Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira,
Edoardo Patelli, Hélder S. Sousa, Rui A. Silva and Monica Santamaria
 Published by Research Publishing, Singapore.
doi: 10.3850/ESREL2026061419_esrel26-p26507-cd

A Physics-Informed Deep Learning Framework For Updating Digital Twins: Application On Robot Fault Diagnosis

Pu Huang

Laboratoire Génie Industriel, CentraleSupélec - Université Paris-Saclay, France.
E-mail: pu.huang@centralesupelec.fr

Xiaochun Liang

IDMC, Université de Lorraine, France. E-mail: xiaochun.liang9@etu.univ-lorraine.fr

Zhiguo Zeng

Chair on Risk and Resilience of Complex Systems, Laboratoire Génie Industriel, CentraleSupélec - Université Paris-Saclay, France. E-mail: zhiguo.zeng@centralesupelec.fr

Anne Barros

Chair on Risk and Resilience of Complex Systems, Laboratoire Génie Industriel, CentraleSupélec - Université Paris-Saclay, France. E-mail: anne.barros@centralesupelec.fr

Digital twins have been widely applied to generate training data for training AI models for industrial applications. Often, one suffers from the so-called "sim-to-real" gap problem, i.e., the digital twin simulation is not accurate enough which significantly affects the performance of the trained AI model for downstream tasks. Another critical issue in deploying digital twins is that the simulation requires numerical solvers that require a huge computational burden. In this work, we present a physics-informed deep learning framework to create surrogate modeling of digital twins and correct the prediction errors. We consider a particular scenario where observations can only be made on indirect output variables but not on the state variables, which directly reflect the effect of digital twin dynamics. The framework includes two key components: an Ideal Behavior Generator (IBG) and a Residual Generator (RG). The IBG is an MLP-based surrogate model that emulates the system nominal behavior with high precision and efficiency, while the RG employs a physics-informed Mixture Density Network (MDN) trained on limited real data to predict and correct sim-real-gaps by applying a probabilistic correction term on the unobserved state variables. To train the MDN, we only need observations on the output variables, not the state variables themselves. Sampling internal states from these distributions and integrating them with IBG outputs yields realistic behaviors that resemble real observations. The framework was validated using a robot arm fault diagnosis use case. Results show that classifiers trained on data generated by our framework achieved higher accuracy, notably improving the recognition of healthy trajectories (6% $\rightarrow$ 62%), while requiring only about 1% of the computation time of conventional simulations. Overall, the proposed framework provides an efficient, scalable, and physics-consistent approach for data-driven simulation and optimization of industrial systems, establishing a solid foundation for indirect-observation-variable-based health monitoring in industrial environments.

Keywords: Sim-to-real gap, Physics-informed deep learning, Surrogate modeling, Fault diagnosis, Digital twins.

1. Introduction

Digital Twins (DT) technology, characterized by its low-cost and high accuracy simulation of physical assets, has been widely applied to various industrial sectors (Tao et al., 2018; Grieves and Vickers, 2017). Its growth has reshaped the reliability assessment, condition monitoring, and predictive maintenance for complex systems, where samples for testing and operational data are scarce or expensive to acquire (Liu et al., 2021). Many classical reliability-oriented methods rely on large amounts of real data for statistical inference and parameter estimation. However, in practice, high-value equipment and systems often cannot support extensive samples for long-duration experiments, which leads to small-sample, rare-data condition. DT-based data generation has been widely used

to address this problem by augmenting training sets for downstream reliability-related applications (Xu et al., 2019; Booyse et al., 2020).

Although DT technology benefits mentioned research domains a lot, the sim-to-real gap between DT generated data and real data is a key barrier to DT's application (Rasheed et al., 2020). Reducing this gap requires adjusting DT parameters so that the simulation responses align with the actual behavior of the physical system. Over the past decade, Parameter Identification (PI) methods like Bayesian updating or Deterministic Optimization (DO) are widely used (Mottershead and Friswell, 1993; Kennedy and O'Hagan, 2001; Psaros et al., 2023). However, those existing approaches often face difficulty to deploy in time-critical or resource-constrained scenarios. First, BMU and DO methods can become computationally infeasible when the simulation process is time-consuming, and are not suitable for real-time updating. Second, many systems only provide partial, low-dimensional observations (e.g., indirect sensor outputs), while PI requires inferring high-dimensional latent parameters. This setting is ill-posed and suffers from practical non-identifiability (Tarantola, 2005). Third, as a traditional approach to address the computational issues with digital twins, data-driven surrogate models often learn average behaviors and do not clearly represent stochastic discrepancies induced by unmodeled dynamics, measurement noise, or hidden variables, which limits their ability to generate realistic training data for downstream tasks (Kapteyn et al., 2021).

To address these limitations, we propose a physics-informed deep learning approach for efficient and transferable DT construction and calibration under indirect observations. The core idea is to decompose the DT into two components in series: Ideal Behavior Generator (IBG) and Residual Generator (RG). An IBG is trained as a fast surrogate of the nominal DT, providing accurate predictions for the system's ideal behavior. A RG is then trained to model the discrepancy between real observations and IBG-generated predictions. By composing outputs of IBG and RG, a realistic simulation of system behavior can be obtained. While surrogate modeling, probabilistic discrepancy modeling, and MDNs are individually well established, the novelty of this work lies in combining them into a structured IBG–RG framework for digital twin updating under output-only observations, where direct state measurements are unavailable and the correction is performed through physically constrained, output-consistent discrepancy variables. This work mainly makes two contributions:

(i) We develop a computationally efficient pipeline to build high accuracy DT surrogates that are suitable for fast data generation and on-time updates.
(ii) We provide a general non-parametric approach for model updating under indirect and partial observations.

2. Problem Formulation

We consider a physical system driven by an input sequence $\boldsymbol{u} = \{u_t\}_{t=1}^{T}$. Let $\boldsymbol{y}_{\text{obs}} = \{y_t^{\text{obs}}\}_{t=1}^{T}$ denote the observed output sequence and $\boldsymbol{x} = \{x_t\}_{t=1}^{t}$ denotes the system's internal state. A digital twin is a parameterized simulator with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$. Given $\boldsymbol{u}$ and $\boldsymbol{\theta}$, the DT produces simulated outputs $\boldsymbol{y}_{\text{sim}}$ as

$$\begin{aligned} \boldsymbol{x}_{\text{sim}} &= F_{\text{sim}}(\boldsymbol{u}; \boldsymbol{\theta}), \\ \boldsymbol{y}_{\text{sim}} &= H\left(\boldsymbol{x}_{\text{sim}}, \boldsymbol{u}\right), \end{aligned} \tag{1}$$

where $F_{\text{sim}}(\cdot)$ denotes the high-fidelity dynamics solver of DT, and $H(\cdot)$ is the output operator that maps internal states to observable outputs.

DT model updating aims to adjust $\boldsymbol{\theta}$ such that the simulated outputs match the observations. We can get parameters $\boldsymbol{\theta}$ by solving a deterministic optimization problem (Mottershead and Friswell, 1993):

$$\boldsymbol{\theta}^{\star} \in \arg\min_{\boldsymbol{\theta}} \; \Delta\left(\boldsymbol{y}_{\text{obs}}, \boldsymbol{y}_{\text{sim}}(\boldsymbol{\theta})\right), \tag{2}$$

where $\Delta(\cdot)$ denotes a discrepancy measure between sim and real outputs. Or, the parameters $\boldsymbol{\theta}$ can also be solved via Bayesian updating (Kennedy and O'Hagan, 2001) by inferring the posterior

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta} \mid \boldsymbol{y}_{\text{obs}}, \boldsymbol{u}), \tag{3}$$

which quantifies the uncertainty in the updated parameters.

In practice, model updating is almost always performed under indirect observations, i.e., we can not get real latent state observations $\boldsymbol{x}_{\text{obs}}$. Under this situation, there are two critical limitations:

First, both optimization-based and Bayesian updating typically require repeated evaluations of $F_{\text{sim}}(\cdot)$ over many candidate $\boldsymbol{\theta}$, which can be computationally prohibitive and incompatible with frequent/online updates.

Second, when $H(\cdot)$ strongly compresses internal information, the mapping

$$\boldsymbol{\theta} \mapsto \boldsymbol{x} \mapsto \boldsymbol{y}_{\text{sim}}$$

may be non-injective, and the inverse problem is structurally underdetermined: multiple internal settings can induce the same outputs.

In this case, seeking a unique updated $\boldsymbol{\theta}$ is ill-posed and highly sensitive to initial point, while the practical objective in many DT model updating tasks is to generate outputs consistent with real observations. To address this ill-posed problem, we formulate updating as learning an efficient output-space correction that makes DT-generated outputs consistent with real observations, rather than inferring and calibrating physically "true" internal states $\boldsymbol{x}$. Section 3 provides in-depth details about this output-space correction approach.

3. Method

3.1. *Overview*

Following the formulation introduced in Section 2, we propose a PIML-based model updating framework that corrects DT outputs in output-space rather than inferring true internal states $\boldsymbol{x}$ of the system. Moreover, this framework improves the efficiency of the data generation process.

The key design of this framework is a "nominal–plus-residual" hierarchy. The system response is decomposed into a nominal component and a stochastic discrepancy:

$$\begin{aligned} \hat{\boldsymbol{y}}_{\text{IBG}} &= G_{\boldsymbol{\phi}}(\boldsymbol{u}) \approx \hat{\boldsymbol{y}}_{\text{DT}}, \\ \boldsymbol{z} &\sim q_{\boldsymbol{\psi}}\left(\boldsymbol{z} \mid \boldsymbol{u}, \hat{\boldsymbol{y}}_{\text{IBG}}\right), \\ \tilde{\boldsymbol{y}} &= g\left(\hat{\boldsymbol{y}}_{\text{IBG}}, \boldsymbol{z}\right). \end{aligned} \tag{4}$$

where $\hat{\boldsymbol{y}} = \{\hat{y}_t\}_{t=1}^{T}$ is the ideal (nominal) output of the system, $\boldsymbol{z} = \{z_t\}_{t=1}^{T}$ is system-level latent discrepancy, $\tilde{\boldsymbol{y}} = \{\tilde{y}_t\}_{t=1}^{T}$ is the simulated output of the system, and:

- **Ideal Behavior Generator (IBG, $G_{\boldsymbol{\phi}}$):** A fast deterministic surrogate (e.g., LSTM) approximating the ideal prediction of systems generated by DT.
- **Residual Generator (RG, $q_{\boldsymbol{\psi}}$):** A probabilistic model (MDN) capturing the distribution of system-level latent discrepancies $\boldsymbol{z}$.
- **Correction Operator (g):** A structured function mapping latent discrepancies to the output space. After applying $g(\cdot)$, the output of the DT is corrected to align with real observation, i.e. $\tilde{\boldsymbol{y}} \approx \boldsymbol{y}_{\text{obs}}$.

It should be noted that we introduce $\boldsymbol{z}$ to address the unobservable problem of $\boldsymbol{x}$, as $\boldsymbol{z}$ can be inferred from observations. In this paper, $\boldsymbol{z}$ is not interpreted as the true physical state itself, but as an output-relevant latent discrepancy variable that parameterizes the correction from nominal DT outputs to real observations.

Training Strategy: The learning problem is to estimate $(\boldsymbol{\phi}, \boldsymbol{\psi})$. Although an end-to-end optimization is conceptually appealing, the end-to-end approach with scarce real data may cause the IBG to absorb discrepancy effects, reducing interpretability and increasing overfitting risk. Therefore, we train the IBG and the RG, as detailed in Section 3.2 and 3.3, respectively.

3.2. *Ideal Behavior Generator*

The IBG is a machine learning based surrogate (e.g., LSTM) model $G_{\boldsymbol{\phi}}$ which is trained to approximate the nominal (ideal) system outputs in low time-cost. The IBG is trained on the simulated dataset $\mathcal{D}_{\text{sim}} = \{(\boldsymbol{u}, \hat{\boldsymbol{y}}_{\text{DT}}, \boldsymbol{y}_{\text{sim}})\}$ by minimizing a supervised loss such as

$$\mathcal{L}_{\text{IBG}}(\boldsymbol{\phi}) = \sum ||G_{\boldsymbol{\phi}}(\boldsymbol{u}) - \hat{\boldsymbol{y}}_{\text{DT}}||_2^2 . \tag{5}$$

After training, the IBG replaces the numerical solver, accelerating downstream calibration and data generation.

3.3. *Residual Generator*

The outputs of the IBG differ from those of the real system, and the RG outputs the distributions of the system-level state discrepancy $\boldsymbol{z} \sim q_{\boldsymbol{\psi}}(\boldsymbol{z} \mid \boldsymbol{u}, \hat{\boldsymbol{y}})$ that captures multi-modality and heteroscedasticity in sim-to-real gaps. The RG is trained on the observed dataset $\mathcal{D}_{\text{real}} = \{(\boldsymbol{u}, \hat{\boldsymbol{y}}_{\text{IBG}}, \boldsymbol{y}_{\text{obs}})\}$.

To describe and parameterize $q_{\boldsymbol{\psi}}$, we introduce a Mixture Density Network (MDN). The MDN predicts the parameters of a K-component Gaussian mixture, and can be trained to predict distributions with multi-modality and heteroscedasticity. For each timestamp t:

$$q_{\boldsymbol{\psi}}(\cdot) = \sum_{k=1}^{K} \pi_{t,k} \mathcal{N}(\cdot \mid \mu_{t,k}, \Sigma_{t,k}), \quad (6)$$

where $\pi_{t,k} \geq 0$ and $\sum_{k=1}^{K} \pi_{t,k} = 1$. Here, $\mu_{t,k}$ and $\Sigma_{t,k}$ denote the mean vector and covariance matrix of the k-th component at time t.

The major objective of MDN Training is minimizing the conditional negative log-likelihood (NLL) of $z_t \sim q_{\boldsymbol{\psi}}$:

$$\mathcal{L}_{\text{NLL}}(\boldsymbol{\psi}) = -\sum_{i=1}^{N} \sum_{t=1}^{T} \log q_{\boldsymbol{\psi}} \left(z_t^{*(i)} \right). \quad (7)$$

To encourage physically plausible discrepancies, we add regularization that encodes generic system-level constraints on $\boldsymbol{z}$. Let the MDN mixture mean at time t be

$$\bar{z}_t = \sum_{k=1}^{K} \pi_{t,k} \mu_{t,k}, \quad (8)$$

we impose (i) temporal smoothness and (ii) boundedness:

$$\begin{aligned} \mathcal{R}_{\text{s}} &= \sum_{t=2}^{T} \|\bar{z}_t - \bar{z}_{t-1}\|_2^2, \\ \mathcal{R}_{\text{b}} &= \sum_{t=1}^{T} \|\text{ReLU}(|\bar{z}_t| - b)\|_2^2, \end{aligned} \quad (9)$$

where b is a vector of admissible bounds determined from engineering knowledge or data ranges.

The final training objective for the residual model is

$$\mathcal{L}_{\text{RG}}(\boldsymbol{\psi}) = \mathcal{L}_{\text{NLL}}(\boldsymbol{\psi}) + \lambda_s \mathcal{R}_{\text{s}} + \lambda_b \mathcal{R}_{\text{b}}, \quad (10)$$

where λ_s and λ_b balance statistical fit and physical plausibility.

4. Case Study

4.1. *Background and setting*

4.1.1. *Physical system and Digital Twin*

The experimental platform is a *Hiwonder ArmPi FPV*, a 6-DOF robot arm (Fig 1). To simulate a constrained industrial monitoring scenario, we focus on the four primary actuators (ID: Motor 3–6) driven by command $u_t \in \mathbb{R}^4$. The only observable output is the end-effector 3D position $y_t^{\text{obs}} \in \mathbb{R}^3$, measured at 100 Hz. Internal states (e.g., joint encoder angles) are strictly hidden. The nominal Digital Twin is built in MATLAB/Simulink using Simscape Multibody, based on the robot's URDF kinematic parameters (Mc Court et al., 2024). While the DT captures ideal rigid-body dynamics, it lacks the modeling of communication delays, gear backlash, and servo dead-zones inherent in the physical control loop, creating a significant sim-to-real gap.

4.1.2. *Fault definition and injection protocol*

We consider a 9-class diagnosis task (1 Healthy + 8 Faulty). Faults are injected into the physical robot via Hardware-in-the-Loop (HIL) control software to mimic hardware degradation:

(i) **Motor Stuck:** Freezing the command of the i-th motor to mimic seizure;
(ii) **Motor Steady-State Error (SSE):** Adding a constant bias which exceeds the tolerance to the i-th motor's reference.

4.2. *IBG and RG implementation*

4.2.1. *Datasets*

We have two datasets for IBG/RG training:

- **DT-simulated dataset A:** A DT-generated dataset $\mathcal{D}_{\text{sim}}^{A} = \{(\boldsymbol{u}, \hat{\boldsymbol{y}}_{\text{DT}}, \boldsymbol{y}_{\text{sim}})\}$ with 3600 trajectories (400 trajectories per class for 9 classes).

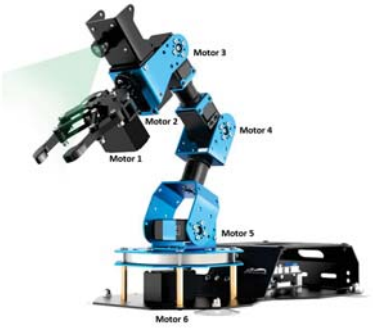


Fig. 1. Schematic of the robot arm

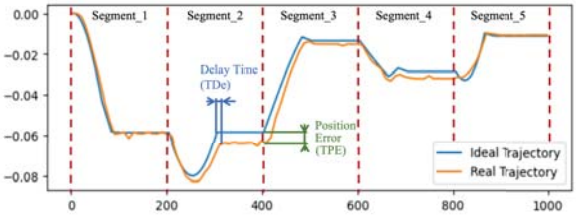


Fig. 2. An example of the projection of the terminal trajectory onto the x-axis, illustrating segmentation, TDe, and TPE.

- **Real dataset A:** A real observation dataset $\mathcal{D}^{A}_{\text{real}} = \{(\boldsymbol{u}, \hat{\boldsymbol{y}}_{\text{IBG}}, \boldsymbol{y}_{\text{obs}})\}$ with 90 trajectories (10 trajectories per class for 9 classes).

4.2.2. *IBG implementation*

The IBG is trained on $\hat{\boldsymbol{y}}_{\text{DT}} \in \mathcal{D}^{A}_{\text{sim}}$ to approximate the nominal input–output mapping $G_{\boldsymbol{\phi}}$. In our implementation, $G_{\boldsymbol{\phi}}$ is a long short-term memory (LSTM) model, mapping $u_t \mapsto \hat{y}_t$. This design is intentionally simple to maximize speed and stability.

4.2.3. *RG implementation*

In this case study, z is instantiated as a segment-level discrepancy variable composed of a delay term and a 3D position-error term, i.e., $z = (\text{TDe}, \text{TPE})$ (Fig. 2). The RG uses a lightweight shared encoder with two heads: a 1D Gaussian mixture for TDe and a 3D Gaussian mixture for TPE with full covariance. Dropout, validation monitoring, learning-rate decay, and early stopping are used to reduce overfitting.

4.2.4. *Correction operator implementation*

Given $\hat{\boldsymbol{y}}$ and a sampled $\boldsymbol{z}$ for each segment, the correction operator $g(\cdot)$ produces the corrected trajectory $\tilde{\boldsymbol{y}}$.

For each motion segment, we shift the segment by TDe timestamps. For each steady segment, we add the 3D offset TPE to the steady position. We use interpolation at segment boundaries to avoid discontinuities.

4.3. *Validation via fault diagnosis task*

We validate the corrected data generator (IBG + RG) by training the same downstream fault diagnosis model under different data sources and testing on real observations.

4.3.1. *Datasets*

We have three datasets for fault diagnosis task:

- **DT-simulated dataset B:** A DT-generated dataset $\mathcal{D}^{B}_{\text{sim}}$ that is *different* from $\mathcal{D}^{A}_{\text{sim}}$
- **IBG-RG-simulated dataset:** The IBG-RG-simulated dataset $\mathcal{D}^{\text{IR}}_{\text{sim}} = \{(\boldsymbol{u}, \hat{\boldsymbol{y}}_{\text{IBG}}, \tilde{\boldsymbol{y}})\}$ that matches the DT dataset size and class balance.
- **Real dataset B:** A real dataset $\mathcal{D}^{B}_{\text{real}}$ that is *different* from $\mathcal{D}^{A}_{\text{real}}$

4.3.2. *Fault classifier implementation*

We implement the fault classifier as a LSTM model. The classifier input is the concatenation of the nominal trajectory and the trajectory residual:

$$\begin{aligned} r_{1:T} &= \tilde{\boldsymbol{y}} - \hat{\boldsymbol{y}} \quad \text{or} \quad \boldsymbol{y}_{\text{obs}} - \hat{\boldsymbol{y}} \\ \boldsymbol{x} &= \text{concat}(\hat{\boldsymbol{y}}, r_{1:T}) \in \mathbb{R}^{6\times T}, \end{aligned} \tag{11}$$

and the LSTM maps $\boldsymbol{x}$ to a 9-way prediction.

4.3.3. *Experimental design*

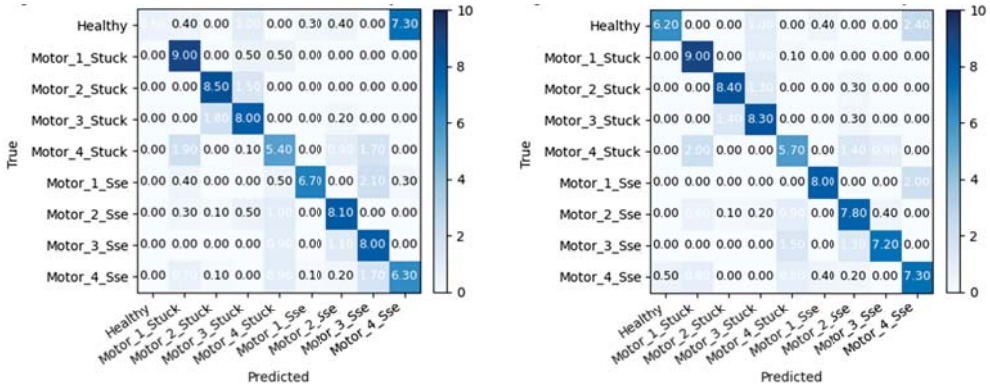
We compare two training data sources $\mathcal{D}^{B}_{\text{sim}}$ and $\mathcal{D}^{\text{IR}}_{\text{sim}}$ while keeping the fault classifier architecture, the training protocol, and the real test set fixed. For both sources, we use a $0.8/0.2$ split of the 3600 trajectories for training/validation.

To reduce randomness, we repeat the full training and evaluation procedure with 10 different

Table 1. Fault diagnosis metrics over 10 runs.

Training source / Statistic value	Macro-F1 (mean±std)	Balanced Acc. (mean±std)	AUROC (mean±std)	Healthy Recall (mean)
IBG+RG	$\mathbf{0.753 \pm 0.026}$	$\mathbf{0.754 \pm 0.025}$	$\mathbf{0.945 \pm 0.006}$	$\mathbf{62\%}$
DT baseline	0.644 ± 0.036	0.673 ± 0.036	0.931 ± 0.008	6%
Mean diff.	0.1100	0.0811	0.0139	–
t (df)	7.860 (16.55)	5.868 (15.98)	4.392 (16.86)	–
Holm-adjusted p	< 0.001	< 0.001	0.0004	–

Note: For each metric, the null hypothesis is that the mean value under IBG+RG is equal to that under the DT baseline. Welch's t-test was used for each metric, with Holm correction for multiple comparisons.



(a) DT baseline (b) IBG+RG

Fig. 3. Confusion matrices on the real test set $\mathcal{D}^B_{\text{obs}}$. Left: DT baseline; Right: IBG+RG.

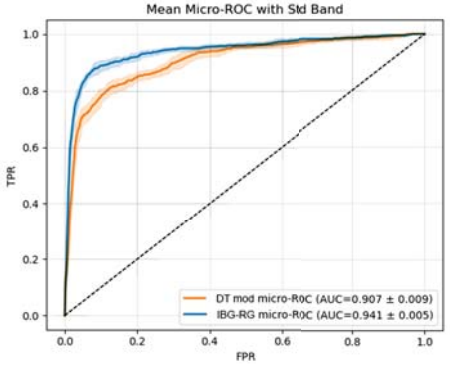


Fig. 4. ROC curves on $\mathcal{D}^B_{\text{obs}}$ for the DT baseline and the IBG+RG generator (mean over 10 runs; shading indicates ± one standard deviation).

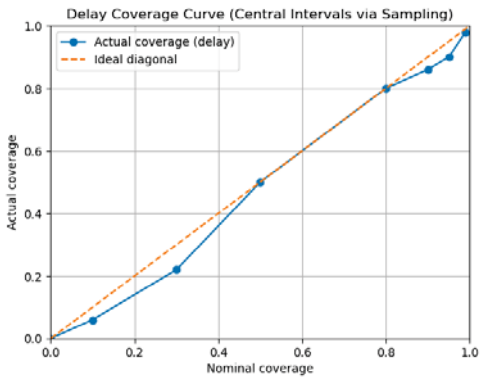
random seeds. In each run, we (i) generate the $\mathcal{D}^B_{\text{sim}}$ and $\mathcal{D}^B_{\text{real}}$, (ii) train the LSTM classifier, and (iii) evaluate on $\mathcal{D}^B_{\text{real}}$.

5. Results and discussion

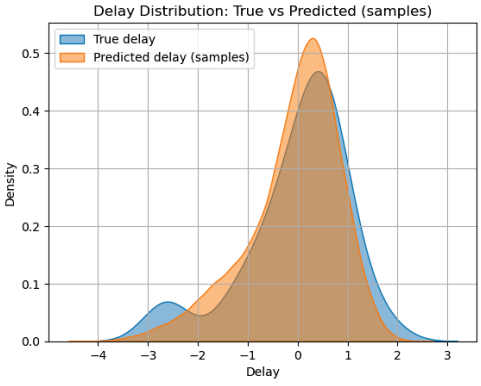
We summarize the diagnostic performance under the two training sources (DT baseline vs. IBG+RG generator) and analyze the effect of output-consistent correction on sim-to-real generalization.

Time cost of data generation. Generating 3600 trajectories (400 per class) with the original DT requires about 8.5 hours on an Apple MacBook Pro (M1 Pro). In contrast, the surrogate-based generator (IBG+RG) produces the same scale of trajectories within about 4–5 minutes on the same platform.

Overall performance. Table 1 shows the mean

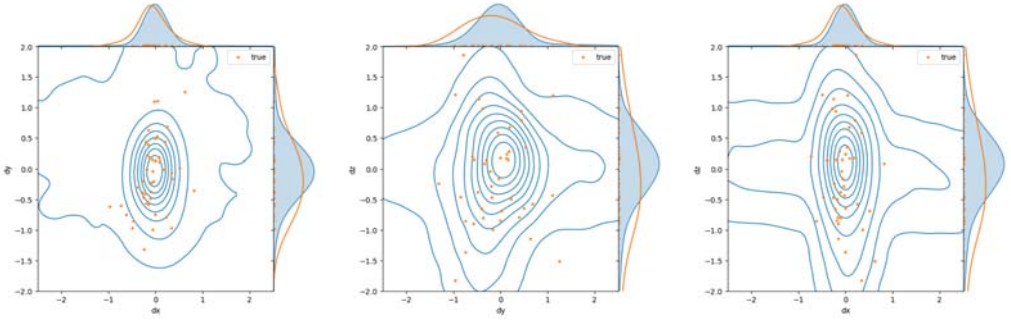


(a) Coverage Curve



(b) Distribution visualization

Fig. 5. Comparison of TDe distributions: observed vs. RG-predicted.



(a) X-Y plane (b) Y-Z plane (c) X-Z plane

Fig. 6. Comparison of TPE distributions: observed vs. RG-predicted.

and standard deviation (over 10 runs) of AUROC, macro-F1, and balanced accuracy on the real test dataset $\mathcal{D}^{B}_{\text{real}}$. Overall, training on IBG+RG generated trajectories yields consistently better generalization than training on DT-only simulated trajectories. We test the null hypothesis that the mean value of each primary metric is the same for the two training sources. Run-level Welch's t-tests with Holm correction further confirm that the improvements in all primary metrics are highly significant ($p < 0.01$).

Class-wise behavior and healthy recall. Figure 3 shows the confusion matrices for the two training sources. We highlight the Healthy class recall in Table 1, as false alarms are particularly costly. The IBG+RG generator substantially improves the recognition of healthy trajectories, while maintaining competitive discrimination among fault classes.

ROC analysis. To assess robustness across decision thresholds, Figure 4 presents the micro-ROC curves (mean and variability across runs). The IBG+RG generator leads to higher micro-ROC, suggesting that the corrected generator provides more reliable class separation on real observations.

Distributional analysis of discrepancy variables. Figure 5 and Figure 6 show that the learned MDN captures the observed variability of TDe/TPE rather than only a mean correction. The results indicate that the learned discrepancy model provides a realistic stochastic correction that is

consistent with observations.

Discussion. These results support two conclusions. First, output-level discrepancy modeling with a structured correction operator improves sim-to-real generalization under output-only sensing. The strong gain in Healthy recall suggests that the nominal DT does not reproduce the normal delay and position-error patterns of the real robot well, which makes healthy trajectories more likely to be confused with fault classes. The IBG+RG generator reduces false alarms while preserving competitive discrimination among fault classes. Second, the approach is computationally efficient: it replaces repeated calls to a high-fidelity DT solver with a fast surrogate and a stochastic correction model, enabling large-scale data generation on practical time scales.

6. Conclusion

This paper presents a physics-informed machine learning framework for efficient digital twin model updating under output-only observations. The method follows a nominal-plus-residual hierarchy: a nominal surrogate (IBG) learns a fast input-output mapping from DT simulations, and a discrepancy model (RG) learns a conditional distribution of output-relevant latent discrepancies from scarce real measurements. By sampling discrepancies and applying a structured correction operator, the corrected generator produces trajectories that are statistically consistent with real observations without repeatedly running the high-fidelity solver.

In the robot arm case study, training a fault diagnosis model on IBG+RG-generated trajectories improves generalization to held-out real measurements, with a clear gain in healthy-class recognition, while reducing data-generation time from hours to minutes. The formulation targets settings with limited sensor coverage and output-only observations. Future work will study joint fine-tuning with output-level consistency objectives and extend the correction structure to broader classes of discrepancy processes.

Acknowledgement

This work has received funding from the European Union's Horizon Europe research and innovation programme under the Marie Skłodowska-Curie Actions Doctoral Networks (MSCA-DN) grant No. 101120366 (Training42Phase). Additionally, Anne Barros and Zhiguo Zeng are partially supported by Chair of Risk and Resilience of Complex Systems (EDF, Natran, Orange, RTE and SNCF). Zhiguo Zeng also thanks the funding from ANR under grant No. ANR-22-CE10-0004.

References

Booyse, W. P., D. N. Wilke, and S. Heyns (2020). Deep digital twins for detection, diagnosis and prognostics. *Mechanical Systems and Signal Processing 140*, 106612.

Grieves, M. and J. Vickers (2017). *Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems*. Springer.

Kapteyn, M. G., J. V. Pretorius, and K. E. Willcox (2021). A probabilistic graphical model foundation for enabling predictive digital twins. *Nature Computational Science 1*(5), 337–347.

Kennedy, M. C. and A. O'Hagan (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63*(3), 425–464.

Liu, M., S. Fang, H. Dong, and C. Xu (2021). Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems 58*, 346–361.

Mc Court, K., X. Mc Court, S. Du, and Z. Zeng (2024). Use digital twins to support fault diagnosis from system-level condition-monitoring data. *arXiv preprint arXiv:2411.01360*.

Mottershead, J. E. and M. I. Friswell (1993). Model updating in structural dynamics: a survey. *Journal of sound and vibration 167*(2), 347–375.

Psaros, A. F., K. Kawaguchi, and G. E. Karniadakis (2023). Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *Journal of Computational Physics 477*, 111902.

Rasheed, A., O. San, and T. Kvamsdal (2020). Digital twin: Values, challenges and enablers from a modeling perspective. *IEEE Access 8*, 21980–22012.

Tao, F., H. Zhang, A. Liu, and A. Y. Nee (2018). Digital twin in industry: State-of-the-art. *IEEE Transactions on Industrial Informatics 15*(4), 2405–2415.

Tarantola, A. (2005). *Inverse problem theory and methods for model parameter estimation*. SIAM.

Xu, Y., Y. Sun, X. Liu, and Y. Zheng (2019). Digital-twin-driven intelligent fault diagnosis of rolling bearing. *IEEE Transactions on Instrumentation and Measurement 68*(11), 4596–4604.