant speed-up in collision checking relative to physics-based engine calculations, achieving real-time performance suitable for integration into motion planning pipelines. Prediction accuracy remains within acceptable bounds for practical deployment, and the rule-based structure allows collision logic to be audited without specialized simulation tools. From a human factor's perspective, the approach reduces dependence on manual robot teaching and path validation tasks that remain labour-intensive and error-prone in current industrial practice. By lowering the computational and operational barriers to collision-safe motion planning, the proposed system supports safer, more efficient human-robot collaboration in manufacturing environments.

Keywords: Human-robot collaboration, Collision detection, Industrial robot manipulator, Explainable AI, Rule-based systems, Real-time motion planning

Received May 22, 2026; Revised June 4, 2026; Accepted June 18, 2026; Available online July 20, 2026

INTRODUCTION

Industrial robot manipulators are widely deployed in shared workspaces alongside human operators, making collision-safe motion planning essential for both productivity and worker safety (Russell and Norvig, 2002). As manufacturing systems become more collaborative (Villani et al., 2018; Matheson et al., 2019), the demand for autonomous, collision-free operation and systems grows. In current industrial practice, however, this autonomy is largely achieved through human effort: For instance in a car factory, engineers manually teach and validate robot paths on the teach pendant which is a labor-intensive, error-prone process that can consume days to weeks per cell and must be repeated whenever the workspace changes (Latombe, 2012). This workload is not incidental. It is the cost operators pay for trusting an automation system whose internal logic they cannot inspect (Lee and See, 2004; Parasuraman and Riley, 1997). The bottleneck lies not only in computational expense but in opacity. The core computation comes with physics-based and geometric computation methods that buries collision logic inside bounding-volume hierarchies, mesh intersection tests, and forward-kinematic chains (Ericson, 2004; Corke, 2017) which are the structures that are mathematically rigorous and practically opaque to the engineers who must validate them. Hence, learning-based alternative solutions (García et al., 2002; Pan and Manocha, 2015; Das and Yip, 2020) can improve performance but, by replacing geometrical computation with neural-network weights, deepen rather than resolve the opacity problem (Rudin, 2019). The result is a deployment workflow in which operators are asked to trust systems they cannot examine, and to compensate for that gap through extensive manual checking that is a misallocation of human attention and human-factors research has documented across automation contexts (Hoff and Bashir, 2015).

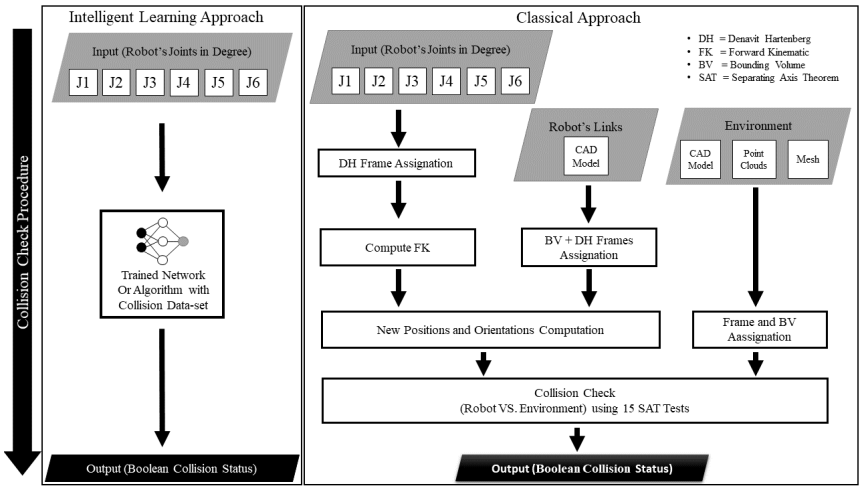

Figure 1: Intelligent collision estimation versus classical geometrical computation.

To our knowledge, no prior work on collision detection combines constant-time inference with an explicitly inspectable representation of collision logic that is the combination that industrial deployment, and the operators within it, require. This paper addresses that gap with a rule-based approach that learns the collision function from joint-space sampling and compiles it into an explicit, human-readable knowledge base. Contributions are threefold: (1) an interpretable rule base extracted from an ensemble classifier, allowing safety engineers and operators to inspect collision logic directly without specialized simulation tools or expertise in geometric kinematics; (2) a simplified collision-detection pipeline that replaces iterative geometric computation with a learned symbolic representation derived from joint-space sampling, decoupling collision queries from scene geometry at runtime as shown in (Figure 1); and (3) a constant-time inference, demonstrated on a 6-DoF industrial manipulator with an empirical ~23,000× speedup over physics-engine reference implementation, making the approach practical for integration into existing motion-planning pipelines while reducing the operator's manual-validation burden in deployment phase. By lowering both the computational and the operational barriers to collision-safe motion planning, the approach supports safe and efficient human-robot collaboration in manufacturing environments.

RELATED WORK

Prior work on collision detection for robot manipulators divides along two axes: how collisions are computed, and whether the resulting logic is inspectable by humans. We review both literatures, then turn to the rule-extraction and human-factors work that motivates our approach.

Geometric and Physics-Based Methods

The dominant approach in industrial collision detection (CD) uses explicit geometric representations of robot links and environment objects, paired with intersection tests at query time. Foundational treatments are given by Ericson (2004) and Corke (2017). To control the cost of pairwise tests, the field has developed a layered hierarchy of acceleration structures such as hierarchical sphere approximations and octree decompositions. Modern physics engines add broad-phase culling (sweep-and-prune, parallel spatial subdivision) on top of these structures (Nvidia, 2013), and downstream applications typically treat CD as an opaque subroutine called from inside the planner (Figure 2).

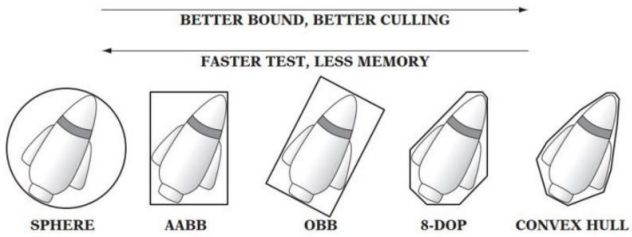

Figure 2: Bounding volume hierarchy and geometrical computation cost intensity.

The fundamental scaling problem persists across this lineage: query cost grows with both the number of collision shapes and the manipulator's degrees of freedom, and each query traverses a chain of forward-kinematic transformations and per-pair tests. Beyond runtime cost, these representations are practically opaque to the engineers responsible for validating robot behavior. Collision logic is distributed across geometry files, transformation matrices, and engine internals, none of which can be meaningfully inspected without specialized tooling.

Learning-Based Collision Detection

A smaller body of work has applied machine learning to bypass geometric computation at query time. García et al. (2002) trained a neural network to predict collision between two boxes, demonstrating the feasibility of learned CD but limited to simple geometries. Pan and Manocha (2015) used support vector machines to approximate collision-free regions of the configuration space for motion planning, accelerated by GPU computation. Das and Yip (2020) proposed a proxy collision-learning approach that updates the learned model as the environment changes, supporting dynamic scenes. Our own prior work (Jafarian, 2021) applied Natural Gradient Boosting to obtain probabilistic collision predictions on a 6-DoF manipulator, and the present paper extends that line by replacing the boosted ensemble with an extracted rule base.

These methods successfully shift collision detection from runtime geometric computation to inference over a learned model, achieving substantial speedups. However, they share a structural limitation: the learned function of neural-network weights, support vectors, or boosted-tree ensembles is itself opaque. From the operator's perspective, one black box has been replaced with another (Rudin, 2019). The validation burden does not decrease; it shifts from inspecting geometry to trusting a non-inspectable model.

Rule Extraction From Ensemble Models

A parallel literature in machine learning has developed techniques for extracting compact, human-readable decision rules from ensemble classifiers. Friedman and Popescu (2008) developed predictive learning via rule ensembles, deriving compact rule sets from gradient-boosted trees while preserving most of the predictive performance of the underlying ensemble.

Random forests (Breiman, 1996) provide a natural substrate for such extraction, since each tree in the ensemble is itself a sequence of interpretable splits. These techniques have been applied to domains where prediction must be both fast and explainable — medical decision support, credit scoring, fault diagnosis — but to our knowledge, not yet for industrial robot collision detection.

Interpretability and Human Oversight in Industrial Automation

Human-factors research has documented for decades that opacity in automation is not a peripheral concern but a primary determinant of how operators use, misuse, or abandon automated systems (Parasuraman and Riley, 1997). Trust calibration — the operator's ability to judge when automation should be relied upon — depends substantially on whether the system's reasoning can be examined (Lee and See, 2004; Hoff and Bashir, 2015). For safety-critical industrial automation, this argues against post-hoc explanations of black-box models and in favor of models that are interpretable by construction (Rudin, 2019). Recent surveys of human–robot collaboration in manufacturing similarly emphasize that operator-facing transparency is essential for safe and efficient shared workspaces (Villani et al., 2018; Matheson et al., 2019).

METHODOLOGY

Overview

The proposed system is organized as an offline training pipeline followed by an online inference pipeline. The offline stage generates labeled joint-space data, trains an ensemble classifier on this data, and extracts a compact rule base from the trained ensemble. The online stage uses only the extracted rule base — the ensemble itself is discarded after extraction. Figure 3 shows the full pipeline with the offline/online boundary marked explicitly.

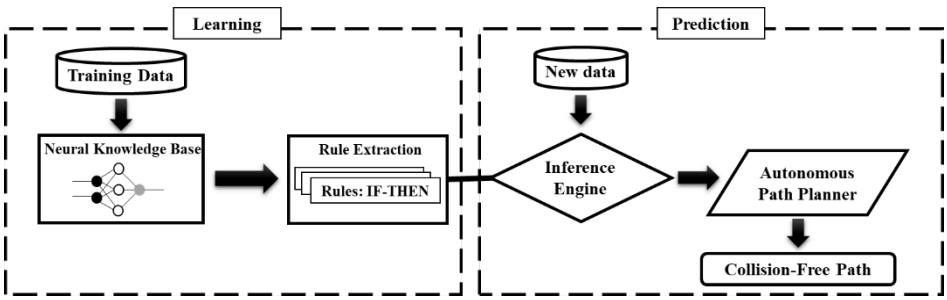

Figure 3: Training and deployment pipeline.

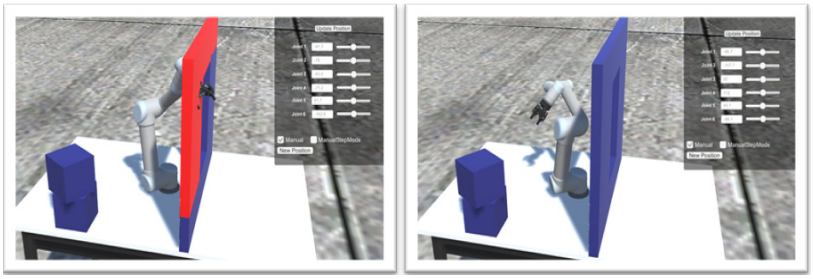

Figure 4: UR5 robot states In-collision (left) and No-Collision (right) in an environment.

Data Generation

We reformulate collision detection as function approximation over the joint space given the manipulator's six-dimensional joint configuration q, predict a Boolean collision label, with the prediction made by a compact rule base extracted from a trained ensemble rather than by the ensemble itself. We use the Universal Robots UR5, a six-degree-of-freedom collaborative manipulator, as the case study; the methodology is in principle applicable to other industrial manipulators with comparable kinematic structure, though broader empirical validation is left to future work. The method assumes a static cell similar to a real industrial environment during a deployment session and a known robot collision geometry, both standard for fixed industrial work cells. Joint configurations are sampled uniformly at random within the manipulator's operational subspace Q, defined by per-joint limits chosen to exclude self-collision and ground impact in the target application. The resulting ranges are given in Table 1.

Table 1: Operational joint ranges for the UR5 case study, in degrees.

	J1	J2	J3	J4	J5	J6	Collision label
Min	−180	−180	−150	−180	−110	−180	0 = free
Max	180	−2	130	180	110	180	1 = collision

Each sampled configuration q is loaded into the physics simulator, the manipulator is posed accordingly, and the ground-truth collision label $f_{geom}(q)$ is recorded from the simulator's contact-event output. (Figure 4) shows representative in-collision and collision-free configurations of the UR5 in the simulated work cell. The robot's collision geometry uses mesh colliders derived from the manufacturer's CAD model (Figure 5); environment objects use axis-aligned bounding boxes. A self-collision adjacency table, derived from the URDF kinematic chain (full table omitted for space), excludes adjacent-link pairs from the test, leaving only pairs that can physically collide. Sampling continues until the dataset reaches 300,000 labeled configurations, partitioned 50/50 into training and held-out test sets. The resulting class distribution is approximately balanced after restricting Q to the operational ranges in Table 1.

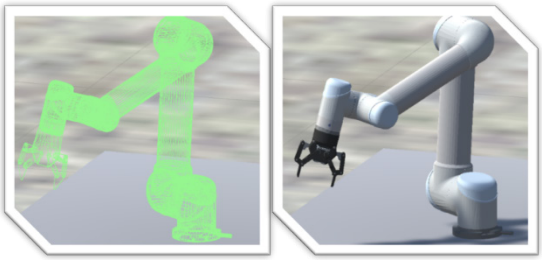

Figure 5: UR5 robot collision geometries (left) versus UR5 robot manipulator (right).

Ensemble Training and Rule Extraction

We train a random-forest classifier on the labeled training set, with joint angles (q_1, …, q_6) as the only input features and the collision label as the target. Random forests are chosen for two reasons: each constituent tree is itself a sequence of interpretable splits, and the ensemble provides a natural substrate for rule extraction (Breiman, 1996). Hyperparameters are tuned by 5-fold cross-validation on the training partition; the held-out test set is used only for the final evaluation reported in the Results section.

From the trained forest we extract a compact rule base using the rule-ensemble framework of Friedman and Popescu (2008), chosen for its sparse L1-regularized rule selection, which yields a compact rule set amenable to manual inspection; alternative approaches such as decision-tree distillation could substitute here without fundamentally changing the pipeline. Each rule r_k is a conjunction of axis-aligned conditions on joint variables — for example, "if $q_2 < -90°$ and $q_3 > 45°$ then in-collision" — corresponding to a root-to-leaf path in one of the forest's trees. The full set of candidate rules across all trees is then reduced: rules with low predictive contribution are pruned, near-duplicate rules are merged, and the surviving rules are assigned weights that minimize an L1-regularized loss on the training data. The output is a small set of weighted rules — 23 in our experiments — that together approximate the ensemble's decision function.

The rule base is stored as a plain-text knowledge file: one rule per line, each rule expressed as a human-readable conjunction of joint-angle conditions. No floating-point weights or learned embeddings appear at query time; the rules themselves are the deployed artefact.

RESULTS

We evaluate the proposed system along three dimensions — query-time speed (R1), classification accuracy (R2), and inspectability (R3) — using the 6-DoF UR5 case study with the 300,000-sample dataset partitioned 50/50 into

training and held-out test sets. All measurements use a single CPU core, with both the physics-engine baseline and the rule-based inference single-threaded.

Speed and Accuracy (R1, R2)

Generating the labeled dataset took approximately 8 hours of wall-clock time. Our PhysX-based reference implementation, configured with mesh colliders for the robot and axis-aligned bounding boxes for environment objects, achieves approximately 625 collision evaluations per minute on a single CPU core — representative of unoptimized research deployments rather than production-tuned planners. Training the random-forest classifier on the labeled training partition took approximately 2 minutes, with rule extraction adding a further few seconds. The extracted rule base evaluates the 150,000-sample held-out test set in 0.61 seconds — approximately 4 microseconds per query, or 14 million queries per minute — a relative speedup of approximately 23,000× under matched conditions. The absolute magnitude reflects the unoptimized baseline; the load-bearing claim is the relative scaling — query cost decoupled from scene complexity, environment objects, or manipulator degrees of freedom. This is consistent with order-of-magnitude gains reported for recent neural collision-detection methods (Joho et al., 2024), while replacing their opaque network with an inspectable representation.

Figure 6 shows the ROC curve for the random-forest ensemble with the rule-base operating points superimposed as red markers. The random forest achieves an area under the ROC curve close to 1.0; the rule base reproduces this behavior with only a modest drop in true-positive rate at matched false-positive rate, corresponding to precision values around 0.67 in the high-recall region. The precision-recall behavior is consistent. The rule base trades a modest amount of recall for a substantial gain in inference speed and a complete gain in inspectability. The precision drop implies that some collision-free configurations near the decision boundary are conservatively flagged as in-collision, costing the planner a fraction of its sampling effort; this cost is acceptable when CD queries are cheap enough to over-sample, which is precisely the regime our method establishes (Liu et al., 2024).

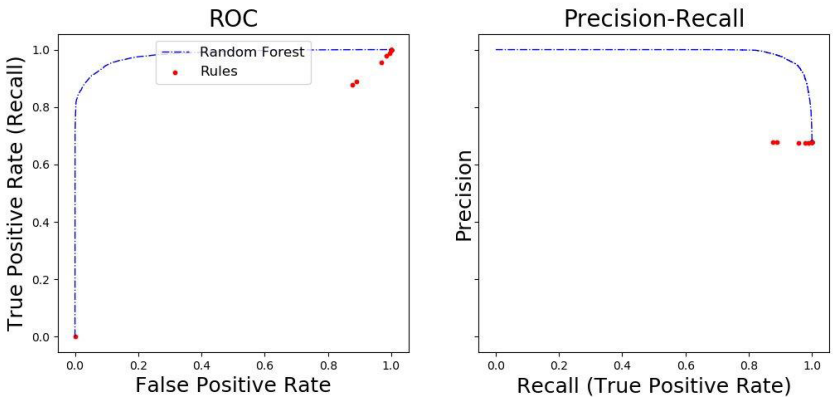

Figure 6: ROC and Precision–Recall curves for the random-forest ensemble (line) with the extracted rule base (red points) on the held-out test set.

Inspectability (R3)

The extracted rule base contains 23 weighted rules, each a conjunction of axis-aligned conditions on the six joint variables. Highly valued representative rules drawn from the extracted set, with the highest-weight rules being the natural starting point for operator inspection, take the form:

Rule A: IF $q_2 < -90°$ AND $q_3 > 45°$ THEN in-collision (weight 0.83)

Rule B: IF $q_1 > 120°$ AND $q_2 < -150°$ AND $q_5 > 60°$ THEN in-collision (weight 0.71)

A safety engineer examining Rule A can verify it against the workspace geometry — the range of shoulder and elbow angles brings the upper arm into the rear obstacle — without launching a simulator, opening a CAD file, or interpreting learned weights. The full rule base fits on a single page of plain text, can be version-controlled, diffed across cell reconfigurations, and reviewed in the same workflow as any other deployment artefact. This is the form of interpretability-by-construction advocated for safety-critical industrial applications in recent XAI surveys (Vassiliades et al., 2025) and aligned with the operator-trust requirements of human-robot collaboration in modern manufacturing (Pietrantoni et al., 2024; Keshvarparast et al., 2023).

DISCUSSION AND CONCLUSION

We have presented a rule-based approach to collision detection for industrial robot manipulators that combines constant-time query performance with an inspectable representation of collision logic. By training an ensemble classifier on simulator-labeled joint configurations and extracting a compact rule base, the deployed system replaces physics-based collision-detection cost with the lookup cost of a small knowledge file, achieving an approximate 23,000× speedup on a 6-DoF UR5 manipulator with classification accuracy adequate for motion-planning use. The rule base inherits the random forest's blind spots: configurations near the decision boundary, where the ensemble is uncertain, are also where rule-base disagreements concentrate, with false negatives clustering around thin geometric clearances. A motion planner using the rule base should therefore treat its output as a fast first-pass filter and confirm any planned trajectory with a single physics-engine call before execution — a hybrid pattern increasingly common in safety-critical AI deployment (Pereira and Thomas, 2024). The approach also assumes a fixed cell configuration; environment changes require regeneration of the rule base. For fixed work cells in shared workspaces this matches operational reality, since cell changes already trigger formal re-validation under existing safety standards, while dynamic environments would require either periodic re-training or a hybrid scheme combining a static rule base for fixed geometry with a fast online check for moving objects.

Future work includes fuzzy and probabilistic rule extraction near the decision boundary, hybrid static-plus-dynamic schemes for non-static cells, and a user study with industrial safety engineers to directly test the operator-oversight claims that motivate this work.

REFERENCES

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123–140.

Corke, P. (2017). *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (2nd ed.). Springer.

Das, N., and Yip, M. (2020). Learning-based proxy collision detection for robot motion planning applications. *IEEE Transactions on Robotics*, 36(4), 1096–1114.

Ericson, C. (2004). *Real-Time Collision Detection*. CRC Press.

Friedman, J. H., and Popescu, B. E. (2008). Predictive learning via rule ensembles. *Annals of Applied Statistics*, 2(3), 916–954.

García, I., Martín-Guerrero, J. D., Soria-Olivas, E., Martínez, R. J., Rueda, S., and Magdalena, R. (2002). A neural network approach for real-time collision detection. In *IEEE International Conference on Systems, Man and Cybernetics* (Vol. 5). IEEE.

Hoff, K. A., and Bashir, M. (2015). Trust in automation: Integrating empirical evidence on factors that influence trust. *Human Factors*, 57(3), 407–434.

Jafarian, H. (2021). Natural gradient boosting collision detection in robot manipulators. *Robotics: Science and Systems (RSS) 2021 Workshop*.

Joho, D., Schwinn, J., and Safronov, K. (2024). Neural implicit swept volume models for fast collision detection. In *2024 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 15402–15408). IEEE.

Keshvarparast, A., Battini, D., Battaia, O., and Pirayesh, A. (2023). Collaborative robots in manufacturing and assembly systems: literature review and future research agenda. *Journal of Intelligent Manufacturing*, 1–54.

Latombe, J.-C. (2012). *Robot Motion Planning*. Springer Science and Business Media.

Lee, J. D., and See, K. A. (2004). Trust in automation: Designing for appropriate reliance. *Human Factors*, 46(1), 50–80.

Liu, Q., Cao, F., and Zhao, W. (2024). Review on motion planning of robotic manipulator in dynamic environments. *Journal of Sensors*, 2024, Article 5969512.

Matheson, E., Minto, R., Zampieri, E. G. G., Faccio, M., and Rosati, G. (2019). Human–robot collaboration in manufacturing applications: A review. *Robotics*, 8(4), 100.

Nvidia. (2013). *PhysX Geometry Classes Guide*. Retrieved from https://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/guide/Manual/Geometry.html

Pan, J., and Manocha, D. (2015). Efficient configuration space construction and optimization for motion planning. *Engineering*, 1(1), 46–57.

Parasuraman, R., and Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. *Human Factors*, 39(2), 230–253.

Pereira, A., and Thomas, C. (2024). Artificial intelligence for safety-critical systems in industrial and transportation domains: A survey. *ACM Computing Surveys*, 56(7), Article 175.

Pietrantoni, L., Favilla, M., Fraboni, F., Mazzoni, E., Morandini, S., Benvenuti, M., and De Angelis, M. (2024). Integrating collaborative robots in manufacturing, logistics, and agriculture: Expert perspectives on technical, safety, and human factors. *Frontiers in Robotics and AI*, 11, 1342130.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5), 206–215.

Russell, S., and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Vassiliades, A., Bassiliades, N., and Patkos, T. (2025). Unveiling the footprints of eXplainable AI in Industry 4.0/5.0: A systematic review and bibliometric exploration. *Journal of Industrial Information Integration*, 45, 100835.

Villani, V., Pini, F., Leali, F., and Secchi, C. (2018). Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. *Mechatronics*, 55, 248–266.