FACTORYNET: A LARGE-SCALE DATASET TOWARD INDUSTRIAL TIME-SERIES FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

anonymous.4open.science/r/FactoryNet-E0C5
huggingface.co/datasets/FactoryNet4/factorynet

ABSTRACT

Industrial machines are actuated systems, yet most open time-series corpora record sensor outcomes alone; the few that do log commanded signals expose them in vendor-specific forms that differ from machine to machine, so *commanded intent* and *measured response* cannot be compared across embodiments. We introduce **FactoryNet**, a pretraining corpus for industrial time-series: 113M datapoints across 57k end-to-end task executions (15k real, 42k synthetic) on seven embodiments and four tasks, with 27 annotated anomaly types alongside healthy baselines and counterfactual pairs. Every signal in every source is mapped into a single control-theoretic schema, *Setpoint, Effort, Feedback, Context* (S-E-F-C), which expresses any actuated system in a common representational frame and makes commanded-versus-realized dynamics readable as explicit prediction residuals. Using the corpus, we find that a task-phase model trained on simulation, with no real phase labels, recovers roughly two thirds of the real-data ceiling on both robots, and that transfer holds within a robot family but rarely across manufacturers. Fault signatures behave differently: an uncalibrated simulator gives no detectable fault signal on either robot, and aligning the feature distributions does not repair it. What separation there is comes from *Effort*: motor current distinguishes faults where measured tool motion is close to chance, and calibrating the simulator to reproduce that one channel recovers a payload fault on the real robot. We read this as simulation capturing the stable structure of a task while fault behaviour stays diverse and fault-type specific. We release FactoryNet as a growing, multi-embodiment substrate for industrial foundation models.

1 INTRODUCTION

The manufacturing sector accounts for approximately 15% of global GDP, relying heavily on the continuous operation of complex actuated machinery World Bank (2026). While predictive maintenance and process optimization present significant opportunities for machine learning, industrial AI remains largely confined to single-machine, bespoke deployments. Foundation models have transformed vision and language by pretraining on large, structurally coherent corpora Brown et al. (2020); Dosovitskiy et al. (2021), yet no analogous substrate exists for industrial time-series. The gap is not merely volume: existing anomaly detection and forecasting datasets Brockmann et al. (2024); Leporowski et al. (2021) record sensor outcomes without separating *commanded intent* from *measured response*. For actuated systems, learning transferable dynamics requires observing the full control loop from target trajectory through actuation effort to the resulting physical state. Partial solutions exist for specific machines (Section 2), but no unified open dataset provides this explicit decomposition across multiple embodiments.[1]

Every signal is mapped into the Setpoint-Effort-Feedback-Context (S-E-F-C) schema: a machine-agnostic signal taxonomy we propose, inspired by the function-oriented reference designation principles of IEC 81346, so that signals from different vendors and machine types fall into the same four roles. Setpoint and Context fix commanded intent and boundary conditions, while Effort and

[1] An *embodiment* is a distinct machine platform, UR3 and UR3e counted as one: UR3/UR3e, UR5, UR10, UR30, KUKA KR10, the voraus-AD cobot, the UMich CNC machining centre.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

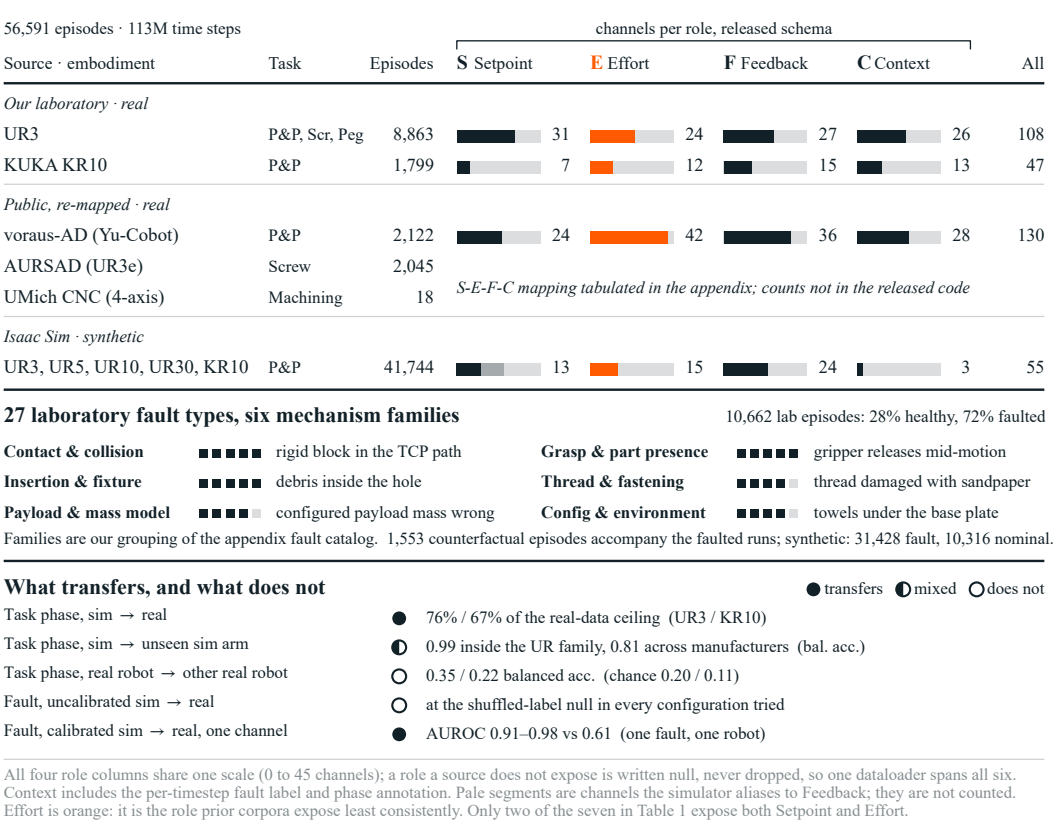

Figure 1: **FactoryNet at a glance.** 56,591 task executions and 113M logged time steps across six sources and seven embodiments, every signal mapped into one control-theoretic schema. *Top:* channels each source exposes per role, on one shared scale; a role a source does not expose is written null rather than dropped, so every source loads through the same dataloader. Effort, highlighted, is the role prior corpora expose least consistently: only two of the seven in Table 1 expose both Setpoint and Effort. *Middle:* the 27 laboratory fault types, grouped by mechanism. *Bottom:* what the transfer study (Section 4.3) finds carries between domains and what does not.

Feedback make *commanded versus realized* dynamics comparable as explicit prediction residuals rather than opaque reconstruction scores. Paired real and simulated episodes under the same schema let sim-to-real mismatch be read as forward-model error under matched inputs.

The schema makes a split visible. Simulation transfers the *task structure* of a cycle, meaning which stage is running and in what order, but not its *fault physics*. S-E-F-C says why: the fault signal sits in *Effort*, the actuation channel, and Effort is precisely what a kinematic simulator reproduces worst and what open corpora expose least consistently. Simulation buys commanded motion, not contact physics. That is a claim about sim-to-real transfer in actuated systems rather than about robot arms, and it is measurable only because one schema puts different machines in a common frame.

2 Related Work

2.1 Industrial Fault-Detection Datasets

The availability of open-source data in the manufacturing domain lags significantly behind other modalities. Most existing datasets focus on single-machine, run-to-failure scenarios or specific rotating machinery components. Canonical examples include the NASA C-MAPSS turbofan degradation benchmark Saxena et al. (2008) and recent robotic datasets: voraus-AD Brockmann et al. (2024) provides 2,122 episodes of industrial robot recordings and AURSAD Leporowski et al. (2021) offers 2,045 episodes. However, these datasets are bounded in scale and lack an explicit control-loop

Table 1: Comparison of open-source industrial time-series datasets. FactoryNet is, to our knowledge, the only multi-machine corpus that requires a strict control-loop structure decoupling intended Setpoint from applied Effort. The Eps. column is not a like-for-like count: ^{r} continuous run-to-failure or steady-state recordings, counted as files, for which any episode boundary is a segmentation choice; ^{c} machining cuts (six cutters $\times$ 315 cuts); ^{m} measurement runs (32 bearings $\times$ 4 operating conditions $\times$ 20 repeats); all others, and FactoryNet, are end-to-end task executions. The Setpoint and Effort columns, not the count, are what distinguishes the corpora.

Dataset	Year	Machine Type	Signals	Eps.	Setpoint	Effort
CWRU Loparo (2000)	2000	Bearings	Vibration	161^{r}	No	No
PHM 2010 PHM Society (2010)	2010	CNC mill	Force, vib.	1,890^{c}	Partial	Yes
Paderborn Lessmeier et al. (2016)	2016	Bearings	Vib., current	2,560^{m}	RPM only	Partial
MAFAULDA Marins et al. (2016)	2016	Rotating mach.	Vib., audio	1,951^{r}	No	No
XJTU-SY Lei et al. (2019)	2019	Bearings	Vibration	15^{r}	No	No
AURSAD Leporowski et al. (2021)	2021	UR3e robot	Joint sig.	2,045	Yes	Yes
voraus-AD Brockmann et al. (2024)	2024	Cobot	130 channels	2,122	Yes	Yes
FactoryNet (Ours)	**2026**	**Multi-machine**	**Multimodal**	**56,591**	**Required**	**Required**

structure mapping commands to outcomes. By unifying these sources alongside novel laboratory data under a standard control-theoretic decomposition, FactoryNet provides the largest open-source, fault-injected industrial robot dataset to date (see Table 1 for a detailed comparison with existing datasets).

Beyond single-component datasets, broader industrial anomaly detection benchmarks such as SKAB Katser & Kozitsin (2020), MetroPT Veloso et al. (2022), WADI Ahmed et al. (2017) and the Tennessee-Eastman process Downs & Vogel (1993) provide system-level multivariate time-series. Similarly, general-purpose benchmarks such as TSB-AD Liu & Paparrizos (2024) were built to evaluate methods like Anomaly-Transformer Xu et al. (2022), and toolkits such as TODS Lai et al. (2021) to automate their application. However, these datasets record holistic system states without the explicit commanded-versus-measured structural decomposition necessary for learning robotic dynamics.

2.2 Foundation Models for Time-Series and Robotics

Current foundation model research diverges into two distinct tracks. In robotics, Open X-Embodiment O'Neill et al. (2024) pools data across diverse kinematics, and DROID Khazatsky et al. (2024) across diverse scenes and tasks on a single platform, to train generalizable behavioral policies. In the structured time-series domain, models such as Chronos Ansari et al. (2024), TimesFM Das et al. (2024), Moirai Woo et al. (2024), Timer Liu et al. (2024) and Lag-Llama Rasul et al. (2023) leverage pretraining corpora for zero-shot forecasting, while MOMENT Goswami et al. (2024) targets broader analysis and adaptation and TabPFN-TS Hoo et al. (2025) adapts a tabular prior-fitted network with no time-series pretraining. However, because these TS-FMs are predominantly trained on web-scraped, financial, or ecological data, we hypothesize that they lack grounding in physical actuation and may struggle to disentangle static payload biases from dynamic behavior. FactoryNet bridges these tracks: an industrial-scale, control-theoretically structured corpus for training and evaluating TS-FMs on complex dynamics across embodiments.

2.3 Physics-Informed and Structured Dynamics

In parallel to data-driven forecasting, Physics-Informed Neural Networks (PINNs) Raissi et al. (2019) and grey-box models incorporate known governing equations into the learning process. While these approaches excel in scenarios with well-defined partial differential equations, complex multi-joint robotic systems are commonly approached with structured priors or residual-dynamics learning, as in TossingBot Zeng et al. (2020). FactoryNet complements this literature with the empirical Setpoint-Effort-Feedback decomposition needed to train and evaluate such models at scale.

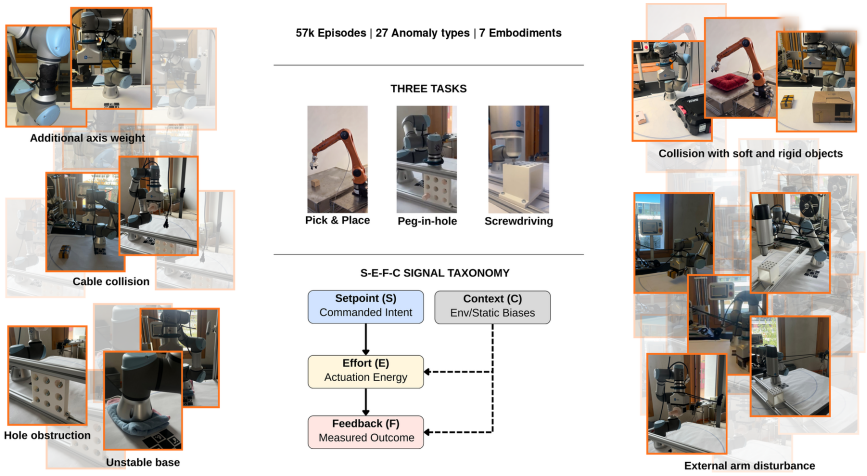

Figure 2: **FactoryNet overview.** A multi-embodiment industrial time-series corpus spanning 57k episodes (14.8k real, 41.7k simulated) on seven embodiments, 27 anomaly types, and three manipulation tasks. Every signal is mapped into the Setpoint-Effort-Feedback-Context (S-E-F-C) taxonomy, a control-theoretic decomposition that enables cross-embodiment learning. Representative fault types are shown alongside the corresponding laboratory setups.

3 The FactoryNet Dataset

FactoryNet v1.0 is to our knowledge, the largest open-source industrial-robot time-series dataset containing labelled anomalies and organized around a control-theoretic schema. It comprises three pillars: real-world laboratory recordings, standardized open-source adaptations, and a synthetic generation pipeline. Because industrial data is sampled at high frequencies across many joints and sensors, 56.6k episodes yield a high-density, continuous corpus of the kind required to learn complex physical dynamics.

3.1 Dataset Composition

As detailed in Table 2 and Figure 1, the corpus encompasses multiple data streams. The laboratory track consists of novel recordings we collected from UR3 and KUKA KR10 industrial robotic arms executing three complex tasks: Pick & Place, Screwdriving, and Peg-in-Hole; Figure 2 shows the cells and representative injected faults. We ingested and restructured existing high-quality open-source datasets (voraus-AD, AURSAD, and UMich CNC Sun (2018)) into our unified schema. The dataset is structured using a hierarchical taxonomy we propose, inspired by the reference designation principles of IEC 81346, which lets the schema integrate diverse modalities such as CNC milling centres and rotating machinery. A parallel synthetic track generated via NVIDIA Isaac Sim provides procedurally scaled data for model pretraining.

Table 2: FactoryNet composition. Episodes are normal and fault recordings (1,553 real counterfactual episodes come in addition). Datapoints are logged time steps and are approximate.

Source	Machine	Tasks	Faults	Episodes	Datapoints
Our Lab (Real)	UR3	P&P, Screw, Peg	Yes	8,863	11M
Our Lab (Real)	KUKA KR10	P&P	Yes	1,799	4M
Open (Real)	voraus-AD (Yu-Cobot)	P&P	Yes	2,122	16M
Open (Real)	AURSAD (UR3e)	Screw	Yes	2,045	3M
Open (Real)	UMich CNC	Machining	Yes	18	18K
Synthetic	Isaac Sim (5 arms)	P&P	Yes	41,744	79M
Total				56,591	113M

3.2 THE S-E-F-C SIGNAL TAXONOMY: A UNIFIED VOCABULARY

The defining feature of FactoryNet is its control-theoretic structure. Most existing benchmarks simply log raw sensor streams, intrinsically tangling the controller's target variables with the machine's actual physical execution. This conflates cause and effect while suffering from vendor-specific naming conventions.

To resolve this, we employ embodiment-specific adapter scripts to programmatically map over 300 heterogeneous data columns into four standardized roles (machine-readable tables: Appendix B). These categories are: **Setpoint (S)** (commanded intent, e.g., target joint positions), **Effort (E)** (actuation energy expended, e.g., motor current or torque, where exposed), **Feedback (F)** (measured physical outcome, e.g., actual positions), and **Context (C)** (environmental or static variables, e.g., payload mass).

This taxonomy allows a single dataloader to work across UR3, KUKA KR10, and CNC machinery: a 6-DOF rotational arm and a 4-axis CNC gantry expose the same four roles, only with different axis counts and units. By enforcing this taxonomy, S-E-F-C acts as a unified vocabulary for cross-embodiment models, providing a principled inductive bias for learning the difference between expected dynamics and external disturbances. Signal availability varies by embodiment: KUKA KR10 (KSS 8.3) does not expose joint velocities, commanded TCP pose, or TCP force/torque via its RSI interface; these channels are marked absent in the S-E-F-C mapping (Appendix B).

3.3 SYNTHETIC PIPELINE AND SIM-TO-REAL

To reduce dependence on real-only data, FactoryNet includes a synthetic pipeline in NVIDIA Isaac Sim 5.1 / Isaac Lab 2.3 (GPU-batched, PhysX at 500 Hz). It yields 41,744 validated Pick & Place episodes across five arms (UR3, UR5, UR10, UR30, KUKA KR10) with aligned S-E-F-C metadata: 10,316 nominal and 31,428 fault episodes spanning six fault classes. Control and logging run at 125 Hz (UR) and 83.3 Hz (KR10, the KRC interpolation cycle), with no resampling. The UR3 and KR10 motion profiles (velocity, acceleration, path types) are fitted to real recordings; the UR5, UR10 and UR30 profiles have no real counterpart and are scaled from the UR3 fit, an assumption we make explicit.

- **Payload and contact:** part mass is sampled per episode from $U(0.30, 1.20)$ kg (UR) and $U(0.30, 2.50)$ kg (KR10). Part friction is sampled from $U(0.6, 1.2)$ against a fixed pad coefficient of 1.0. Robot link masses and arm drive gains are fixed. The gripper gain is derived per episode from the required grip force rather than sampled.
- **Task and geometry:** pick and place poses are sampled within each station's workspace with random yaw, and part dimensions are randomized within bounds. Execution speed is scaled by $U(0.76, 1.24)$.
- **Faults:** six classes (payload addition, motor miscommutation, gripper activation failure, gripper release, collision, path obstacle) are injected at a phase drawn uniformly among the valid phases of each episode (Appendix D).
- **Physics and control:** each arm is an Isaac Lab articulation with a welded Robotiq 2F-85 gripper, driven by joint-space PD controllers (critically damped; stiffness fixed per arm and scaled from the UR3 values by torque limit). A software model of the vendor motion layer sits above them: trapezoidal joint and linear profiles, analytic inverse kinematics, computed-torque feedforward (gravity, Coriolis, inertia) and a joint-friction prior. The UR3 and KR10 profiles are fitted to 109 and 280 real nominal episodes. This is a simplified stand-in for the proprietary UR and KRC controllers, not a copy of their control law, and it has known deviations (for example, the logged tracking error is inflated relative to the real robot); gains, limits and deviations are listed in Appendix F.

Validity gate. Each generated episode passes a two-stage gate before release. The structural stage requires a plausible length, complete task phases, a placed part for nominal episodes, the injected event present in the log, and non-zero contact force. The trace stage requires every fault episode to leave a measurable physical trace, such as an outcome failure, a torque excursion above the nominal envelope, a stalled grip, or a protective stop; otherwise it is quarantined. Of 50,000 generated episodes, 41,744 pass (nominal 10,316 of 12,500; fault 31,428 of 37,500). Most fault rejections (5,138 of

6,072) are faults that left no trace. This choice ensures that every released fault label corresponds to a physical effect, but it also biases the synthetic faults toward more detectable instances, and acceptance rates differ across fault classes (e.g., motor miscommutation passes far less often than gripper faults). Motor miscommutation is excluded for the KR10, where its effect is sub-threshold. Per-arm acceptance rates are given in Appendix D.

Measured sim-to-real gap. An earlier UR3 campaign paired 1,155 real and simulated episodes and measured the per-phase gap directly (Appendix H); it used Isaac Sim 4.5.0 and a different gripper from the laboratory setup, so its absolute pose errors are conservative. Whether the simulator is close enough for a *model* to transfer is the question Section 4.3 tests directly, and is the one that matters for this corpus.

3.4 Faults and Anomalies

To support anomaly detection and robust control research, of the 10,662 laboratory normal and fault episodes, approximately 28% are healthy and 72% contain injected faults across 27 anomaly types spanning three tasks: **Pick & Place**, **Screwdriving**, and **Peg-in-Hole**. A further 1,553 real counterfactual episodes accompany the faulty runs.

3.5 Data Accessibility and Licensing

Novel laboratory and synthetic data are released under the MIT license; adapted subsets retain their original licenses, listed per source in Appendix A, with voraus-AD under CC BY-NC-SA 4.0, so that subset stays NonCommercial and ShareAlike. The anonymized link on the first page provides S-E-F-C Parquet files, metadata and framework-native dataloaders: at submission it serves every real subset in final form and the simulation data behind the reported experiments, while the full Isaac Lab synthetic campaign is still uploading during review. The verification script reproduces every reported value from stored result files, without the corpus download.

4 Dataset Utility & Validation

We ask two questions of the corpus. First, does the S-E-F-C schema support competitive anomaly detection on a single machine with far fewer signals? Second, which parts of a monitoring model carry over from simulation to real robots and between real robots? The first is a baseline that shows the schema is usable; the second is the main study. Confidence intervals are 95% bootstrap intervals over episodes unless stated otherwise.

4.1 Single-Machine Baseline: Validating the S-E-F-C Schema

We use voraus-AD to test whether S-E-F-C enables competitive anomaly detection via supervised dynamics rather than holistic reconstruction. We follow the official protocol of Brockmann et al. (2024): training on 948 healthy episodes only, testing on the 1,174-episode labelled set (419 healthy, 755 anomalous). Unstructured baselines reconstruct all 130 channels end-to-end and use the architectures and hyperparameters of the original benchmark. Our **S-E-F-C MLP** is a regressor on 24 signals: it maps 18 Setpoints (`setpoint_pos_0...5`, `setpoint_vel_0...5`, `setpoint_acc_0...5`) to 6 Efforts (`effort_motor_torque_0...5`), and the per-episode mean absolute error on Effort is the anomaly score. The network has three hidden layers (512, 256, 128; ReLU, no dropout) and is trained with Adam (learning rate 5×10^{-4}, weight decay 10^{-5}, batch size 4,096, cosine annealing, up to 500 epochs, early stopping with patience 30). On 24 signals alone it reaches **83.2%** mean AUROC. Table 3 shows it beats weaker full-channel baselines (1-NN, GANF, PCA) but not the strongest ones (CAE, LSTM-VAE, HMM, MVT-Flow). Per-category results are in Appendix E: the MLP is strongest on mechanically distinctive faults (miscommutation 99.2, additional axis weight 95.8) and weakest on brief gripping failures (collision with cables 67.6, losing can 71.8), where a single-step model has no temporal context.

Table 3: Mean AUROC on voraus-AD. Rows marked (Ours) are this work's own results; all other values are quoted from Brockmann et al. (2024). Baselines use 130 channels, our S-E-F-C MLP 24 (Setpoint→Effort). ‡Std across 12 fault categories.

Method	Input Channels	Mean AUROC
MLP - All signals (Ours)	130	69.1 ± 3.1‡
1-NN	130	77.5
GANF	130	79.9 ± 12.7‡
PCA	130	80.0
S-E-F-C MLP (Ours)	**24**	**83.2 ± 9.6**‡
CAE	130	85.2 ± 9.2‡
LSTM-VAE	130	86.7 ± 10.1‡
HMM	130	87.4 ± 5.8‡
MVT-Flow	130	93.6 ± 5.7‡

4.2 Which S-E-F-C Roles Carry the Signal

The schema is our central claim, so we test it directly rather than by a single number. Holding the architecture, the training set and the evaluation protocol of Section 4.1 fixed, we vary only which roles supply the inputs and which supply the prediction target, and run five seeds per variant (Table 4).

Table 4: S-E-F-C role ablation on voraus-AD, official split, identical architecture and protocol throughout; only the role assignment changes. Mean AUROC over the 12 fault categories, ± standard deviation across 5 seeds (10 draws for the random baseline). MVT-Flow, the strongest published baseline, scores 93.6 on 130 channels.

Inputs	Target	Channels	Mean AUROC
Setpoint (pos, vel, acc)	Feedback (torque sensors)	18→6	**94.8 ± 0.3**
Feedback + Context	Effort (motor torque)	64→6	89.5 ± 0.7
Setpoint + Feedback	Effort	60→6	87.9 ± 0.6
All non-Effort roles	Effort	88→6	86.4 ± 1.0
Random 18 of the other 124	Effort	18→6	*83.3 ± 5.2*
Setpoint (pos, vel, acc)	Effort *(Sec. 4.1)*	18→6	82.8 ± 0.6
All 124 non-target signals	Effort	124→6	76.2 ± 0.8
Setpoint (pos, vel, acc)	Feedback (joint pos, vel)	18→12	61.3 ± 0.2
Context only	Effort	28→6	60.9 ± 1.9

Role choice matters more than channel count. Across the role assignments and channel budgets we evaluate, mean AUROC varies by 33.9 points, and the variant that sees *every* available signal is the third worst: adding channels without regard to their role actively hurts. The target role dominates: from the same 18 commanded-motion inputs, predicting measured torque reaches 94.8, above the strongest published voraus-AD baseline, predicting motor Effort 82.8 (a 12.0-point drop at an identical 18→6 budget), and predicting measured *kinematics* 61.3, because a position residual carries almost no information about load while a torque residual carries most of it.

The taxonomy is a search space, not a guarantee. The particular pairing used in Section 4.1 is not itself special: a random 18-channel input set scores the same on average (83.3 against 82.8). The two deviations are not comparable (5.2 across channel draws, 0.6 across seeds at one fixed assignment), and the substantive point is the ceiling: no random draw reaches the Setpoint→Feedback result. We therefore read S-E-F-C as a small structured space of role pairings that is cheap to search and contains a state-of-the-art point, not as a claim that any one pairing is good. This also bounds the claim: the ablation covers one benchmark and one architecture family, and the ranking of roles on other machines is untested.

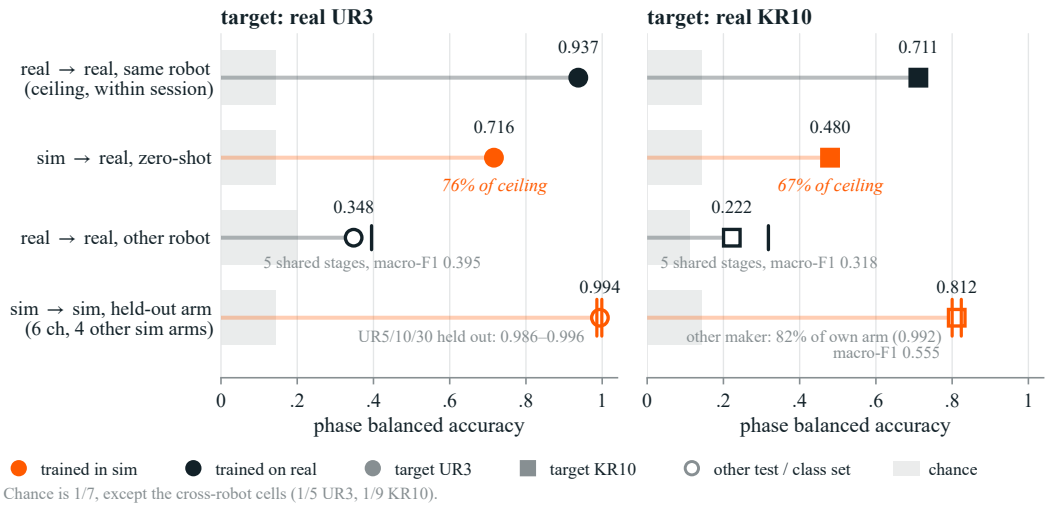

Figure 3: **Task phase transfers from simulation; between real robots it does not.** 7-stage balanced accuracy on the real UR3 and KR10: the within-session real ceiling, sim→real zero-shot, real cross-robot transfer (tick: macro-F1 on the five shared stages) and sim→sim leave-one-arm-out on the same six channels. Whisker = 95% bootstrap interval. A model trained with no real phase labels recovers 76% (UR3) and 67% (KR10) of the real-data ceiling on the sessions that emit all seven stages – 68% and 66% pooling every session (Section I.3) – while a model trained on the other *real* robot does markedly worse.

4.3 Transferability Study: Simulation to Real and Robot to Robot

We use FactoryNet to ask what a monitoring model learns in simulation, or on another robot, that still works on a real machine. We test two tasks: phase identification (which of the ten stages of the pick-and-place cycle is running) and binary fault detection. Both use MiniRocket, a fixed convolutional-kernel transform from the ROCKET family that is not pretrained (Dempster et al., 2021), with a ridge classifier, and are compared against shuffled-label nulls and bootstrap intervals. Real test sets are single sessions, for the reason given under Caveats below. Protocol and all cells are in Appendix I.

Task phase carries over from simulation. A model trained on simulation, with no real phase labels, recovers roughly three quarters of the accuracy of a model trained on real data (76% on the UR3, 67% on the KR10; Figure 3). Its confusions are systematic rather than random, which suggests that it captures the order of the stages better than the timing of real recordings. Between simulated arms, transfer to an unseen arm is strong inside the UR family and weaker across manufacturers.

Between real robots, transfer is limited. A phase model trained on one real robot transfers only weakly to the other (Figure 3), while the same robot transfers reasonably across programs and sessions. The stage labels appear to mark the same physical events on both robots, whereas the task programs differ in cycle time, pick-to-place distance and per-stage speed. Simulation does not separate the explanations: with a shared task program, simulated UR3 and KR10 transfer to each other only weakly as well. We therefore hypothesise that the gap reflects task program and embodiment jointly; our data do not tell them apart.

Fault signatures appear to be more specific. Uncalibrated simulation gives no detectable fault signal on the real robots in any configuration we tried, and aligning the feature distributions does not change this (Figure 4). Real-to-real detection works within a session for fault types seen in training, and is near chance for unseen types. This suggests, tentatively, that the models pick up fault-specific signatures rather than a generic anomaly direction; adding real episodes from other sessions did not raise the ceiling.

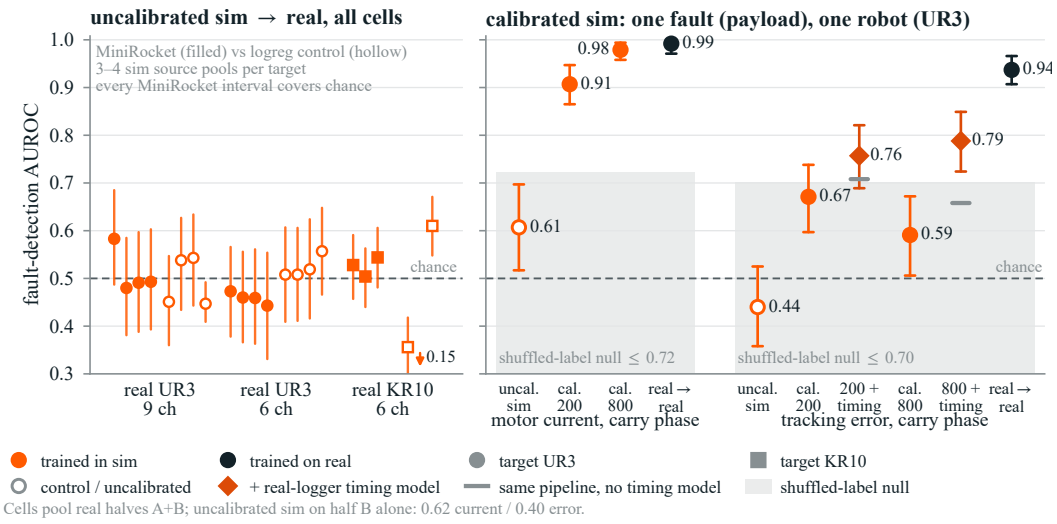

Figure 4: **Fault signatures do not transfer from uncalibrated simulation, and calibrating one channel recovers one fault.** Fault AUROC on real data. Left: uncalibrated sim→real, in every configuration we tried. Right: simulation calibrated to a single payload fault on a single robot (UR3), with and without the real logger's timing model, against the real→real ceiling. "cal. 200" and "cal. 800" are calibration campaigns of 200 and 800 simulated episodes. The right-hand panel is a case study, not a general result.

Calibration may help where the effort channel is matched. As a first probe we calibrated the simulator to one fault, a payload change on the UR3. A detector trained on simulation alone detects it in real motor current (AUROC 0.91–0.98, against 0.61 uncalibrated), and in joint tracking error only once the simulator also models the real logger's timing (0.76–0.79, against 0.59–0.67). One reading is that sim-trained detectors transfer when the simulator reproduces the physics of the channel they rely on. It is one fault on one robot, calibrated with real episodes of that fault and not carrying over to the KR10: a case study.

Caveats. Recording program, day and speed override partly predict the fault label on both robots, so we test within single sessions; only two of the four KR10 fault types are free of a speed confound, and the clean test sets are small. In the calibrated case the real classes also differ in gripper dwell, so a class-linked recording effect cannot be fully excluded. UR5, UR10 and UR30 exist only in simulation, so their evidence is simulation to simulation.

5 Conclusion & Outlook

FactoryNet provides a large open corpus of fault-injected industrial-robot time series, real and simulated, unified by S-E-F-C. The schema is not cosmetic: at a fixed architecture and a matched 18→6 channel budget, which roles predict which is worth 12 AUROC points on voraus-AD. Our transfer study suggests that task phase may be a stable structure simulation can capture, recovering roughly two thirds of the real-data ceiling with no real phase labels, that this holds inside a robot family more than across manufacturers, and that fault signatures are more specific: uncalibrated simulation gave no detectable fault signal on real robots, while a simulator calibrated to one payload fault detected it on the real UR3. These are hypotheses, not settled explanations.

Each boundary marks work the corpus makes possible. Recording the UR5, UR10 and UR30, today simulation-only, would make the multi-arm evidence real; a task program matched across manufacturers would separate program, tempo and embodiment as the cause of weak cross-robot transfer; session-balanced collection would remove the recording context that predicts fault labels. None of that needs a new dataset, only the same schema on more machines that share no vendor, kinematics or sensor set: the step from models of one machine toward foundation models for industry.

Reproducibility statement

All model architectures, optimizers, learning rates and training schedules are specified in Section 4.1. Evaluation splits follow the published protocol of the source benchmarks and are stated explicitly per experiment. The complete S-E-F-C signal mapping for every embodiment, the fault catalog, and the dataset datasheet are given in the appendix. The dataset, the adapter scripts that produce the S-E-F-C Parquet files, and the training and evaluation code are available at the anonymized links on the first page; a verification script there re-resolves every value plotted in Figures 3 and 4 against the raw result files, and runs without downloading the corpus.

Ethics statement

This work releases telemetry recorded from industrial robot arms and CNC machinery; it contains no human subjects, personal data, or personally identifiable information. Faults are deliberately injected under controlled laboratory conditions on dedicated hardware. Novel laboratory and synthetic data are released under the MIT license; adapted open-source subsets retain their original licenses and are redistributed in accordance with them.

AI use statement

We used generative AI assistants in preparing this manuscript, and disclose their use in the two categories the ICLR policy distinguishes. In the supportive category, they were used to draft and revise sections of the paper, to improve readability, and to write auxiliary scripts that render the figures. In the research-critical category, they were used as a coding and checking tool: to help write analysis code for the transfer study, and to cross-check reported numbers against the underlying result files.

We did not use generative AI to conceive the S-E-F-C schema, to formulate any claim or hypothesis in this paper, or to generate any data reported here. The corpus, the laboratory recordings, the simulation pipeline and its synthetic episodes are the authors' own work; the synthetic track is generated procedurally by a physics simulator, not by a language model. Every experimental result was produced by running the released code on the released data.

All AI-assisted text and code was reviewed by the authors before use. Every quantitative claim in the paper was checked against the underlying result files by at least one author, and the verification script released with the code re-resolves each value plotted in Figures 3 and 4 from those files independently. The authors take full responsibility for all content of this work, including any text, claims or artifacts produced with the aid of generative AI.

References

Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P. Mathur. WADI: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, pp. 25–28, New York, NY, USA, 2017. Association for Computing Machinery. doi: 10.1145/3055366.3055375. URL https://doi.org/10.1145/3055366.3055375.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=gerNCVqqtR.

Jan Thieß Brockmann, Marco Rudolph, Bodo Rosenhahn, and Bastian Wandt. The voraus-AD dataset for anomaly detection in robot applications. *IEEE Transactions on Robotics*, 40:438–451, 2024. doi: 10.1109/TRO.2023.3332224. URL https://ieeexplore.ieee.org/document/10315239.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing*

Systems, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167. PMLR, 2024. URL https://proceedings.mlr.press/v235/das24c.html.

Angus Dempster, Daniel F. Schmidt, and Geoffrey I. Webb. MiniRocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 248–257. ACM, 2021. doi: 10.1145/3447548.3467231.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

J.J. Downs and E.F. Vogel. A plant-wide industrial process control problem. *Computers & Chemical Engineering*, 17(3):245–255, 1993. doi: 10.1016/0098-1354(93)80018-I. URL https://doi.org/10.1016/0098-1354(93)80018-I.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12): 86–92, 2021. doi: 10.1145/3458723. URL https://doi.org/10.1145/3458723.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 16115–16152. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/goswami24a.html.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending TabPFN-v2 to time series forecasting. *arXiv preprint arXiv:2501.02945*, 2025.

Iurii D. Katser and Vyacheslav O. Kozitsin. Skoltech anomaly benchmark (SKAB). https://www.kaggle.com/dsv/1693952, 2020.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, et al. Droid: A large-scale in-the-wild robot manipulation dataset. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. URL https://www.roboticsproceedings.org/rss20/p120.html.

Kwei-Herng Lai, Daochen Zha, Guanchu Wang, Junjie Xu, Yue Zhao, Devesh Kumar, Yile Chen, Purav Zumkhawaka, Minyang Wan, Diego Martinez, and Xia Hu. TODS: An automated time series outlier detection system. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 16060–16062, 2021.

Yaguo Lei, Tianyu Han, Biao Wang, Naipeng Li, Tao Yan, and Jun Yang. XJTU-SY rolling element bearing accelerated life test datasets: A tutorial. *Journal of Mechanical Engineering*, 55(16):1–6, 2019. doi: 10.3901/JME.2019.16.001.

Błażej Leporowski, Daniella Tola, Casper Hansen, and Alexandros Iosifidis. AURSAD: Universal robot screwdriving anomaly detection dataset. *arXiv*, 2021. URL https://arxiv.org/abs/2102.01409.

Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, and Walter Sextro. Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: a benchmark data set for data-driven classification. In *Proceedings of the European Conference of the Prognostics and Health Management Society*, 2016.

Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. In *Advances in Neural Information Processing Systems*, 2024. Datasets and Benchmarks Track.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Generative pre-trained transformers are large time series models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32369–32399. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24cb.html.

Kenneth A. Loparo. Case Western Reserve University Bearing Data Center. https://engineering.case.edu/bearingdatacenter, 2000. Accessed 2026-09-26. The data centre states no collection or publication date; 2000 follows common usage in this literature.

Matheus A. Marins, Felipe M. L. Ribeiro, Sergio L. Netto, and Eduardo A. B. da Silva. MAFAULDA :: Machinery fault database. http://www02.smt.ufrj.br/~offshore/mfs/page_01.html, 2016. Signals, Multimedia, and Telecommunications Laboratory, COPPE/Poli/UFRJ; contact: felipe.ribeiro@smt.ufrj.br.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10611477. URL https://ieeexplore.ieee.org/document/10611477.

PHM Society. 2010 PHM Society Conference Data Challenge. https://phmsociety.org/phm_competition/2010-phm-society-conference-data-challenge/, 2010. Dataset and challenge page for remaining useful life estimation of high-speed CNC milling machine cutters; accessed 2026-05-08.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.10.045. URL https://doi.org/10.1016/j.jcp.2018.10.045.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, et al. Lag-llama: Towards foundation models for probabilistic time series forecasting. *arXiv*, 2023. URL https://arxiv.org/abs/2310.08278.

Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, pp. 1–9. IEEE, 2008. doi: 10.1109/PHM.2008.4711414. URL https://ieeexplore.ieee.org/document/4711414.

Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016. doi: 10.1609/aaai.v30i1.10306.

Sharon Sun. CNC Mill Tool Wear, 2018. URL https://www.kaggle.com/datasets/shasun/tool-wear-detection-in-cnc-mill. CC0 Public Domain; machining experiments at the System-level Manufacturing and Automation Research Testbed (SMART), University of Michigan; dataset metadata last modified 2018-04-06 on Kaggle.

Bruno Veloso, Rita P. Ribeiro, João Gama, and Pedro Mota Pereira. The MetroPT dataset for predictive maintenance. *Scientific Data*, 9(1):764, 2022. doi: 10.1038/s41597-022-01877-3.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53140–53164. PMLR, 2024. URL https://proceedings.mlr.press/v235/woo24a.html.

World Bank. Manufacturing, value added (% of GDP): World aggregate, 2026. URL https://data.worldbank.org/indicator/NV.IND.MANF.ZS. Series NV.IND.MANF.ZS, World aggregate; 2025 observation (latest available), 14.9%. Accessed: 2026-04-30.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly Transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=LzQQ89U1qm_.

Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. TossingBot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4): 1307–1319, 2020. doi: 10.1109/TRO.2020.2988642.

A DATASHEET FOR DATASETS (GEBRU ET AL., 2021)

Following the recommendations of Gebru et al. (2021), we provide a structured datasheet for FactoryNet v1.0 (Table 5).

Table 5: Datasheet for FactoryNet v1.0.

Question	**Answer**
Motivation	
Purpose	Pretraining substrate for industrial time-series foundation models, anomaly detection benchmarking, and cross-embodiment transfer research.
Creators	Anonymized for double-blind review; the full creator list is disclosed in the camera-ready version.
Funding	Anonymized for double-blind review.
Composition	
Instances	Episodes of robotic task execution, each containing multi-channel time-series at the controller rate with S-E-F-C role annotations.
Count	56,591 episodes: 8,863 real lab (UR3), 1,799 real lab (KUKA KR10), 4,185 open-source (voraus-AD 2,122; AURSAD (UR3e) 2,045; UMich CNC 18), and 41,744 synthetic (Isaac Sim, five arms). 113M logged time steps. 1,553 real counterfactual episodes come in addition.
Sampling	Lab data: programmatic task execution with configurable fault injection; of the 10,662 laboratory episodes, 28% are healthy and 72% contain an injected fault. Open-source: full datasets ingested. Synthetic: procedural generation with domain randomization.
Missing data	Sensor dropouts $<0.1\%$ of time steps; handled via linear interpolation. No episodes removed due to missing data.
Confidentiality	No PII. Industrial parameters only (joint angles, torques, velocities).
Collection Process	
Acquisition	Lab episodes recorded by the authors via robot controller APIs (RTDE for UR3, RSI for KUKA KR10). Synthetic episodes are logged at 125 Hz (UR) and 83.3 Hz (KR10, the KRC interpolation cycle), with no resampling. Open-source subsets downloaded and re-processed into S-E-F-C format via adapter scripts.
Mechanisms	Automated scripts executing task programs with fault injection at configurable rates. 27 anomaly types, prepared physically on the cell where the fault requires it (for example a loosened fixture or an added axis weight) and otherwise injected programmatically (payload variation, misgrips, collisions, stripped threads, jamming, friction changes, etc.).
Who collected	Anonymized for double-blind review.
Time frame	March 1 – April 20, 2026.
Ethical review	N/A: no human subjects.
Preprocessing, Cleaning, and Labeling	
Preprocessing	NaN interpolation and S-E-F-C column mapping per source via adapter scripts; each source keeps its native controller rate. No per-channel normalisation applied (raw physical units preserved).
Raw data	Raw recordings available on request. Distributed format: S-E-F-C Parquet files with metadata.
Labeling	Programmatic fault injection with known ground truth; no manual annotation required for lab data. Open-source subsets retain original labels, mapped to unified anomaly taxonomy.
Uses	
Prior uses	Anomaly detection baselines (this paper), cross-embodiment transfer experiments (this paper).
Other uses	Forecasting, remaining useful life estimation, control policy learning, sim-to-real benchmarking, counterfactual analysis.
Misuse potential	Dataset reflects specific robot models and lab conditions; generalisation claims to unseen embodiments or industrial deployments should be scoped accordingly.
Distribution	

Question	Answer
Distribution	Distributed as S-E-F-C Parquet files with metadata and framework-native dataloaders; the anonymized review copy is linked from the first page of this paper.
License	Novel lab and synthetic data under MIT. Open-source subsets retain their original licenses and are redistributed under them: voraus-AD is CC BY-NC-SA 4.0 (NonCommercial, ShareAlike), AURSAD is CC BY 4.0, and the UMich CNC subset follows its source terms.
Restrictions	The voraus-AD subset is CC BY-NC-SA 4.0: NonCommercial and ShareAlike. Commercial use of that subset, and redistribution of works derived from it under more permissive terms, are not permitted. Novel and synthetic data carry no such restriction. No export controls apply.
Maintenance	
Maintainer	Anonymized for double-blind review. Issue tracker: project repository.
Updates	Semantic versioning (v1.0 at submission). New embodiments and tasks increment minor version; schema changes increment major version. Prior versions remain available.
Retention	All released versions are retained permanently.
Errata	Reported and tracked in the public issue tracker of the released repository; corrections published as patch versions.

B SIGNAL-TO-S-E-F-C MAPPING

We provide one table per source mapping raw signal names to S-E-F-C roles. UR3, KUKA KR10 and Isaac Sim mappings are listed with raw signal names confirmed; CNC, AURSAD, and voraus-AD mappings are fully confirmed.

B.1 UR3 LABORATORY SOURCE

Signals of UR3 can be found at Table 6

B.2 KUKA KR10 LABORATORY SOURCE

Signals of KUKA KR10 can be found at Table 7

B.3 UMICH CNC SOURCE

Mapping of CNC columns (Table 8)

B.4 AURSAD SOURCE (UR3E SCREWDRIVING)

Per-joint signals follow a fixed pattern for joints $i \in \{0, \ldots, 5\}$; one representative joint (joint 0) is shown below. (Table 9)

B.5 VORAUS-AD SOURCE (YU-COBOT)

Per-joint signals follow the same index pattern for joints $i \in \{1, \ldots, 6\}$ (1-indexed in source); one representative joint is shown. (Table 10)

B.6 ISAAC SIM UR5 (SYNTHETIC)

Mapping of Isaac Sim UR5 columns (Table 11)

C FULL FAULT CATALOG

Table 12 lists all 27 annotated anomaly types used during laboratory data collection. One of the 27, the loosening phase, is a normal phase with a distinct signature rather than a fault; it is counted as an

Table 6: S-E-F-C mapping for the UR3 laboratory source.

Raw Signal Name	S-E-F-C Role	Unit	Notes
machine_id	Metadata	–	Episode metadata
ur3_robot_target_joint_0...5	Setpoint	rad	6 joints
ur3_robot_target_joint_vel_0...5	Setpoint	rad/s	6 joints
ur3_robot_joint_0...5	Feedback	rad	Actual positions, 6 joints
ur3_robot_joint_vel_0...5	Feedback	rad/s	Actual velocities, 6 joints
ur3_robot_joint_current_0...5	Effort	A	6 joints
ur3_robot_target_joint_current_0...5	Effort	A	Commanded current, 6 joints
ur3_robot_joint_temp_0...5	Context	°C	6 joints
ur3_robot_joint_control_output_0...5	Effort	–	6 joints
ur3_robot_joint_mode_0...5	Context	enum	6 joints
ur3_robot_tcp_x/y/z	Feedback	m	Actual TCP position
ur3_robot_tcp_rx/ry/rz	Feedback	rad	Actual TCP orientation
ur3_robot_target_tcp_x/y/z	Setpoint	m	Target TCP position
ur3_robot_target_tcp_rx/ry/rz	Setpoint	rad	Target TCP orientation
ur3_robot_tcp_speed_x/y/z/rx/ry/rz	Feedback	m/s & rad/s	Actual TCP speed (6 axes)
ur3_robot_target_tcp_speed_x/y/z/rx/ry/rz	Setpoint	m/s & rad/s	Target TCP speed (6 axes)
ur3_robot_tcp_force_x/y/z	Effort	N	TCP force
ur3_robot_tcp_torque_x/y/z	Effort	Nm	TCP torque
ur3_robot_robot_mode	Context	enum	Enum
ur3_robot_safety_mode	Context	enum	Enum
ur3_robot_digital_inputs	Context	bitmask	Bitmask
ur3_robot_digital_outputs	Context	bitmask	Bitmask
ur3_robot_robot_status	Context	enum	
ur3_robot_safety_status	Context	enum	
ur3_robot_runtime_state	Context	enum	
ur3_robot_main_voltage	Context	V	V
ur3_robot_robot_voltage	Context	V	V
ur3_robot_robot_current	Context	A	A
ur3_robot_analog_input_0/1	Context	V/A	
ur3_robot_analog_output_0/1	Context	V/A	
ur3_robot_speed_scaling	Context	%	
ur3_robot_target_speed_fraction	Context	%	
ur3_robot_momentum	Context	kg·m/s	
realsense_camera_frame_count	Context	count	
realsense_camera_connected	Context	bool	
realsense_camera_streaming	Context	bool	

Table 7: S-E-F-C mapping for the KUKA KR10 laboratory source. Note hardware limitations of KSS 8.3.

Signal Group	Described Fields	S-E-F-C Role	Unit
Position (actual)	Actual joint positions (×6)	Feedback	degrees
Position (commanded)	Commanded joint positions (×6)	Setpoint	degrees
TCP pose (actual)	Actual TCP pose (×6)	Feedback	mm / degrees
Motor current	Motor current per axis (×6)	Effort	% of max
Motor torque	Motor torque per axis (×6)	Effort	Nm
Motor temperature	Motor temperature per axis (×6)	Context	Kelvin
Cartesian accel.	Accel. X, Y, Z, abs (×4)	Feedback	m/s^2
Speed override	Speed override (scalar)	Context	%
Process state	State enum (FREE/ACTIVE/STOP/END)	Context	enum
Digital I/O	Digital inputs bitmask	Context	bitmask
Digital I/O	Digital outputs bitmask	Context	bitmask
Not available on KSS 8.3: joint velocities, commanded TCP pose, TCP force/torque, joint voltage, tool accelerometer, joint control output.			

Table 8: S-E-F-C mapping for the UMich CNC milling source. Axes: X1, Y1, Z1 (linear), S1 (spindle).

Raw Signal Name	S-E-F-C Name	Role	Unit	Axis
X1_CommandPosition	setpoint_pos_0	Setpoint	mm	X
X1_CommandVelocity	setpoint_vel_0	Setpoint	mm/s	X
X1_CommandAcceleration	setpoint_acc_0	Setpoint	mm/s^2	X
X1_ActualPosition	feedback_pos_0	Feedback	mm	X
X1_ActualVelocity	feedback_vel_0	Feedback	mm/s	X
X1_ActualAcceleration	feedback_acc_0	Feedback	mm/s^2	X
X1_CurrentFeedback	feedback_current_0	Feedback	A	X
X1_OutputCurrent	effort_current_0	Effort	A	X
X1_OutputVoltage	effort_voltage_0	Effort	V	X
X1_OutputPower	effort_power_0	Effort	W	X
X1_DCBusVoltage	ctx_busvoltage_0	Context	V	X
(Pattern repeats for Y1 → axis 1, Z1 → axis 2)				
S1_CommandPosition	setpoint_pos_3	Setpoint	deg	Spindle
S1_CommandVelocity	setpoint_vel_3	Setpoint	rpm	Spindle
S1_CommandAcceleration	setpoint_acc_3	Setpoint	rpm/s	Spindle
S1_ActualPosition	feedback_pos_3	Feedback	deg	Spindle
S1_ActualVelocity	feedback_vel_3	Feedback	rpm	Spindle
S1_ActualAcceleration	feedback_acc_3	Feedback	rpm/s	Spindle
S1_CurrentFeedback	feedback_current_3	Feedback	A	Spindle
S1_OutputCurrent	effort_current_3	Effort	A	Spindle
S1_OutputVoltage	effort_voltage_3	Effort	V	Spindle
S1_OutputPower	effort_power_3	Effort	W	Spindle
S1_DCBusVoltage	ctx_busvoltage_3	Context	V	Spindle
S1_SystemInertia	ctx_inertia_3	Context	–	Spindle
M1_CURRENT_PROGRAM_NUMBER	ctx_program_number	Context	–	–
M1_sequence_number	ctx_sequence_number	Context	–	–
M1_CURRENT_FEEDRATE	ctx_feedrate	Context	mm/min	–
Machining_Process	ctx_process_phase	Context	enum	–

anomaly type to match the AURSAD label schema, so the corpus contains 26 fault types plus that phase.

D Synthetic Validity Gate and Acceptance Rates

Fault onsets are drawn uniformly over the phases in which the fault is valid (payload addition: lift, move, lower; gripper activation failure: settle, close; gripper release: lift, move; path obstacle: move; miscommutation and collision: motion phases), with a 30-step margin. A fault episode is kept only if it leaves a physical trace: an outcome failure or slip, a peak torque above 1.15× the nominal envelope,

Table 9: S-E-F-C mapping for the AURSAD source (representative joint 0 pattern; repeats for joints 1–5 with incremented index).

Raw Signal Name	S-E-F-C Name	Role	Unit	Notes
target_q_0	setpoint_pos_0	Setpoint	rad	Joint 0
target_qd_0	setpoint_vel_0	Setpoint	rad/s	
target_qdd_0	setpoint_acc_0	Setpoint	rad/s^2	
target_current_0	setpoint_current_0	Setpoint	A	
target_moment_0	setpoint_torque_0	Setpoint	Nm	
actual_q_0	feedback_pos_0	Feedback	rad	
actual_qd_0	feedback_vel_0	Feedback	rad/s	
actual_current_0	effort_current_0	Effort	A	
actual_control_output_0	effort_control_0	Effort	–	
actual_joint_voltage_0	effort_voltage_0	Effort	V	
joint_temperatures_0	ctx_temp_0	Context	°C	
joint_mode_0	ctx_joint_mode_0	Context	enum	
target_TCP_pose_0	setpoint_pos_cartesian_0	Setpoint	m/rad	
target_TCP_speed_0	setpoint_vel_cartesian_0	Setpoint	m/s	
actual_TCP_pose_0	feedback_pos_cartesian_0	Feedback	m/rad	
actual_TCP_speed_0	feedback_vel_cartesian_0	Feedback	m/s	
actual_TCP_force_0	effort_force_cartesian_0	Effort	N	
actual_tool_accelerometer_0/1/2	auxiliary_accel_tool_0/1/2	–	m/s^2	Tool IMU
output_double_register_24	feedback_torque_tool	Feedback	Nm	Tool torque
output_double_register_25	effort_torque_tool	Effort	Nm	
output_double_register_26	setpoint_torque_tool	Setpoint	Nm	
output_double_register_27	setpoint_torque_gradient_tool	Setpoint	Nm/s	
actual_main_voltage	ctx_main_voltage	Context	V	
actual_robot_voltage	ctx_robot_voltage	Context	V	
actual_robot_current	ctx_robot_current	Context	A	
speed_scaling	ctx_speed_scaling	Context	–	
target_speed_fraction	ctx_target_speed_fraction	Context	–	
actual_momentum	ctx_momentum	Context	kg·m/s	
robot_mode	ctx_robot_mode	Context	enum	
safety_mode	ctx_safety_mode	Context	enum	
runtime_state	ctx_runtime_state	Context	enum	
label	raw_label	–	–	0=Normal, 1=Damaged screw, 2=Extra part, 3=Missing screw, 4=Damaged thread

a carry-torque shift above 3 standard deviations, a grip stall below 0.5× the nominal minimum, or a protective stop. Path-obstacle episodes additionally need contact evidence. Table 13 lists valid episodes per arm and condition (ids: 6 payload, 12 miscommutation, 13 gripper activation failure, 14 gripper release, 16 collision, 17 path obstacle). Rates are out of 2,500 nominal and 1,250 per fault condition (KR10 path obstacle: 2,500 generated; miscommutation not generated).

E Per-Category Anomaly Detection on voraus-AD

Table 14 breaks the mean AUROC of Table 3 down by fault category. The S-E-F-C MLP is strongest where the fault changes the load the controller has to hold, such as miscommutation (99.2) and additional axis weight (95.8, the best of any method), and weakest on brief gripping failures such as collision with cables (67.6) and losing the can (71.8). Those are short, contact-driven events, and a single-step regressor from Setpoint to Effort has no temporal context with which to see them. The pattern is consistent with the role ablation in Section 4.2: what the model can detect is bounded by what the target role records.

F Simulator Physics and Controller

Engine. Isaac Sim 5.1.0 with Isaac Lab 2.3.2, GPU PhysX, headless, batched (1,250 environments per wave). Physics steps at 500 Hz (64/64 solver iterations); control and logging run every 4 physics steps (125 Hz) for the UR arms and every 6 (83.3 Hz, the KRC interpolation cycle) for the KR10.

Cell and task. Two stations with colliding decks, a ground plane and a dynamic cuboid part. The UR3 cell follows our laboratory cell; the UR5, UR10 and UR30 cells are the UR3 cell scaled horizontally by the ratio of reach (1.88, 2.84, 2.88). The KR10 cell is fitted to real KR10 station angles. The task shuttles the part between the stations (direction drawn per episode) in ten phases: above pick, descend, settle, close, lift, move, lower, release, retreat, return.

Table 10: S-E-F-C mapping for the voraus-AD source (representative joint 1 pattern; repeats for joints 2–6).

Raw Signal Name	S-E-F-C Name	Role	Unit	Notes
target_position_1	setpoint_pos_0	Setpoint	rad	Joint 1→idx 0
target_velocity_1	setpoint_vel_0	Setpoint	rad/s	
target_acceleration_1	setpoint_acc_0	Setpoint	rad/s^2	
target_torque_1	setpoint_torque_0	Setpoint	Nm	
joint_position_1	feedback_pos_0	Feedback	rad	
joint_velocity_1	feedback_vel_0	Feedback	rad/s	
motor_position_1	feedback_motor_pos_0	Feedback	rad	
motor_velocity_1	feedback_motor_vel_0	Feedback	rad/s	
torque_sensor_a_1	feedback_torque_a_0	Feedback	Nm	
torque_sensor_b_1	feedback_torque_b_0	Feedback	Nm	
motor_torque_1	effort_motor_torque_0	Effort	Nm	
motor_iq_1	effort_current_iq_0	Effort	A	
motor_id_1	effort_current_id_0	Effort	A	
power_motor_el_1	effort_power_el_0	Effort	W	
power_motor_mech_1	effort_power_mech_0	Effort	W	
power_load_mech_1	effort_power_load_0	Effort	W	
motor_voltage_1	effort_voltage_0	Effort	V	
computed_inertia_1	ctx_inertia_0	Context	–	
computed_torque_1	ctx_computed_torque_0	Context	Nm	
supply_voltage_1	ctx_busvoltage_0	Context	V	
brake_voltage_1	ctx_brake_voltage_0	Context	V	
robot_voltage	ctx_robot_voltage	Context	V	Global
robot_current	ctx_robot_current	Context	A	Global
io_current	ctx_io_current	Context	A	Global
system_current	ctx_system_current	Context	A	Global
anomaly	ctx_is_anomaly	–	bool	Label
category	ctx_anomaly_category	–	str	Fault type
setting	ctx_setting	–	str	

Motion layer. A software model of the vendor motion layer generates trapezoidal joint and Cartesian profiles with analytic inverse kinematics (all eight branches, batched in torch) and a fixed configuration branch. The UR3 profiles are fitted to 109 real UR3 pick-and-place episodes (joint speed 0.96 rad/s, acceleration 1.03 rad/s^2; linear acceleration 0.54 m/s^2), the KR10 profiles to 280 real KR10 nominal episodes (0.504 rad/s, 0.63 rad/s^2). The UR5, UR10 and UR30 profiles have no real data: the UR3 profile is scaled with the arm's reach and wrist, an assumption we declare rather than validate.

Joint control. Each joint has an implicit PD drive (stiffness k_p from Table 15, damping $2\zeta\sqrt{k_p(M_{ii}+I_a)}$ with $\zeta = 1$) plus computed-torque feedforward (gravity, Coriolis, inertia) clamped to 0.8 of the effort limit and a joint-friction prior (UR3 Coulomb 2, 2, 1.5, 0.6, 0.6, 0.6 N m). The UR3 stiffnesses are set from a torque disturbance of 1.5 mrad; the UR5, UR10 and UR30 values are the UR3 values scaled by the ratio of effort limits and were not measured. A protective-stop monitor is included. The gripper is one actuated joint with five mimic joints; its stiffness is derived each episode from the grip force needed to hold the sampled mass at the sampled friction, clipped to $[2000, 40000]$.

Known deviations from the real robots. (i) The real UR and KRC controllers are proprietary; the model reproduces their motion profiles, not their internal control law. (ii) In the released corpus the commanded joint position is logged one control tick ahead of the measured position, so the logged joint tracking error is approximately -8 ms times the commanded velocity, which inflates nominal tracking-error magnitude three to six times over the real robot. The calibrated case study (Section 4.3) corrects the alignment; the main corpus does not. (iii) The payload feedforward is applied for the nominal part mass for the whole episode with the wrong sign, adding the part's weight instead of compensating it; a corrected path exists but was used only in the calibration runs. (iv) The simulated TCP "tracking error" channel of an early ten-channel set is the distance to the current pick or place target, not a control error; the transfer study does not use it. (v) The protective-stop monitor is

Table 11: S-E-F-C mapping for the earlier Isaac Sim (UR5) synthetic source, retained for completeness: signals are procedurally generated and logged at a base 60 Hz simulation step before interpolation. This is *not* the campaign behind the results reported in the main text, which runs at 500 Hz physics and logs at 125 Hz (UR) and 83.3 Hz (KR10) with no resampling (Appendix F).

Signal Group (Raw Prefix)	Described Fields	S-E-F-C Role	Unit
episode, step, time	Global/episode steps, sim/wall time	Metadata	count / s
state_machine, phase	Task, controller phases, event IDs	Context	enum / str
joint_cmd_pos_rad_*	Commanded joint positions (×6)	Setpoint	rad
joint_cmd_vel_radps_*	Commanded joint velocities (×6)	Setpoint	rad/s
joint_pos_rad_*	Actual joint positions (×6)	Feedback	rad
joint_vel_radps_*	Actual joint velocities (×6)	Feedback	rad/s
joint_accel_radps2_*	Actual joint accelerations (×6)	Feedback	rad/s^2
joint_torque_nm_*	Joint applied torques (×6)	Effort	Nm
joint_pos_error_rad_*	Tracking error per joint (×6)	Context	rad
ee_cmd_pos_*, quat_*	Commanded TCP pose (pos, quat)	Setpoint	m / –
ee_pos_*, quat_*	Actual TCP pose (pos, quat)	Feedback	m / –
ee_euler_*_rad	Actual TCP orientation (RPY)	Feedback	rad
ee_linvel_*, angvel_*	TCP velocities (linear, angular)	Feedback	m/s / rad/s
ee_linacc_*, angacc_*	TCP accelerations (linear, angular)	Feedback	m/s^2 / rad/s^2
gripper_cmd_rad	Commanded gripper position	Setpoint	rad
gripper_pos_rad	Actual gripper position	Feedback	rad
gripper_attached	Gripper logical contact state	Context	bool
contact_force_*_n	Physical EE contact force (XYZ, mag)	Effort	N
contact_torque_*_nm	Physical EE contact torque (XYZ, mag)	Effort	Nm
cube_pos_*, quat_*	Target object pose (pos, quat)	Context	m / –
cube_linvel_*, angvel_*	Target object velocities	Context	m/s / rad/s
ee_cube_offset_*	Relative distance to target	Context	m
cube_mass_kg	Domain rand: payload mass	Context	kg
cube_friction_coeff	Domain rand: target object friction	Context	–
cube_width/depth/height_m	Domain rand: target dimensions	Context	m

effectively disarmed on the UR5 shoulder lift and UR10 joints 1–4, and rare contact impulses occur on the UR10 shoulder lift. (vi) Part dimensions are drawn once per process and reused across waves.

G Real-Data Transfer Summary

Table 16 collects the real-data transfer cells that do not appear in the main-text figures.

H Batch sim-to-real validation: an earlier UR3 campaign

This subsection reports an earlier UR3 validation campaign, run before the five-arm corpus above. We run sim2real rollouts in Isaac Sim 4.5.0 (headless Docker, PhysX) using the built-in UR3 USD asset and a position-control stack (`ArticulationAction`) with phase-consistent waypoint replay from real `target_joint_*` signals. The task is UR3 pick-and-place with a 10-phase structure (`above_pick` to `return`). Simulation steps at 240 Hz and logs at 10 Hz. Real episodes are exported from the laboratory recording stack to per-episode RTDE-shaped CSVs, converted to per-episode simulation configs, replayed in Isaac, and paired by episode ID/filename. Simulated logs preserve recording-stack-compatible fields (`joint_*`, `target_joint_*`, `joint_current_*`, `tcp_*`, `task_phase`, and status/context fields). Gap metrics are computed phase-wise with time normalization/interpolation. Results of the gap analysis are shown in Table 17.

An important source of residual pose discrepancy is end-effector mismatch: the Isaac setup used a Robotiq 2FG85-style gripper configuration, whereas the real laboratory episodes used an OnRobot 2FG14 gripper. Differences in tool geometry/TCP definition and mounting can bias absolute TCP and orientation metrics. We therefore interpret remaining TCP rotation spread conservatively and treat gripper-accurate tool calibration as future work. Whether the simulator is close enough to the real robots for a model to transfer is tested directly in Section 4.3.

Table 12: Complete Fault Catalog. Checkmarks (✓) indicate whether the fault is included in the Pick-and-Place (PP), Screwing (Scr), and Peg-in-Hole (PiH) task recordings. Crosses (×) indicate absence.

Fault	Explanation	Injection Procedure	PP	Scr	PiH
Damaged screw thread	Screw thread physically damaged, preventing proper engagement during tightening.	Pre-damaged the screw thread with sandpaper.	×	✓	×
Missing screw	Tightening is attempted with no screw present.	Removed the screw from screwdriver before the cycle.	×	✓	×
Damaged plate thread	The threaded hole in the plate is damaged, preventing engagement.	Pre-damaged the plate hole with a metal screwdriver.	×	✓	×
Loosening phase	A counterclockwise loosening rotation replaces tightening; a normal phase with a distinct signal. Counted as an anomaly to match AURSAD schema.	Reversed the rotation direction in the controller program to execute a counterclockwise loosening pass.	×	✓	×
Gripper activation failure	The vacuum gripper fails to activate and never picks up the box.	Disabled the gripper activation command in the robot program.	✓	×	×
Gripper release during motion	The gripper releases the payload mid-trajectory, causing an abrupt payload loss.	Triggered a programmatic gripper-release command at a scripted timestep mid-trajectory.	✓	×	×
Additional axis payload	A dead weight attached to one link increases inertia and gravity loading on all joints.	Bolted calibrated weights of varying mass to one robot link.	✓	✓	×
Collision with foam object	Contact with a soft foam block produces a brief TCP force spike without a protective stop.	Placed a foam cube directly in the programmed TCP trajectory.	✓	✓	✓
Unexpected payload weight	The transported box has a weight that deviates from the nominal payload configuration.	Swapped the nominal box for a known heavier or lighter one.	✓	×	×
Invalid gripping position	Timing error: the gripper closes after the arm has begun lifting, gripping the object off-centre.	Added a sleep() call in the controller program so gripper closure lags the lift motion.	✓	×	×
Unstable mounting platform	Base instability introduces low-frequency vibrations into the arm.	Placed 8 layers of towels under the base plate.	✓	×	×
Joint position limit violation	A joint moves outside its configured soft or hard position limits, triggering a safety stop.	Random waypoint slightly beyond the configured soft joint limit.	✓	×	×
TCP frame misconfiguration	TCP frame or mounting orientation misconfigured at the controller; gravity torques deviate.	Entered an incorrect TCP offset or mounting angle in the controller settings.	✓	✓	✓
Payload weight misconfiguration	Payload mass misconfigured while a tool or workpiece is physically attached.	Left configured payload weight at multiple wrong mass configurations.	✓	✓	×
External arm disturbance	A continuous external force pulls or pushes the TCP during motion.	Anchored an elastic rope between the bench frame and the tool flange.	✓	✓	✓
Mild payload CoG misconfig.	Payload CoG specified at an incorrect offset from the tool flange; gravity compensation is wrong.	Entered a CoG offset in the payload configuration that differs from physical CoG.	✓	✓	✓
Collision with hanging cable	A loose cable drags along a robot link or the tool flange during motion.	Suspended a loose cable across the trajectory at arm height.	✓	✓	✓
Collision with cardboard	A cardboard carton in the trajectory provides moderate resistance; may trigger a protective stop.	Placed a free-standing or braced cardboard box in the programmed trajectory.	✓	✓	✓
Collision with rigid object	A rigid plastic block in the TCP path triggers an immediate protective stop.	Clamped a rigid plastic block in the programmed TCP path.	✓	✓	✓
Peg insertion misalignment	Peg approaches the hole at an angular or lateral offset and jams against the rim.	Added a recorded lateral offset at the insertion approach waypoint.	×	×	✓
Hole obstruction	Foreign object or debris in the hole prevents full insertion.	Placed a small object (metal screw) inside the hole before insertion.	×	×	✓
Incorrect insertion depth	Insertion terminates at an incorrect Z depth due to a misconfigured waypoint.	Shifted the insertion endpoint waypoint Z by recorded offset from nominal.	×	×	✓
Peg surface contamination	Contamination or roughness on the peg raises insertion friction and produces stick-slip.	Applied dry chalk powder to the peg surface.	×	×	✓
Fixture displacement	Insertion fixture shifted laterally, breaking the match between programmed approach and actual hole.	Physically shifted the hole fixture by recorded offset without updating program.	×	×	✓
Self-collision	Arm configuration causes a link to contact the robot body or mounting fixture.	Random waypoint that forces a self-colliding configuration.	✓	×	×
Missing box	Box absent from the pick position; the gripper closes on empty air.	Removed the box from the pick position before the episode started.	✓	×	×
Missing peg	Arm descends into the hole empty-handed; insertion produces no contact force.	Removed the peg from the gripper before the episode started.	×	×	✓

Table 13: Valid episodes per arm and condition after the validity gate.

Arm	nominal	6	12	13	14	16	17	total
UR3	2,000	1,097	439	635	1,250	1,139	1,159	7,719
UR5	2,053	1,180	663	809	1,249	993	1,234	8,181
UR10	2,012	1,165	868	989	1,250	866	1,241	8,391
UR30	1,753	1,217	985	1,056	1,250	1,086	1,235	8,582
KR10	2,498	1,197	–	1,103	1,248	1,176	1,649	8,871
Total	10,316	5,856	2,955	4,592	6,247	5,260	6,518	41,744

Table 14: Per-category anomaly detection AUROC on voraus-AD. Unstructured baselines use 130 signals; S-E-F-C MLP uses only 24 signals. N = number of anomalous episodes per category. † 95% bootstrap CI over episodes, for the S-E-F-C MLP only. ‡ Baseline deviations are those reported by Brockmann et al. (2024) for each method and are not computed here; they are not comparable to the bootstrap interval, and the aggregate-row uncertainty in Table 3 is defined separately.

Fault Category	N	**1-NN**	**PCA**	**GANF**	**CAE**	**LSTM-VAE**	**HMM**	**MVT-Flow**	**S-E-F-C MLP (Ours)**
Additional friction	144	74.8	76.4	$88.5_{\pm 4.5}$	$89.4_{\pm 0.2}$	$88.7_{\pm 1.5}$	$88.0_{\pm 0.7}$	$96.6_{\pm 0.6}$	80.0
Miscommutation	89	80.8	87.0	$98.8_{\pm 1.1}$	$99.1_{\pm 0.0}$	$98.1_{\pm 0.5}$	$93.7_{\pm 1.3}$	$99.8_{\pm 0.3}$	99.2
Misgrip can	11	100.0	100.0	$47.6_{\pm 13.1}$	$100.0_{\pm 0.0}$	$100.0_{\pm 0.0}$	$71.0_{\pm 3.3}$	$95.3_{\pm 3.3}$	83.8
Losing can	74	68.7	70.1	$72.1_{\pm 5.8}$	$72.6_{\pm 0.3}$	$70.4_{\pm 2.9}$	$88.8_{\pm 1.0}$	$96.2_{\pm 0.4}$	71.8
Add. axis weight	156	75.0	79.2	$93.2_{\pm 2.4}$	$93.5_{\pm 0.1}$	$82.7_{\pm 1.1}$	$89.6_{\pm 1.2}$	$94.1_{\pm 0.7}$	**95.8**
Collision w/ foam	72	69.6	73.9	$81.2_{\pm 6.2}$	$81.5_{\pm 0.2}$	$81.5_{\pm 2.1}$	$89.8_{\pm 1.5}$	$87.5_{\pm 1.2}$	74.7
Collision w/ cables	48	74.5	75.7	$82.7_{\pm 6.4}$	$79.6_{\pm 0.3}$	$77.3_{\pm 3.6}$	$91.8_{\pm 1.4}$	$84.7_{\pm 1.2}$	67.6
Collision w/ cardboard	22	82.8	83.6	$77.5_{\pm 5.8}$	$78.6_{\pm 0.3}$	$82.8_{\pm 4.8}$	$86.3_{\pm 1.2}$	$88.3_{\pm 1.2}$	76.7
Varying can weight	80	63.7	64.2	$68.9_{\pm 8.7}$	$72.3_{\pm 0.3}$	$71.4_{\pm 2.1}$	$90.9_{\pm 2.0}$	$85.1_{\pm 1.1}$	76.2
Cable at robot	10	63.0	71.6	$76.6_{\pm 8.1}$	$83.1_{\pm 0.3}$	$96.0_{\pm 1.4}$	$84.4_{\pm 1.1}$	$100.0_{\pm 0.0}$	91.3
Invalid gripping pos.	12	93.4	92.1	$86.6_{\pm 10.8}$	$88.8_{\pm 0.3}$	$97.7_{\pm 1.3}$	$91.8_{\pm 1.4}$	$100.0_{\pm 0.0}$	89.1
Unstable platform	37	83.6	85.9	$84.6_{\pm 3.8}$	$83.9_{\pm 0.2}$	$93.6_{\pm 1.9}$	$82.5_{\pm 1.2}$	$96.1_{\pm 0.7}$	92.8
Mean	755	77.5	80.0	$79.9_{\pm 12.7}$‡	$85.2_{\pm 9.2}$‡	$86.7_{\pm 10.1}$‡	$87.4_{\pm 5.8}$‡	$93.6_{\pm 5.7}$‡	**83.2 [81.5, 86.1]**†

I TRANSFER STUDY: PROTOCOL AND SUPPORTING RESULTS

I.1 PROTOCOL

Data. Simulated pick-and-place episodes on five arms (UR3, UR5, UR10, UR30, KUKA KR10), each with the same ten-stage task program (`above_pick`, `descend`, `settle`, `close`, `lift`, `move_xy`, `lower`, `release`, `retreat`, `return`). Real targets are our laboratory UR3 and KR10 pick-and-place recordings; a *session* is one recording program and day on the UR3 and one recording source and day on the KR10.

Channels. Phase identification uses the six channels both real robots log: TCP position and velocity in a normalised task-space frame. Joint space transfers even worse between robots in simulation (UR3→KR10 phase macro-F1 0.177 in joint space vs 0.433 in task space; KR10→UR3 0.455 vs 0.536). Fault detection additionally uses per-joint tracking error (setpoint minus position) and motor current where both are available. Sim-to-sim cells use the same six TCP channels as the real cells. An earlier ten-channel sim set is not used: its simulated TCP "tracking error" was the distance to the current pick or place target (the commanded TCP position takes two values per episode), so it encoded phase and reach directly.

Models and metrics. Windows of 64 samples with stride 32 and a modal-label threshold of 0.6. MiniRocket + ridge (Dempster et al., 2021) is the main model; a logistic regression on summary statistics is the control. Phase is scored with macro-F1 over the ten stages (union of true and predicted labels) or with balanced accuracy over the seven stages that emit windows under this windowing (chance 0.143); every table names its scale. Faults are scored with AUROC and a 95% bootstrap interval, grouped by counterfactual baseline where one exists. Shuffled-label nulls use five permutations; we report their maximum. Five draws do not calibrate a null distribution, so throughout we read these maxima as exploratory diagnostics and not as significance thresholds; where a null run uses a different channel configuration from the result it accompanies, we say so at that point.

Table 15: Arm drive parameters (joints 1–6).

Arm	k_p (N m/rad)	Effort limit (N m)
UR3	6000, 6000, 4000, 800, 800, 600	56, 56, 28, 12, 12, 12
UR5	16071, 16071, 21429, 1867, 1867, 1400	150, 150, 150, 28, 28, 28
UR10	35357, 35357, 21429, 3733, 3733, 2800	330, 330, 150, 56, 56, 56
UR30	79071, 79071, 61857, 7133, 7133, 5350	738, 738, 433, 107, 107, 107
KR10	8000, 34000, 12000, 1900, 1900, 700	550, 650, 200, 35, 30, 20

Table 16: Real-data transfer cells not shown in Figures 3 and 4. Phase: macro-F1 (10 stages; "5 stages" = the five stages every real session emits). Faults: AUROC. Each real test set is one recording session; full cells in Section I.

Task	Train → test	UR3 target	KR10 target
Phase	real, same session (ceiling)	0.928	0.661
	real, same robot, other program/session (5 stages)	0.78–0.80	0.69–0.72
	real, other robot	0.227	0.145
	real, other robot (5 stages)	0.395	0.318
	sim, other manufacturer	0.377	0.366
Faults	real, same session (joint err. + current)	0.85–0.90	0.90
	speed-clean faults only (11, 30)	–	0.83 / 0.82
	real, other session, seen fault types	0.64–1.00	–
	real, other session, unseen fault types	0.41–0.53	–
	real, other robot (TCP)	0.51–0.67	0.70*

*Driven by speed-confounded faults; the speed-clean faults 11 and 30 score 0.45 and 0.67.

Splits. Real fault tests stay inside one two-class session and are split into halves A and B. For UR3 fault 12 in the case study (Section I.7) the halves hold 55/48 (A) and 54/51 (B) nominal/fault episodes; for the KR10 session with in-session nominals, 21 nominals per half.

I.2 Sim-to-sim transfer

Table 18 holds out one simulated arm at a time; Table 19 gives the pairwise matrix between single UR arms.

Within the UR family transfer holds; across manufacturers it does not. A held-out UR arm reaches 99–100% of its own phase ceiling (7-stage balanced accuracy) and 89–98% of its fault ceiling. A held-out KR10, trained on the four UR arms, reaches 82% of its phase ceiling (macro-F1 0.555) and 0.576 fault AUROC; that interval excludes 0.5 but lies within the range of our shuffled-label runs (0.46–0.64; one permutation per held-out arm, on a nine-channel variant). Pairwise, sim UR3→KR10 scores 0.433 phase macro-F1 and KR10→UR3 0.536 (within arm 0.995 / 0.984); fault AUROC 0.543 and 0.686. The simulated arms share one task program, but the simulated KR10's nominal cycle is about 1.6× longer than the UR3's (about 24 vs 15 s), so simulation does not separate embodiment from tempo.

Reach. Among UR arms, transfer falls with reach distance: $r = -0.87$ (faults) and -0.80 (phase) over the 12 ordered pairs. Across 20 matched-budget source pools, the reach distance of the pool's nearest member correlates with the pool's score ($r = -0.765$ for faults, -0.653 for phase; farthest member -0.274, -0.230). These analyses rest on four UR arms, so the 12 pairs are not independent; that reach stands in for arm size and dynamics in general is a hypothesis.

An honest simulated tracking error. Adding a simulated TCP tracking error computed by forward kinematics, $\mathrm{FK}(q_{\mathrm{cmd}}[k{+}1]) - \mathrm{FK}(q[k])$, raises within-arm fault AUROC (UR3 0.947→0.990, KR10 0.776→0.900) but lowers cross-arm AUROC by 0.04–0.19, and removes the reach relationship for faults ($r = -0.15$). A likely cause is that the error scale differs between arms (a hypothesis). On its own this channel gives 0.19–0.53 leave-one-arm-out phase macro-F1, so it does not leak phase.

Table 17: Batch sim-to-real gap over 1,155 paired episodes (pooled per-episode metrics).

Metric	Mean	Median	P10	P90
Joint RMSE (deg)	3.65	2.83	1.93	5.12
TCP position RMSE (mm)	13.16	13.26	8.87	16.92
EE L2 RMS (mm)	25.11	25.17	16.96	32.58
TCP rotvec RMSE (mrad)	1877	2605	8.75	3136
W1 effort mean (A)	0.83	0.81	0.77	0.91

Table 18: Leave-one-arm-out transfer in simulation (six TCP channels, MiniRocket, trained on the other four arms; logistic-regression control in brackets; 95% bootstrap interval below). "Ceiling" is the held-out arm trained on its own data. Test sets: 1,000 episodes (faults) and 2,059–2,131 windows (phase) per arm.

Held-out arm	Fault AUROC			Phase, 7-stage bal. acc.			Phase macro-F1
	LOO	ceiling	%	LOO	ceiling	%	LOO
UR3	0.842 [0.720] [0.814, 0.868]	0.947	89	0.994 [0.836] [0.986, 0.999]	0.998	100	0.881
UR5	0.913 [0.678] [0.896, 0.930]	0.936	98	0.996 [0.829] [0.992, 1.000]	0.997	100	0.919
UR10	0.906 [0.665] [0.878, 0.931]	0.940	96	0.986 [0.773] [0.980, 0.993]	0.996	99	0.912
UR30	0.939 [0.712] [0.923, 0.952]	0.967	97	0.996 [0.780] [0.994, 0.998]	0.992	100	0.929
KR10	0.576 [0.580] [0.539, 0.612]	0.776	74	0.812 [0.640] [0.800, 0.824]	0.992	82	0.555

Table 19: Transfer between single UR arms in simulation, six TCP channels (rows: trained on; columns: tested on). Fault AUROC / phase macro-F1. Reach: UR3 0.50 m, UR5 0.85 m, UR10 and UR30 1.30 m. Pearson r between the 12 off-diagonal scores and $|\log \text{reach ratio}|$: -0.87 (faults), -0.80 (phase).

Trained on ↓ / tested on →	UR3	UR5	UR10	UR30
UR3	0.947 / 0.995	0.764 / 0.897	0.605 / 0.680	0.602 / 0.689
UR5	0.782 / 0.880	0.936 / 0.970	0.825 / 0.865	0.835 / 0.855
UR10	0.776 / 0.767	0.849 / 0.777	0.940 / 0.800	0.948 / 0.915
UR30	0.554 / 0.719	0.731 / 0.875	0.887 / 0.848	0.967 / 0.940

I.3 Phase transfer between domains and robots

Table 20 reports phase identification by training domain and manufacturer at a matched budget, and Table 21 varies one processing choice at a time to diagnose the weak real cross-robot cell.

Label audit. On both robots each labelled stage onset precedes its physical event: `close` precedes the gripper closing by 0.08 s (UR3) / 0.26 s (KR10), `lift` and `retreat` precede the vertical rise by 0.14 / 0.44 s, and `release` precedes the gripper opening by 0.16 / 0.46 s; every lead is under 2% of the cycle. The vertical-velocity sign within each stage and the gripper width at each boundary match between the robots. The task programs differ: median nominal cycle 16.0 vs 26.5 s in the test sessions; pick-to-place distance 0.43 vs 0.93 m at the same vertical range (about 0.24 m); descend and lift speed 0.10 vs 0.04 m/s; a flat `move_xy` on the UR3 against an arc on the KR10; `return` takes 18.8% of the UR3 cycle and 7.7% of the KR10 cycle. The UR3 test session contains only five stages because `descend`, `settle`, `lift`, `lower` and `retreat` each last about 0.7 s, too short for a 64-sample window; the KR10 test session contains nine. That sim sources beat the other robot's real data may reflect that the four simulated UR arms span several scales and tempos while one real UR3 program does not (a hypothesis; untested on the five shared stages).

Table 20: Phase identification by training domain and manufacturer at a matched budget (∼2,445 windows, 6 channels). Macro-F1 / balanced accuracy (10-stage). Real tests are one session each; the real UR3 test session contains only 5 of the 10 stages, so compare within a column. Cells where the sim pool contains the target arm itself are within-arm references.

Trained on	→ real UR3	→ real KR10	→ sim UR3	→ sim KR10
sim UR (4 arms)	0.317 / 0.570	0.366 / 0.451	0.934 / 0.905	0.531 / 0.624
sim KR10	0.377 / 0.594	0.351 / 0.434	0.614 / 0.648	0.992 / 0.991
sim UR + KR10	0.411 / 0.650	0.407 / 0.497	–	–
real UR3	**0.928 / 0.929**	0.145 / 0.222	0.238 / 0.316	0.248 / 0.320
real KR10	0.227 / 0.348	**0.661 / 0.670**	0.195 / 0.182	0.261 / 0.371
real UR3 + KR10	0.921 / 0.920	0.641 / 0.645	–	–
sim + real UR3	0.918 / 0.918	0.387 / 0.447	–	–
sim + real KR10	0.221 / 0.386	0.625 / 0.636	–	–

Table 21: Diagnosing real cross-robot phase transfer. Each cell is macro-F1 over the union of labels / macro-F1 retrained and tested on the five stages every real session emits (`above_pick`, `close`, `move_xy`, `release`, `return`). Columns change one processing choice at a time.

Cell	Published	Derived vel., de-dup., tick clock	Tempo-free vel.	Raw time, 1.6 s window	Per-episode z-score
UR3→KR10	0.145 / 0.318	0.225 / 0.343	0.280 / 0.322	0.180 / 0.337	0.400 / 0.402
KR10→UR3	0.227 / 0.395	0.294 / 0.278	0.288 / 0.230	0.142 / 0.403	0.396 / 0.444
UR3 program A→B / B→A (5 stages)	0.779 / 0.797	same robot, different recording program			
KR10 session→session (5 stages)	0.69–0.72	same robot, different session			

Few-shot (exploratory). We also trained a CNN on simulation and fine-tuned it on k real UR3 episodes (6 channels, 7-stage balanced accuracy, pooled over sessions; ceiling 0.922 with all 502 real episodes), and compared it with training from scratch and with MiniRocket on pooled sim and real data. Each point is the mean of three random draws of k episodes, and every method sees the same draws at a given k. The resulting curves are not monotone in k (for example, accuracy at $k = 10$ is below $k = 5$ for several methods), which indicates that three draws do not average out episode-selection variance. We therefore do not report a saving in real episodes as a result. The CNN reached 90% of the ceiling at about 352 real episodes from scratch and about 177 after pretraining, where the curves are smoother; for MiniRocket the crossing lies on a noisy part of the curve and we do not interpret it.

I.4 Real-to-real fault detection by session

Table 22 lists the recording sessions these tests draw on.

Table 23 compares the original and full UR3 releases. The original analysis used one two-class UR3 pick-and-place session (program 0012). The full UR3 signal release raises this to three (program 0012 with 109/327 episodes and faults 11, 12, 29, 30; program 0010 day 1 with 377/30, faults 19 and 37; program 0010 day 2 with 677/982, faults 19, 22, 25, 28, 37), plus two two-class screwing sessions. Faults 8, 9, 10, 14, 15, 23, 31 and 38 remain without in-session nominals on the UR3. The KR10 data are unchanged.

Pooled training. Table 24 asks whether the small within-session training sets limit the real fault ceilings. They do not: pooling every other real episode of the same robot (regime b) moves the joint tracking-error + current ceiling by -0.10 to $+0.04$, and adding the other robot (regime c) costs up to a further 0.1. Most of the pooled episodes (77% on the UR3, 68% on the KR10) come from single-class sessions; this suggests they teach a program or session direction that the clean test penalises.

KR10 speed confound. In the KR10 source-5 session the fault episodes ran at a lower speed override than the nominals. Speed alone scores AUROC 0.73 (half B) and 0.78 (half A); per fault, 0.83 (fault 8), 0.96 (fault 10), 0.61 (fault 11) and 0.57 (fault 30). With joint tracking error and current, the speed-clean faults 11 and 30 score 0.83 and 0.82 under regime a and 0.58 and 0.73 under regime b.

Table 22: Real pick-and-place recording sessions (nominal / fault episodes, fault ids with counts where recorded, median cycle, log rate). Days are numbered from each robot's first recording day. Two-class sessions are the only ones that admit a within-session fault test. Counts are those of the session-fingerprint extraction; the full-release coverage count differs slightly for program 0010 days 1 and 2 (377/30 and 677/982 episodes with signals).

Robot	Session	Nom. / fault	Faults (id: n)	Cycle (s)	Log (Hz)
UR3	program 0010, day 1	345 / 29	19: 24, 37: 5	16.4	7.9
UR3	program 0010, day 2	633 / 944	22, 25, 28 ($\sim$300 each), 19, 37	15.4	7.9
UR3	program 0010, day 3	0 / 557	11, 22, 29, 30, 31, …	15.3	8.5
UR3	program 0010, day 4	0 / 1072	23: 468, 15: 247, 10: 182, 38: 142	15.7	8.5
UR3	program 0010, day 5	0 / 175	14: 175	14.8	8.5
UR3	program 0010, day 6	0 / 321	8: 181, 9: 131	11.5	49
UR3	program 0011, day 6	393 / 0	– (no grasp)	5.3	49
UR3	program 0012, day 6	109 / 327	11, 12, 29, 30	15.5–16.0	49
KR10	source 1, day 1	19 / 0	–	34.9	77.8
KR10	source 2, day 2	205 / 0	–	32.0	79.4
KR10	source 2, day 3	0 / 353	12: 217, 38: 102, 28: 34	32.2	79.2
KR10	source 4, day 4	0 / 242	28: 175, 11: 67	33.2	79.2
KR10	source 3, day 6	44 / 0	–	31.7	80.3
KR10	source 4, day 6	20 / 214	25: 214	33.9	79.3
KR10	source 5, day 7	42 / 288	10, 8, 11, 30	26.5 (nom.)	80.1

Table 23: Real fault detection before and after the full UR3 data (MiniRocket + ridge AUROC, mean of both halves, one session per test).

Analysis	Original data	Full data
Within session, joint err.+current	0.87 (program 0012)	0012: 0.87; 0010 day 2: 0.95; 0010 day 1: 1.00 (29 faults)
Within session, TCP only (6 ch.)	0.65 (0012)	0010 day 2: 0.70 (faults 25/28: 0.61; 19/22/37: 0.85–0.93); day 1: 1.00
Cross-session, same fault ids	not testable	faults 19/37: 0.64–1.00 per fault
Cross-session, unseen fault ids	not testable	chance: 0012 0.41–0.53; 0010 day 2 faults 22/25/28 0.45–0.74
Phase, 7-stage bal. acc.	0010: 0.937; 0012: 0.65	0010 day 2: 0.84 within; day 1: 0.66 within, 0.96 trained on day 2

Table 24: Fault AUROC with training regime a (within session: 216 UR3 / 166 KR10 episodes) / b (all other episodes of the same robot: 930 / 1263) / c (b plus all episodes of the other robot). Tests: UR3 program 0012 and KR10 source 5 session, halves B and A. Shuffled-label null maximum $\leq$0.61 (UR3), $\leq$0.74 (KR10, TCP) and $\leq$0.69 (KR10, joint).

Cell	UR3 half B	UR3 half A	KR10 half B	KR10 half A
Joint err.+current, all phases (12 ch.)	0.90 / 0.92 / 0.82	0.85 / 0.89 / 0.87	0.90 / 0.80 / 0.79	0.90 / 0.80 / 0.80
Current, all phases	0.82 / 0.84 / 0.73	0.80 / 0.82 / 0.78	0.89 / 0.75 / 0.73	0.91 / 0.81 / 0.75
Tracking error, carry	0.67 / 0.56 / 0.55	0.62 / 0.58 / 0.57	0.73 / 0.48 / 0.55	0.76 / 0.67 / 0.66
TCP, 6 ch.	0.64 / 0.58 / 0.55	0.66 / 0.56 / 0.54	0.80 / 0.74 / 0.74	0.79 / 0.72 / 0.74

I.5 Cross-robot fault transfer

Real UR↔KR10 fault transfer stays at chance even with all data pooled. With TCP channels, KR10→UR3 scores 0.51–0.67, inside the shuffled-label null ($\leq$0.77); UR3→KR10 scores 0.70, driven by the speed-confounded faults 8 and 10, while the speed-clean faults 11 and 30 score 0.45 and 0.67. With joint channels, pooled cross-robot cells score 0.52–0.62, at chance or inside the null (Table 25). In a matched-budget domain grid, the training domain (sim, real or both) explains 90% of the variance of fault AUROC on real targets and the manufacturer 0.4%.

I.6 Uncalibrated sim-to-real fault detection

Table 26 additionally asks whether feature-distribution alignment rescues any of these cells; it does not.

Table 25: Uncalibrated sim→real fault AUROC (MiniRocket, 95% bootstrap interval). Every interval contains 0.5. The 9-channel set adds TCP tracking error, which on the real UR3 is quantisation noise.

Trained on (sim)	Tested on	Channels	AUROC
KR10 + UR5 + UR10 + UR30	real UR3	9	0.583 [0.487, 0.685]
UR5 + UR10 + UR30	real UR3	9	0.480 [0.381, 0.585]
UR3 (same arm)	real UR3	9	0.491 [0.388, 0.597]
KR10	real UR3	9	0.493 [0.393, 0.603]
same four pools	real UR3	6	0.473 / 0.460 / 0.459 / 0.443
UR3 + UR5 + UR10 + UR30	real KR10	6	0.528 [0.457, 0.591]
KR10 (same arm)	real KR10	6	0.504 [0.440, 0.563]

Table 26: Domain adaptation for uncalibrated sim→real fault detection.

Method	Fault AUROC before → after	Did the alignment itself work?
Per-channel mean/sd matching	0.493 → 0.553	no (channels separable at 0.93–1.00)
CORAL (Sun et al., 2016)	0.501 → 0.519	yes (domain classifier 1.000 → 0.521)
DANN (Ganin et al., 2016)	0.546 → 0.548	no (encoder separable at 1.0)
Task-space representation	0.649 → 0.491	n/a

The reverse direction also fails: a real-trained fault model scores 0.335 on simulated UR3. Pooling simulation with 200 real episodes scores lower than the real episodes alone (0.583 vs 0.686); weighting both domains equally gives 0.681 vs 0.676 for real alone (not significant).

I.7 Calibrated simulation: one UR3 payload fault

Table 27 lists what was calibrated and on which data, and Table 28 the resulting tracking-error scores.

Target fault. In the real UR3 program 0012, fault 12 is a heavier box (1.2 kg in all fault episodes vs 0.6 kg in nominals) with the declared payload fixed at 1.5 kg in both classes. On real data alone, carry-phase tracking error detects it at 0.94 [0.91, 0.97] (grouped 5-fold; 0.89 in both half-to-half directions; null maximum 0.65), and motor current at 0.99–1.00.

Table 27: Calibration components of the simulator and the data each was fitted on. Only the timing-noise model was fitted strictly on the half not being scored.

Component	What it does	Fitted on
Actual box masses	0.6 kg nominal, 1.2 kg fault	real episode metadata (known design)
Fault onset at grasp	mass switches at the first `lift` tick	**real fault-12 episodes**
Declared-payload feedforward	controller compensates a fixed 1.5 kg	real episode metadata
Disturbance observer (load cancelling)	first-order, gain 1, τ 0.05–0.14 s per joint	**real fault-12 episodes (both halves)**
Command lag and acceleration term	lag 0.50–0.98 ms, 101–266 μs^2	109 real nominal episodes
Place-height offset	+10 mm at set-down	simulated episodes
Aligned logging	removes a one-tick command/position offset	simulated twins, real nominals
Logger emulation	replays the real logger's irregular sampling	real logger timing blocks
Timing-noise model	1-tick shifts + ∼0.4 ms jitter + quantisation	real nominals of the *other* half (strict)

Motor current. Across v2 and v3, regular and emulated logging, and both halves, sim-trained current detection scores 0.90–0.99 AUROC (e.g. v3, half B: carry 0.99 [0.97, 1.00], all phases 0.93 [0.88, 0.97]); the uncalibrated simulator scores 0.62 (carry) and 0.41 (all phases). The logistic-regression control on tracking error stays at 0.44–0.54 in every logging arm and 0.60 with the timing-noise model, which suggests that the tracking-error result rests on waveform shape rather than on signal level.

Caveats. (i) The disturbance observer and the logging diagnosis used all real fault-12 episodes, so neither half is fully untouched; swapping the halves is not an independent replication. (ii) Gripper dwell alone separates real fault from nominal episodes (AUROC 1.00); carry-phase cells exclude the dwell phases, and within each class the carry score barely tracks dwell (Spearman $+0.04$ nominal,

Table 28: Calibrated sim→real UR3 fault 12, carry-phase tracking error (MiniRocket AUROC; real half B / half A / pooled). v2 and v3 differ in the number of simulated fault episodes (100 vs 400, each with a nominal twin). Noise arms are means over 5 noise seeds. Pooled 95% intervals are about ±0.06–0.08.

Simulator / logging model	v2	v3
Uncalibrated simulator (half B)	0.40 [0.29, 0.51]	
Calibrated, regular logging	0.66 / 0.69 / 0.67	0.58 / 0.60 / 0.59
+ emulated real logger	0.72 / 0.71 / 0.71	0.68 / 0.65 / 0.66
+ white noise (strict)	0.72 / 0.68 / 0.68	0.78 / 0.73 / 0.73
+ timing-noise model (strict)	0.78 / 0.73 / 0.76	0.82 / 0.77 / 0.79 [0.72, 0.85]
Shuffled-label null maximum (half B)	0.56–0.67	
Real→real ceiling (grouped 5-fold)	0.94 [0.91, 0.97]	

+0.11 fault), but a class-linked recording confound cannot be fully excluded. (iii) The real→real signal survives the controls we could run: 0.87 on the 49 least dwell-mismatched pairs and 0.89 with nominals from the fault recording window. (iv) A load-aware linear model identified from real nominal episodes only, used as a generator, reaches 0.90 on real tracking error; this suggests the remaining sim gap lies in the load-transfer timing at lift-off and set-down rather than in the classifier (a hypothesis).

KR10 check. The only testable KR10 payload fault is fault 10 (an undeclared +1.4 kg between axes 4 and 5 for the whole episode; source-5 session, 42 nominals), and it is speed-confounded (speed alone 0.98). The UR3-calibrated simulator scores 0.52 [0.41, 0.63] on carry tracking error and 0.42 [0.31, 0.53] on carry current; a real UR3-trained model scores 0.12 and 0.41. The current shift has the opposite sign on the two robots. A post-hoc, label-free sign alignment on the three proximal joints gives 0.73 [0.64, 0.82]; because it was chosen after seeing the data, in one speed-confounded session, we do not treat it as confirmatory.

I.8 Session fingerprints

Table 29 asks how well a session can be identified from the signals alone, which bounds how much of any within-session result could be a recording effect.

Table 29: Session identification from signals only (grouped 5-fold, ≤200 episodes per session; balanced accuracy). *Full*: per-phase mean/std of 21 channels plus phase durations. *Timing*: phase durations and speed only. *Shape*: MiniRocket on per-episode z-scored, phase-normalised signals (no level, scale, duration or speed). The label-shuffle null is at chance in every row.

Episodes (number of sessions)	Chance	Full	Timing	Shape
UR3 nominal (4)	0.25	0.86	0.68	0.82
KR10 nominal (5)	0.20	0.98	0.79	0.95
UR3 fault (7)	0.14	0.85	0.62	0.81
KR10 fault (4)	0.25	0.85	0.55	0.85
UR3, faults 19 and 37 only, 4 days of program 0010	0.25	0.65	0.64	0.61

Sessions are identifiable at 3–5× chance, and the signature survives the removal of duration, speed and signal level. It lives in per-phase current and tracking-error levels and in waveform shape rather than in any single metadata-like variable; the exceptions are gross setup changes (a different program, a 49 Hz vs ∼8 Hz log rate, a different speed override, a different TCP configuration). Even the closest UR3 pair remains separable (AUROC 0.78–0.87 with all features). Any train/test split across sessions therefore tests session transfer as well as fault detection. Group labels alone predict the fault label: the UR3 recording program at 0.745 AUROC, the KR10 recording source at 0.818 and calendar day at 0.937, and, within UR3 program 0010, the recording day at 0.90 and the session at 0.91.