

Article

Virtual Sensors Based on Finite Element Method: Balancing Accuracy, Runtime and Offline Effort

Andreas Kormann , Tobias Rosnitschek , Stephan Tremmel  and Frank Rieg

Engineering Design and CAD, University of Bayreuth, Universitätsstraße 30, 95447 Bayreuth, Germany; stephan.tremmel@uni-bayreuth.de (S.T.)

* Correspondence: andreas.kormann@uni-bayreuth.de

Abstract

Access to internal fields such as stress, temperature, and fatigue indicators is essential for condition monitoring, yet direct sensing is often impractical. Finite element method (FEM)-based virtual sensors address this gap by combining sparse measurements with physics-based models. This work compares two virtual sensor workflows. The live FEM approach executes a model on demand and provides high-fidelity estimates at the cost of multi-second runtimes. The lookup database approach shifts computation offline by precomputing responses and answering online queries by fast interpolation. We introduce a quantitative cost model that links measured runtime scaling, offline construction effort, and online latency to deployment choices. The cost model is evaluated through timing studies, accuracy assessments, and an empirical break-even analysis relating offline effort to the expected number of online queries. Two case studies illustrate the method, a nonlinear tension-bar benchmark and a steady-state thermal model of a CPU die. Live FEM runtime follows a power law with $\alpha \approx 1.2$ for the tensile case and an effective $\alpha \approx 0.66$ for the CPU case due to dominant overheads. The resulting rules translate accuracy targets and latency budgets into workflow-selection criteria that support integration into digital-twin and monitoring pipelines.

Keywords: virtual sensors; finite element method; real-time simulation; state estimation; design of experiments; lookup databases; thermal analysis; nonlinear mechanics

1. Introduction

In modern engineering, the ability to monitor, predict, and optimize the behavior of complex systems often hinges on access to internal quantities such as stress fields, temperature distributions, or condition indicators. Direct sensing is frequently limited, because equipping assets with sensors can be cost-intensive, technically infeasible under spatial or environmental constraints, or unreliable over time [1].

A well-established way to mitigate this deficit is the virtual sensor, understood as a software layer that produces indirect measurements by combining and aggregating signals from available sensors via a data fusion function [2]. Operating on the principle of model-based state estimation or sensor virtualization, they utilize algorithms to infer quantities that are not directly measured. The effectiveness of this approach has been demonstrated in various industrial domains. For instance, in electric vehicle battery management, electrochemical models estimate the internal state of charge based on terminal voltage and current to ensure safe operation limits [3], while in structural health monitoring, vibration analysis based on accelerometer data is used to detect damage in civil infrastructure before visible



Academic Editors: Mirosław Szczepanik and Arkadiusz Poteralski

Received: 28 January 2026

Revised: 13 February 2026

Accepted: 16 February 2026

Published: 19 February 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

failure occurs [4]. Virtual sensors are widely used in control and monitoring, particularly when measurements are scarce, and required variables cannot be measured directly [5].

In this work, the term simulation-based virtual sensor (SVS) denotes a virtual sensor that invokes a physics-based numerical model during operation. Three realizations are commonly distinguished. First, SVS-LFE relies on live FEM (LFE), executing a finite element (FE) model on demand to map inputs and measurements to quantities of interest. Second, SVS-LDB relies on a lookup database (LDB) and interpolates from an offline database of precomputed FE responses, shifting most computational effort to preparation and enabling fast online queries. Third, SVS-ML uses data-driven surrogates trained on simulated data. This variant is relevant in broader digital twin practice but is not benchmarked here.

Model-based sensing is central to the digital twin (DT) paradigm, commonly described as a virtual entity that mirrors its physical counterpart, with increasing functionality as the integration between physical and digital entities becomes more bidirectional and automated [6,7]. DT research also emphasizes the need for robust modeling and continuous use of operational data to support monitoring and decision-making [8]. Recent reports further emphasize that surrogate modeling is a foundational requirement for digital twins to bridge the gap between high-fidelity simulation and real-time decision making [9].

However, reliance on high-fidelity FE models exposes a recurring dilemma between fidelity and latency. Typical use of FE programs implies offline computations with a clear separation between model setup, solving, and evaluation, whereas real-time use requires substantially higher computational efficiency [10]. In DT settings, real-time performance is often limited by latency in data processing and synchronization, and finer meshes can further increase computational demand, reinforcing the accuracy–latency trade-off [11].

This exposes a significant problematic area in current research: the lack of quantitative decision support for selecting the appropriate execution architecture. While the general trade-off between reduced-order models and full simulations is well-studied, a distinct challenge arises when high spatial fidelity is non-negotiable. Here, engineers must choose between two competing high-fidelity strategies: live FEM (computing on demand) and lookup databases (precomputing the solution space).

The scientific gap lies in the absence of a unified cost model that accounts for the opposing scaling laws of these strategies. Currently, there are no established guidelines to determine the economic and computational break-even points between these workflows under specific accuracy and latency constraints. This creates a technological gap where engineers select workflows based on intuition rather than empirical data.

The objective of this paper is to develop and validate a decision-making framework for selecting between SVS-LFE and SVS-LDB under accuracy and latency constraints. The investigation is guided by two research questions. RQ1 asks how live FEM runtime and prediction error scale with mesh resolution. RQ2 asks, for a given accuracy target and problem dimension, how the computational break-even between an LFE-based virtual sensor and an LDB-based sensor depends on the offline design size N and the expected number of online queries Q .

This work makes three contributions. First, it introduces a unified cost model for simulation-based virtual sensors that separates offline precomputation, online latency, and measurement cadence and applies it to two representative case studies. Second, it compares three design of experiments strategies, Sobol sampling, Latin hypercube sampling, and orthogonal arrays, for constructing lookup databases in up to nine dimensions and assesses their impact on surrogate accuracy and robustness. Third, it combines the cost models with measured runtimes and errors to obtain empirical scaling laws and break-even points between live FEM simulation and lookup database surrogates.

2. Materials and Methods

This section formalizes the SVS framework and describes the methods used in this study. We introduce the notation, latency and error models, performance metrics, and cost models, and we briefly summarize the three primary implementation strategies found in the literature, which we classify as SVS-LFE, SVS-LDB, and SVS-ML. We then describe the case-study setups, the design of experiments (DoE) strategies and interpolation schemes used to construct the lookup database, the computational environment and logging, and the statistical validation pipeline. Finally, we define generic cost models for both workflows. These models are later combined with the case-study measurements in Section 3.3 to obtain empirical scaling laws and break-even points.

2.1. SVS Variants

In this work, we focus on SVS-LFE and SVS-LDB and treat SVS-ML as relevant context rather than a benchmarked alternative.

The first variant, SVS-LFE (or live FEM), executes a full or simplified finite element (FE) simulation on demand for each new input state streamed from the asset. Research in this area often targets runtime reduction through algorithmic improvements and hardware acceleration, for example GPU-enabled solvers [10,12]. The live FEM approach is conceptually straightforward and limited to FE discretization errors. The online cost increases strongly with the model size, and practical second-level update rates are typically obtained through physics or geometry simplifications or aggressive coarsening, which can shift the dominant error toward modeling assumptions.

The second variant, SVS-LDB, separates offline and online computation by precomputing high-fidelity FE responses over a predefined parameter space. During operation, outputs are obtained by fast interpolation of the lookup database, for example using *k*-nearest neighbors or radial basis functions [13]. By precomputing responses and answering online queries via fast interpolation, this approach obviates the need for real-time solvers while preserving the predictive accuracy of the underlying physics model [14]. This approach enables millisecond-level responsiveness and has been applied in accelerated engineering workflows such as wind turbine gearbox optimization [15]. Its main limitation is the curse of dimensionality, as offline simulation effort and storage grow rapidly with the number of input parameters, and the interpolation accuracy can degrade near domain boundaries or under extrapolation.

The third variant, SVS-ML, includes reduced-order and data-driven surrogates that approximate FE behavior at low online cost [10,16,17]. While recent studies report low errors and substantial speedups in digital twin applications [18], these approaches introduce challenges in training, validation, and physical consistency [19,20]. Although ML models can reduce the inference times from hours to milliseconds with moderate error margins [9,21], they act as statistical approximations that may violate physical consistency [19], whereas SVS-LDB interpolates directly between valid physics-based solutions. SVS-ML is therefore acknowledged but not benchmarked in order to isolate the trade-off between online computation in LFE and offline storage in LDB.

2.2. SVS Formalism and Notation

An SVS estimates otherwise unmeasurable quantities by combining sparse measurements $x(t)$, known or estimated inputs $u(t)$, an assimilation operator E , and a physics-based model or a database representation derived from that model. We write the mapping as

$$\hat{y}(t) = F\left(M^*, x(t), u(t); E\right) \quad (1)$$

where M^* denotes either a physics-based numerical model M (typically an FE model or a certified reduction) or a database of precomputed responses, $DB(M)$.

The real-time applicability of an SVS is constrained not only by the estimation accuracy but also by the achievable end-to-end latency of the sensing and estimation pipeline. In particular, the total SVS latency

$$\Delta t_{\text{SVS}} = \Delta t_{\text{IO}} + \Delta t_{\text{compute}} \quad (2)$$

must remain below an application-dependent latency budget Δt_{budget} , which is dictated by the control-loop frequency and the data-acquisition cadence. Here, Δt_{IO} comprises sensor readout, data transfer, and optional preprocessing of the measurement vector $x(t)$ and inputs $u(t)$, while $\Delta t_{\text{compute}}$ accounts for the execution of the mapping F , including data assimilation E and either the numerical model evaluation or database querying and interpolation.

Typical latency budgets range from tens of milliseconds in closed-loop control applications to several seconds in monitoring or supervisory contexts. Performance is therefore judged not only by the accuracy against trusted references but also by the latency and computational resource usage [6–8,10].

The total estimation error can be decomposed into contributions from discretization, interpolation, numerical approximations, modeling assumptions, and measurement noise as

$$\epsilon_{\text{total}} = \epsilon_{\text{FEM}}(h) + \epsilon_{\text{interp}}(N, m) + \epsilon_{\text{numerical}} + \epsilon_{\text{modeling}} + \epsilon_{\text{meas}} \quad (3)$$

where $\epsilon_{\text{FEM}}(h)$ is the discretization error of the FE model, which typically decreases as the characteristic element size h is refined, and $\epsilon_{\text{interp}}(N, m)$ is the interpolation error in database-based sensors, which depends on the number of precomputed samples N and the input dimensionality m . The latter term captures the curse of dimensionality. The remaining terms account for solver tolerances or convergence limits $\epsilon_{\text{numerical}}$, simplifications in material laws, geometry, or boundary conditions $\epsilon_{\text{modeling}}$, and measurement noise entering the assimilation step ϵ_{meas} .

In the case studies, the abstract quantities are instantiated as follows. The mesh resolution h governs the degrees of freedom processed by the FE solver, N denotes the number of precomputed database samples, and m denotes the dimensionality of the input parameter space. The controlled experimental setting in this study allows the latency, error, and resource utilization to be quantified consistently for SVS-LFE and SVS-LDB. Performance depends on model fidelity, input-domain parameterization, and the specific estimation choices.

2.3. Performance Metrics and Real-Time Bands

This paper assesses performance along two axes, accuracy and latency. For deployment, the sensor must meet an application-specific latency budget while achieving a predefined accuracy target for the quantities of interest (QoI). We report the accuracy relative to trusted FE references, characterize online latencies under representative workloads, and summarize the runtime variability across repeated queries.

The accuracy is evaluated at two levels. For full-field responses, we report a global mean absolute error computed over the spatial field with respect to a high-fidelity FE reference. For decision-relevant scalars, such as peak stress or maximum die temperature, we report the mean absolute error (MAE) on the corresponding QoI. These metrics capture complementary perspectives, since scalar QoIs can remain within tolerance even when local field deviations are present.

Latency is reported as the compute-time component $\Delta t_{\text{compute}}$. We report both the median and the 95th percentile under a workload that mirrors the intended deployment. Variability is visualized using percentile-based error bars, typically the 5th and 95th percentiles, rather than formal hypothesis tests. Unless stated otherwise, a method is considered deployable if at least 95% of queries satisfy the specified latency budget and the accuracy target for the QoI.

To contextualize the performance metrics, we classify the real-time requirements into three bands based on typical industrial update rates found in the literature. Hard real time (<10 ms) is mandated for the closed-loop control of high-rate dynamic systems, where missing a deadline can result in failure. For instance, structural model updating for impact mitigation requires response times of 1 ms or less [22]. Medium real time (10–100 ms) typically serves human-in-the-loop visualization and supervisory monitoring. Recent digital twins for infrastructure monitoring, such as bridge deformation tracking, operate at update rates of 50 Hz (20 ms) to sufficiently capture structural dynamics [23]. Soft real time (>1 s) is sufficient for processes with high physical inertia, such as thermal management. For example, predictive maintenance twins for electric motors often employ update intervals of several seconds, which is adequate to track temperature evolution [24].

2.4. Cost and Scaling Models

Our focus is the trade-off between two direct physics-based implementations that underpin many SVS architectures, SVS-LFE and SVS-LDB. SVS-LFE minimizes offline investment but typically incurs higher online latency. SVS-LDB reduces online latency but requires offline precomputation. In this work, cost refers to the computational runtime. We distinguish the end-to-end latency Δt_{SVS} , defined in Equation (2) and including both I/O and compute time, from the implementation-specific compute-time terms t_{LFE} and t_{LDB} that enter the scaling laws below. For lookup database sensors, we additionally account for the one-time offline investment T_{pre} required to populate the database.

Figure 1 summarizes the SVS trilemma between online speed, offline creation effort, and predictive accuracy.

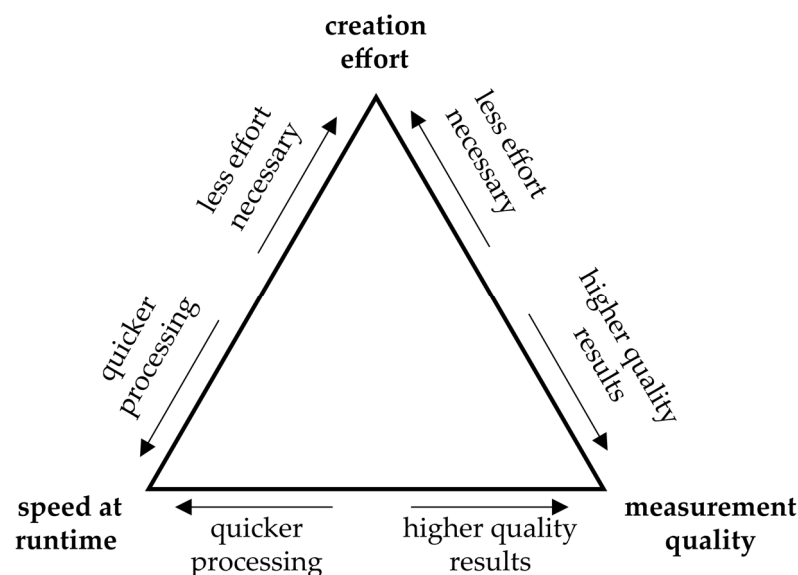


Figure 1. Conceptual trilemma for simulation-based virtual sensors (SVS): online speed (runtime latency), offline creation effort (precomputation and model development), and predictive accuracy (measurement and estimation quality).

For the live FEM approach, the key design variable is the number of degrees of freedom n processed at runtime. This number reflects the mesh resolution of the underlying FE model or a certified reduced representation and directly affects the computational cost and discretization error. The runtime per query is modeled by the empirical scaling law

$$t_{\text{LFE}}(n) = c_1 n^\alpha, \quad \alpha > 1 \text{ typically}, \quad (4)$$

where α captures the effective scaling observed in the measured range. In many settings, the asymptotic trend is faster than linear due to assembly, solver iterations, and memory traffic, while for smaller models, constant overheads can dominate and partially mask the underlying behavior. The parameters c_1 and α are obtained by linear regression in log space from controlled degree-of-freedom sweeps, and the fitted values are reported in Section 3.3.

For lookup database sensors, two variables are central: the number of precomputed samples N and the input dimensionality m . The total offline cost of generating the database is expressed as

$$T_{\text{pre}}(m, N) = c_3 N^\gamma m^\delta, \quad (5)$$

where $\gamma \geq 1$ reflects throughput and batching effects in the FE runs, and $\delta \geq 0$ accounts for overheads from experiment design, validation, indexing, and storage. The coefficient c_3 and the effective exponents are estimated from batch runs across different N and, where available, varying m . The resulting offline times are then used to determine the break-even query count Q_{be} required to amortize the offline investment in Section 3.3.

The online query time depends on the data structure used for interpolation. With spatial indexing such as k-d trees, the expected scaling is logarithmic,

$$t_{\text{LDB}}(N) = c_4 \log N, \quad (6)$$

where c_4 captures implementation and hardware constants. Without indexing, a naive neighbor search grows linearly with N , and empirical slopes are reported instead.

These cost models are linked to the accuracy requirements. For SVS-LFE, an accuracy target ε^* can be translated into a required model resolution and thus into a required $n(\varepsilon^*)$, which determines the predicted runtime via Equation (4). For SVS-LDB, achieving the same target implies a minimum database size that depends on N and m , which then determines both the offline cost via Equation (5) and the online latency via Equation (6). In practice, the sample requirements tend to increase rapidly with m , and the accuracy can deteriorate near parameter-space boundaries. These effects are examined in Sections 2.5 and 2.6.

Together, Equations (4)–(6) provide a compact quantitative basis for the empirical break-even analysis in Section 3.3 and for the design rules discussed in Section 4.

2.5. Design of Experiments for Lookup Databases

For the lookup database workflow, the choice of design of experiments (DoE) directly affects the accuracy and robustness [25]. We consider three widely used classes of space-filling designs. First, Orthogonal Arrays (OA) are included as a baseline typical for physical experiments. However, their projection onto discrete levels limits their ability to capture nonlinearities in deterministic computer models [26]. Second, Latin Hypercube Sampling (LHS) and, third, Sobol sequences are employed as space-filling designs. Recent benchmarks indicate that Sobol sequences often outperform LHS in convergence speed for sensitivity analysis, making them particularly suitable for high-dimensional digital twins [27]. Sobol and Latin hypercube designs were generated using Python (v3.11) and the SciPy package (v1.16.3), while the OA designs follow an orthogonal-array-inspired stratification scheme, and all designs were created with fixed random seeds for reproducibility.

In the one-dimensional tension-bar case (Case Study A), the input domain is discretized by equally spaced points; so, the design choice is trivial. For the CPU case (Case Study B), we construct DoE designs in four, six, and nine dimensions, corresponding to different parameterizations of the power-map generator described in Section 2.8. For each combination of dimensionality and design class, we generate several database sizes N within the practical range for offline simulation.

All designs are generated using standard numerical routines and are checked for obvious defects such as clustering or alignments with coordinate axes. Their impact on accuracy is assessed in the Case Study B using ground-truth errors against an independent reference set and 10-fold cross-validation.

2.6. Interpolation and Online Query

Online queries are answered by interpolating from the precomputed database using k -nearest-neighbors regression with inverse-distance weighting. The number of neighbors is fixed to three. Preliminary screening indicated that, for the dense databases required for engineering accuracy, the sensitivity of the error to small variations in k is negligible compared to the influence of the database size N , while the query latency is primarily governed by the tree depth ($\mathcal{O}(\log N)$) rather than the neighbor count. If multiple design points coincide in the input space, the prediction reduces to the average of their associated responses. Neighbor selection is performed in the bounded input parameter space defined for the design, and we do not consider extrapolation beyond the sampled operating envelope.

The interpolator is implemented in Python using the scikit-learn k -nearest-neighbors regressor, with NumPy and pandas for data management. Latency is measured after a short warm-up period by issuing repeated single-sample queries with fixed hyperparameters, and the latency statistics defined in Section 2.3 are recorded under steady-state conditions.

For Case Study A (1D Tension Bar), the accuracy is evaluated against the high-fidelity experimental reference curve (described in Section 2.7) to quantify the deviation from physical reality.

For the multivariate analysis (Case Study B), the accuracy is evaluated using an independent random hold-out reference set ($N_{\text{val}} = 50$) that is disjoint from the design points described in Section 2.5. To avoid methodological pitfalls in this high-dimensional setting, we report the primary error metrics on an independent hold-out set and provide 10-fold cross-validation results as a diagnostic. Although k -fold cross-validation is widely used, it can exhibit optimistic bias (data leakage), particularly when samples possess structural correlations or when the sample size is limited [28]. By using a never-seen holdout set, we ensure that the reported errors represent the true generalization performance rather than fitting artifacts. We report the MAE for the QoIs together with the corresponding latency statistics under a workload that mirrors the intended deployment.

2.7. Case Study A: Tension Bar

This case study examines a single-input tensile problem with elastic–plastic material behavior to quantify runtime scaling and assess LDB performance in a low-dimensional setting. It is important to distinguish between the spatial dimensionality of the model and the dimensionality of the input space. Although the finite element simulation employs a full 3D hexahedral mesh to capture realistic solver overheads, the virtual sensor input space is one-dimensional ($m = 1$), parameterized solely by the prescribed axial displacement boundary condition. The FEM model serves here as a high-fidelity reference generator to evaluate the surrogate’s approximation error and runtime scaling. A dog-bone specimen was modeled using linear hexahedral elements. The experimental reference data were adopted from the SH4 Inconel 625 dataset [29], which documents tensile tests performed

according to DIN EN ISO 6892-1 [30]. An elastic modulus of 194.75 GPa was adopted. To ensure numerical robustness, the raw stress–strain data were processed using outlier removal and Savitzky–Golay smoothing before being reduced to 100 support points. The physical problem is governed by the quasi-static equilibrium of the Cauchy stress tensor σ ($\nabla \cdot \sigma = 0$), neglecting body forces. The material response is modeled using isotropic elastoplasticity based on the von Mises yield criterion [31].

Three mesh resolutions were analyzed to determine the runtime scaling: a coarse mesh ($h = 3.0$ mm), an intermediate mesh ($h = 1.0$ mm), and a fine reference mesh ($h = 0.3$ mm), spanning 1.6×10^3 to 6.0×10^5 degrees of freedom (Table 1). The finite element discretization is visualized in Figure 2.

Table 1. Mesh resolutions used in the tension-bar case study, including element counts and degrees of freedom for the three grids.

Mesh Resolution	Mesh Edge Length (mm)	Number of Elements	Degrees of Freedom
Coarse	3.0	222	1596
Fine	1.0	4872	21,060
Ultra fine	0.3	176,180	595,584

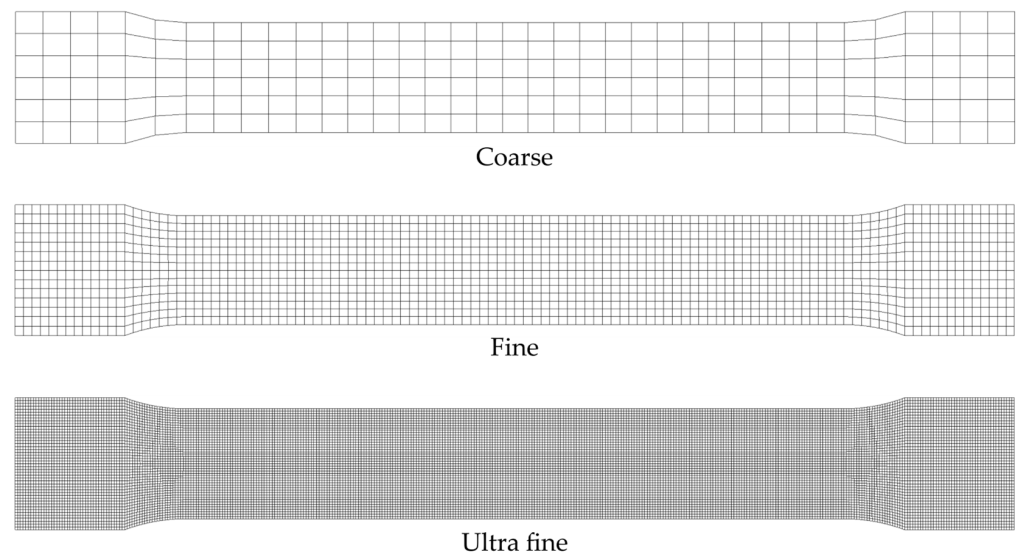


Figure 2. Detailed view of the finite element mesh for the three refinement levels.

Simulations were performed using the Z88Aurora® V6 (Engineering Design and CAD, University of Bayreuth, Germany) nonlinear solver under displacement control up to 30% engineering strain. The workflow was automated using a custom Python orchestration script that programmatically modified the input parameter within the project files for each design point. For each run, a unique directory was generated, the modified files were deployed, and the solver was invoked via the command-line interface. Post-processing scripts subsequently extracted the reaction forces and stress measures from the binary output files. Reaction forces were accumulated over the grip faces, and engineering strain was computed via a virtual extensometer $L_0 = 50$ mm defined within the gauge section, as shown in Figure 3.

For the database workflow, the input domain was discretized into 159 equidistant displacement levels corresponding to engineering strains up to 30%. The database stores the grip force, gauge strain, and stress measures. The accuracy is evaluated relative to the ultra-fine-mesh reference and experimental data, while the query latency is measured for the lookup interpolation step defined in Section 2.6.

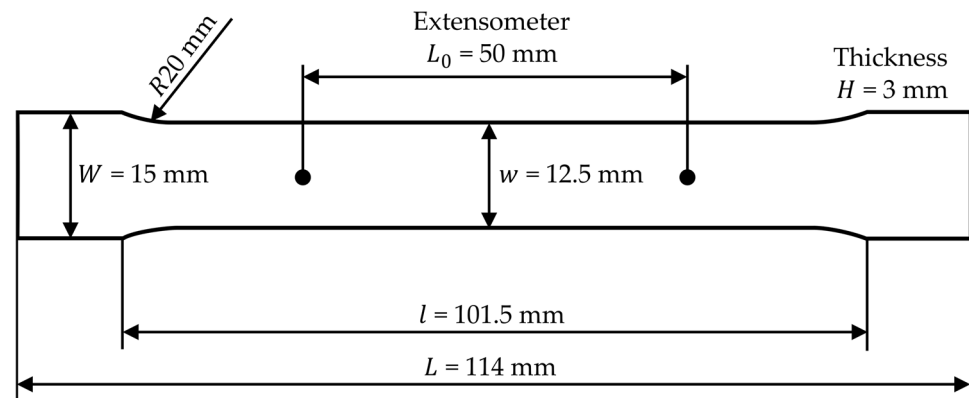


Figure 3. Tensile specimen geometry and boundary conditions for Case Study A. A prescribed axial displacement u is applied at one grip face while the opposite grip is constrained. Engineering strain is evaluated using a virtual extensometer defined by two marker sets in the gauge section with a gauge length of $L_0 = 50$ mm.

2.8. Case Study B: Thermal Analysis of a CPU Die

The second case study uses a simplified steady-state thermal model of a quad-core CPU. Compared to Case Study A, this benchmark introduces a higher-dimensional design space and a spatially varying response field while enabling controlled comparisons of sampling strategies and database sizes. This is a purely numerical benchmark designed to evaluate sampling efficiency and grid convergence without experimental uncertainties. The CPU model, based on Intel Core i7-8650U specifications [32], comprises an FR-4 base, a silicon die ($14 \text{ mm} \times 9 \text{ mm} \times 0.5 \text{ mm}$), and a copper heat sink (Table 2). The thermal behavior is governed by the steady-state heat conduction equation $-\nabla \cdot (k \nabla T) = q_v$ for the temperature field T , where k denotes the thermal conductivity, and q_v represents the volumetric heat generation in active cores ($q_v = 0$ elsewhere). Heat dissipation at the sink interface is modeled via a convective boundary condition $-k \nabla T \cdot n = h(T - T_\infty)$ [33].

Table 2. Thermal material properties assigned to the components of the CPU package assembly.

Component	Material	k ($\text{W m}^{-1} \text{K}^{-1}$)	c_p ($\text{J kg}^{-1} \text{K}^{-1}$)	ρ (kg m^{-3})
FR-4 base board	FR-4 epoxy	6	1150	1900
CPU die	Silicon	174	712	2330
Copper heat sink	Copper	400	385	8933

Thermal loads are applied to the die as shown in Figure 4, with the heat dissipation modeled via convection ($h_{\text{conv}} = 29.5 \text{ W m}^{-2} \text{K}^{-1}$, $T_\infty = 33^\circ\text{C}$).

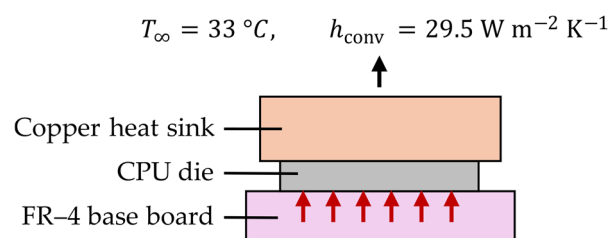
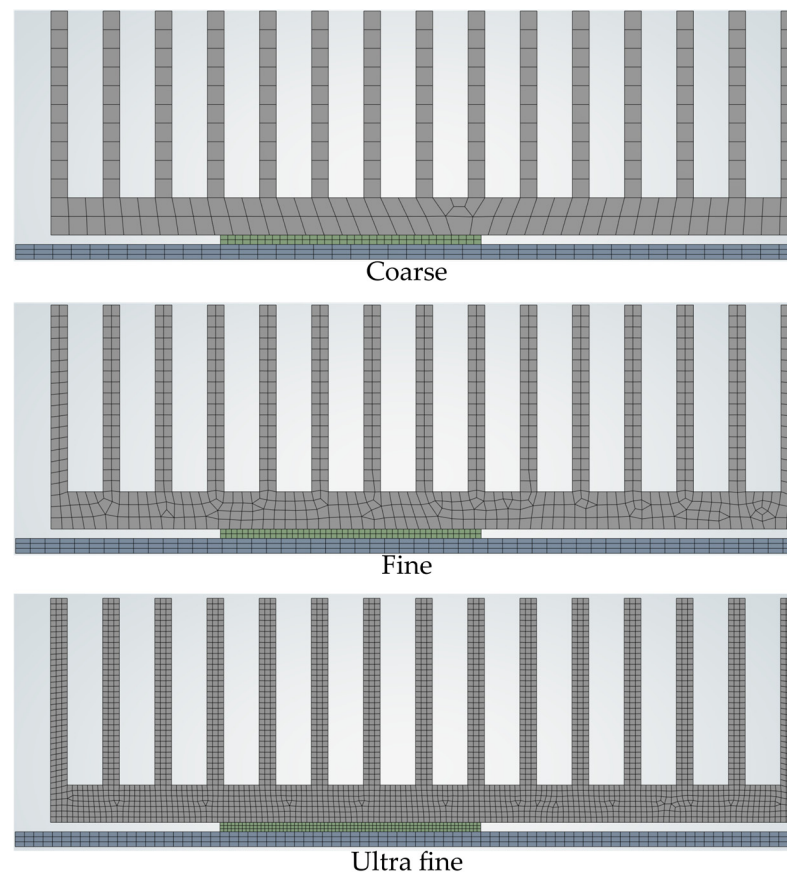


Figure 4. Layered CPU package representation and boundary conditions used in the steady-state thermal model. Heat flux is applied at the die underside, and heat dissipation is modeled via convection at the heat sink interface.

Grid convergence was assessed using three hex-dominant meshes ranging from approx. 5.8×10^4 to 7.8×10^5 degrees of freedom (Table 3). The finite element discretization is visualized in Figure 5.

Table 3. Grid convergence study for the CPU thermal model. Element counts and degrees of freedom (DoF) for the three mesh refinement levels.

Mesh Resolution	Number of Elements	Degrees of Freedom
Coarse	10,010	57,803
Fine	34,482	175,062
Ultra fine	162,864	783,005

**Figure 5.** Detailed view of the finite element mesh for the three refinement levels.

Spatially varying power distributions were generated by discretizing the die into functional blocks (Cores 0–3, Ring/Interconnect, L3 Cache, System Agent, GPU, and DDR) and varying the total power P_{total} from 0.6 W to 10.5 W. Power was mapped to density fields using either uniform distributions or Gaussian profiles to ensure diverse load cases.

The simulation pipeline was automated using a Python-based runner. For each simulation task, the script created a dedicated working directory and generated an ANSYS Workbench journal file, which was executed within ANSYS Mechanical 2024 (Ansys, Inc., Canonsburg, PA, USA) to programmatically update the spatially resolved heat-flux maps, run the steady-state thermal solve, and export the resulting temperature fields. This automated orchestration ensured consistent execution and data integrity across the multi-dimensional parameter space.

To study the curse of dimensionality, LDBs were constructed for input spaces with $m \in \{4, 6, 9\}$ variable parameters (Table 4). Here, m denotes the dimension of the parameter space (number of independently varied thermal loads), while the underlying multiphysics model remains a spatially three-dimensional continuum. From the perspective of Digital Twin design, this variation simulates different levels of sensor coverage and model fidelity. The variable parameters correspond to the active state space monitored by

real-time readings (e.g., core loads), while fixed parameters represent unmonitored subsystems assumed to operate at nominal boundary conditions. By systematically expanding the variable space from 4 to 9 dimensions, we do not aim to maximize physical realism for a specific load case but to benchmark how the required database size and sampling strategy scale with the complexity of the monitored system.

Table 4. Definition of variable input parameters for the 4D, 6D, and 9D CPU lookup database configurations.

Number of Variable Parameters	Variable Parameters	Fixed Parameters
4	cores 0–3	ring/interconnect, L3 cache, system agent, GPU, and DDR
6	cores 0–3, ring/interconnect, system agent	L3 cache, GPU, and DDR
9	cores 0–3, ring/interconnect, L3 cache, system agent, GPU, and DDR	-

For each dimensionality, multiple design of experiments strategies were considered, including Sobol sequences, maximin Latin hypercube sampling, and orthogonal arrays. The evaluated combinations of design class, database size N , and dimensionality m are summarized in Table 5. The sample sizes were adapted to dimensionality: for $m = 4$, sizes ranged from $N = 32$ to 128, whereas for $m \in \{6, 9\}$, the range was shifted to $N = 64$ to 256. For Sobol sampling in 6D, the sweep was extended to $N = 1024$ to analyze asymptotic convergence behavior.

Table 5. Overview of design of experiments strategies and sample sizes N used for lookup database generation in the CPU case study, grouped by input dimensionality m .

Method	Number of Simulations	Number of Variable Parameters
Sobol	32	4
	64	4, 6
	128	4, 6, 9
	256	6, 9
	512	6
	1024	6
LHS-Maximin	32	4
	64	4, 6
	128	4, 6, 9
	256	6, 9
Orthogonal arrays	32	4
	64	4, 6, 9
	128	4, 6, 9
	256	6, 9

Surrogate accuracy is evaluated as the mean absolute error (MAE) of the temperature field relative to an independent validation set ($N = 50$) generated via random sampling, consistent with the metrics in Section 2.3. Query latencies are measured directly from the lookup and interpolation steps.

2.9. Experimental Study Design

To ensure a systematic evaluation of the scaling laws and accuracy trade-offs, the investigation follows a structured study plan. We vary the mesh resolution (n) to assess solver scaling, the input dimensionality (m) to study complexity effects, and the sample size

(N) to determine surrogate performance. Table 6 summarizes the technical configuration and the alignment of the parameters with the results presented in Section 3.

Table 6. Comprehensive experimental study plan and data characterization.

Feature	Case Study A: Tension Bar	Case Study B: CPU Thermal Model
Objective	Numerical vs. Experimental validation	High-dimensional surrogate benchmarking
FE Reference	$h = \{3.0, 1.0, 0.3\}$ mm; DoF up to $6 \cdot 10^5$; Solver: Z88Aurora V6	3 Levels : $5.7 \cdot 10^4$ to $7.8 \cdot 10^5$ DoF; Solver: ANSYS Mechanical
Boundary conditions	Fixed face constraint, Prescribed displacement	Convection (h , T_∞), Applied power mapping
Input Space (m)	$m = 1$ (Prescribed displacement u)	$m \in \{4, 6, 9\}$ (Variable power blocks)
Database Plan (N)	$N = \{19, 39, 79, 159\}$ (Equidistant)	$N \in \{32, 64, 128, 256, 512, 1024\}$ (Sobol, LHS, OA)
Database Structure	Input: u ; Output: Reaction force F , Engineering strain ϵ , Stress σ	Input: Power vector $P \in R^m$; Output: Temperature vector $T \in R^M$
Validation	Against experimental dataset	Against independent random validation set ($N = 50$)

3. Results

3.1. Case Study A: Tension Bar

This case study considers a one-dimensional tensile problem with elastic–plastic material behavior. The goal is twofold: first, to validate the live FEM (LFE) simulation against experimental data; second, to quantify the speed–accuracy trade-off between live FEM and a lookup database (LDB) sensor in a low-dimensional setting.

3.1.1. Live FEM Accuracy and Runtime Scaling

Figure 6 compares the experimental engineering stress–strain curve with the live FEM simulation on the ultra-fine mesh ($h = 0.3$ mm). Over the full range up to 30% engineering strain, the simulated response matches the measurement closely.

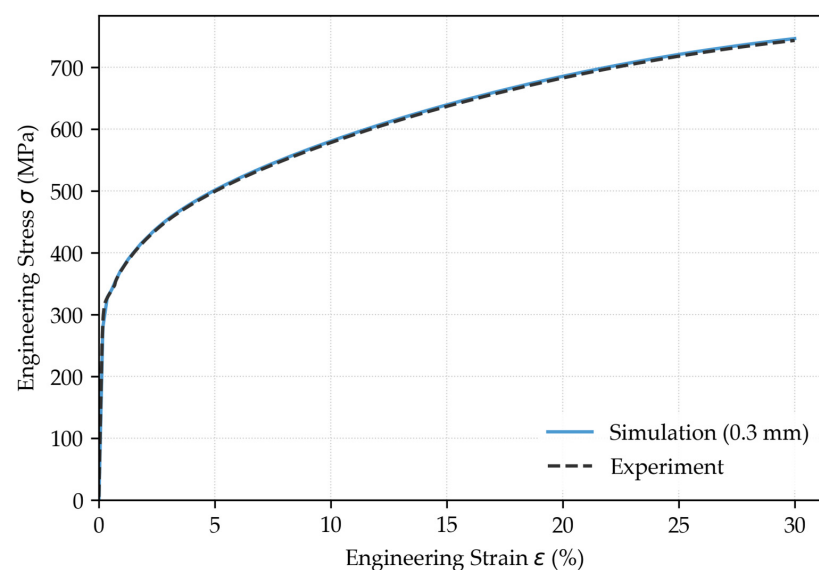


Figure 6. Comparison of engineering stress–strain curves.

The mean absolute error (MAE) between the simulation and experiment is $MAE_{\sigma} \approx 2.54$ MPa, which corresponds to about 0.34% of the maximum experimental stress (743 MPa).

To quantify the discretization effects, three uniform meshes with element sizes $h = 3.0$ mm, 1.0 mm, and 0.3 mm were evaluated. Figure 7a reports the stress MAE relative to the experimental curve, and Figure 7b reports the corresponding wall-clock runtime.

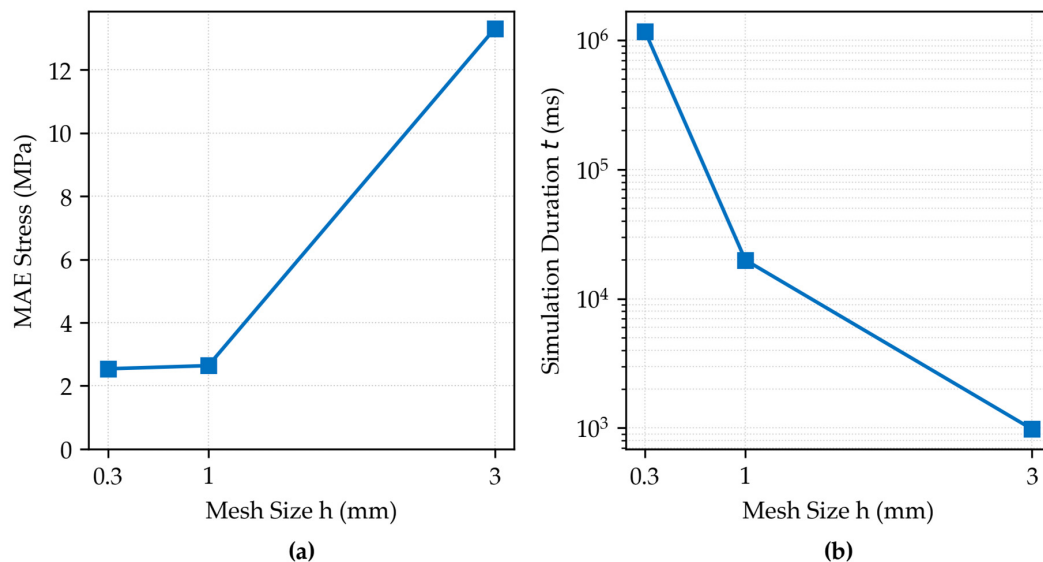


Figure 7. Mesh convergence and computational cost. (a) Mean absolute error (MAE) of engineering stress relative to the experimental curve as a function of mesh element size h . (b) Simulation runtime versus element size h on a logarithmic scale.

The coarsest mesh ($h = 3.0$ mm) shows a noticeable deviation of 13.3 MPa ($\approx 1.8\%$). Refinement to $h = 1.0$ mm reduces the error to 2.64 MPa ($\approx 0.35\%$), which is close to the ultra-fine-mesh result of 2.54 MPa. The 95th-percentile relative errors follow the same trend, decreasing from 3.17% at $h = 3.0$ mm to about 0.44% and 0.39% at $h = 1.0$ mm and 0.3 mm, respectively.

The runtime increases sharply with the refinement. The average wall-clock times are approximately 1.18 s for $h = 3.0$ mm, 21 s for $h = 1.0$ mm, and on the order of 10^3 s for $h = 0.3$ mm (about 20 min). Expressed as a function of degrees of freedom n , these measurements are well described by the power-law model introduced in Equation (4), with an effective exponent of $\alpha \approx 1.2$ over the tested range. Given the span of runtimes over three orders of magnitude, fitting to the 5th and 95th percentiles yields negligible deviation ($< 1\%$), confirming a highly robust scaling law.

3.1.2. Lookup Database Accuracy and Query Latency Versus Database Size

The LDB sensor was constructed from FEM simulations computed on the $h = 1.0$ mm mesh and evaluated against the experimental reference. Starting from the full set of 159 displacement samples, reduced databases were created by subsampling approximately 12%, 25%, and 50% of the points, in addition to the full design ($N = 19, 39, 79, 159$). For each database size, a distance-weighted k-nearest neighbors interpolator was instantiated and evaluated on a dense displacement grid.

Figure 8 shows that the stress MAE decreases from 9.60 MPa at $N = 19$ to 2.75 MPa at $N = 159$. Using 50% of the samples ($N = 79$) yields an MAE of 3.52 MPa and a 95th percentile relative error below 0.5%.

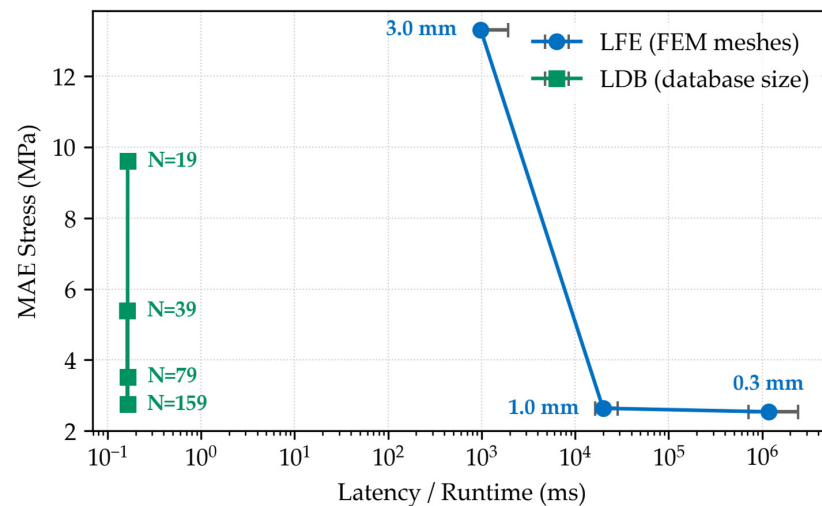


Figure 8. Speed–accuracy trade-off for the tension-bar case study. Live FEM results (circles) are shown for different mesh resolutions using simulation runtime as the horizontal axis. LDB results (squares) are shown for different database sizes N using per-query latency as the horizontal axis. The vertical axis shows stress MAE, and the horizontal axis is logarithmic.

The query latency remains nearly constant across all tested database sizes. Median values lie in the narrow range of 0.163–0.164 ms, with 5–95th percentile intervals of approximately 0.159–0.181 ms.

3.2. Case Study B: Thermal Analysis of a CPU Die

The second case study examines a multi-parameter scenario using a simplified CPU die model. The objective is to evaluate how the live FEM workflow and an LDB sensor scale with respect to the number of inputs and design points when predicting steady-state temperature fields.

3.2.1. Live FEM Grid Convergence and Runtime Scaling

Three hex-dominant meshes with 57,803, 175,062, and 783,005 unconstrained degrees of freedom (DoF) were evaluated for the live FEM workflow. Figure 9a reports the mean absolute error (MAE) of the temperature field relative to the ultra-fine mesh. The coarse mesh exhibits an MAE of 1.98 K, which is reduced to 1.17 K on the fine mesh. By definition, the reference mesh has zero error in this comparison.

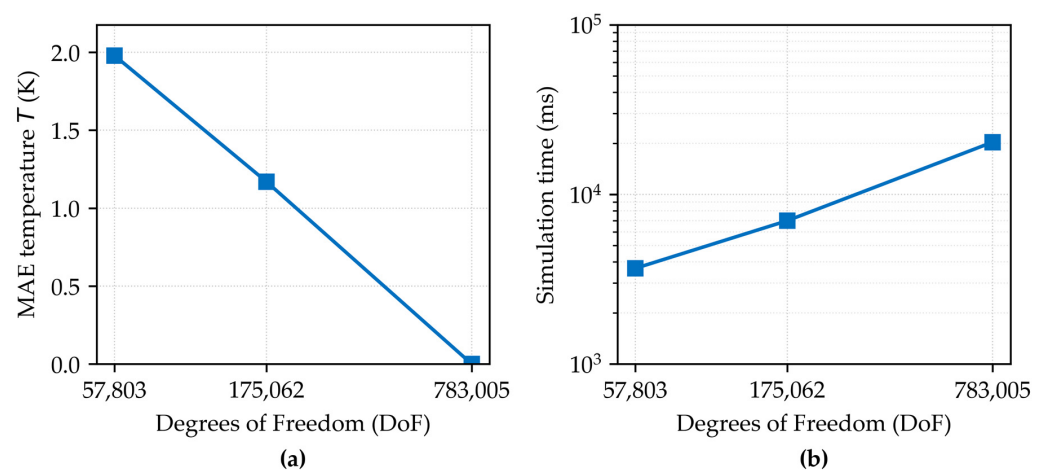


Figure 9. Live FEM grid convergence and runtime. (a) Temperature MAE versus DoF relative to the ultra-fine mesh. (b) Solver runtime versus DoF on a logarithmic scale.

The wall-clock runtimes are summarized in Figure 9b. Median runtimes increase from 3.67 s at 57,803 DoF to 7.00 s at 175,062 DoF and to 20.33 s at 783,005 DoF. The corresponding minimum–maximum ranges are 3–4 s, 6–8 s, and 20–21 s, respectively. Fitting the power-law model from Equation (4) over this limited DoF range yields an effective exponent of $\alpha \approx 0.66$. Sensitivity analysis using the minimum and maximum recorded runtimes yields exponents in the range of 0.63 to 0.73. This confirms that despite the timing variability, the effective scaling remains consistently sublinear ($\alpha < 1$) due to the dominant overheads.

3.2.2. Sampling Strategies in Higher Dimensions

To study the influence of sampling design and input dimensionality on surrogate quality, LDBs were constructed for parameter spaces with $m = 4, 6$, and 9 . For each dimensionality, three sampling strategies were compared at matched database sizes N , namely maximin Latin hypercube sampling (LHS), scrambled Sobol sequences, and orthogonal arrays (OA). Two complementary error measures are reported. Figure 10 shows MAE against an independent ground-truth reference set, while Figure 11 shows the 10-fold cross-validation MAE computed on the available samples.

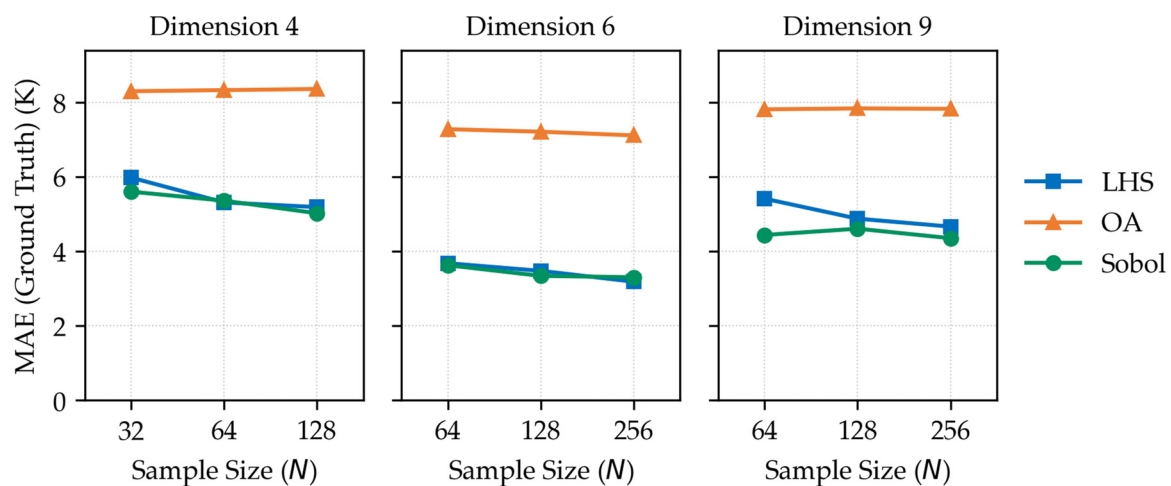


Figure 10. LDB accuracy versus database size. Temperature MAE against an independent ground-truth reference set for LHS, Sobol, and OA in 4D, 6D, and 9D.

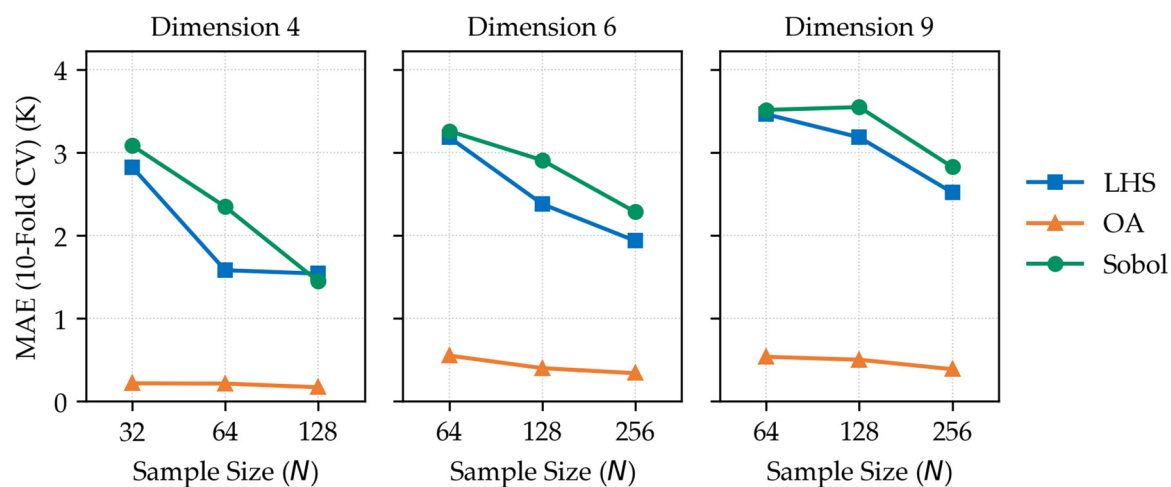


Figure 11. Cross-validation accuracy versus database size. Ten-fold cross-validation MAE for LHS, Sobol, and OA in 4D, 6D, and 9D.

For LHS and Sobol, the ground-truth MAE decreases moderately with increasing N across all tested dimensions. In 4D, LHS improves from 5.98 K at $N = 32$ to 5.18 K at $N = 128$, while Sobol improves from 5.60 K to 5.02 K. In 6D, LHS decreases from 3.68 K at $N = 64$ to 3.18 K at $N = 256$, while Sobol decreases from 3.62 K at $N = 64$ to 3.30 K at $N = 256$ and further to 2.79 K at $N = 1024$. In 9D, LHS improves from 5.41 K at $N = 64$ to 4.66 K at $N = 256$, while Sobol remains comparable in magnitude and reaches 4.35 K at $N = 256$.

The OA designs exhibit a markedly different pattern. Their cross-validation errors are very low, for example 0.17–0.22 K in 4D, 0.34–0.55 K in 6D, and 0.39–0.54 K in 9D. At the same time, the ground-truth MAE is substantially higher and nearly constant with N , approximately 8.29–8.35 K in 4D, 7.11–7.27 K in 6D, and 7.81–7.83 K in 9D.

3.2.3. Speed–Accuracy Comparison of Live FEM and LDB

We next compare the live FEM runtime and accuracy against LDB query latency and accuracy using a representative 6D Sobol configuration. Figure 12 summarizes the results in a combined speed–accuracy plot. The Live FEM achieves temperature-field MAE values of 1.98 K on the coarse mesh and 1.17 K on the fine mesh relative to the ultra-fine reference, with median runtimes of 3.67 s and 7.00 s per query. The finest mesh serves as the reference and has zero error by construction but requires 20.33 s median runtime.

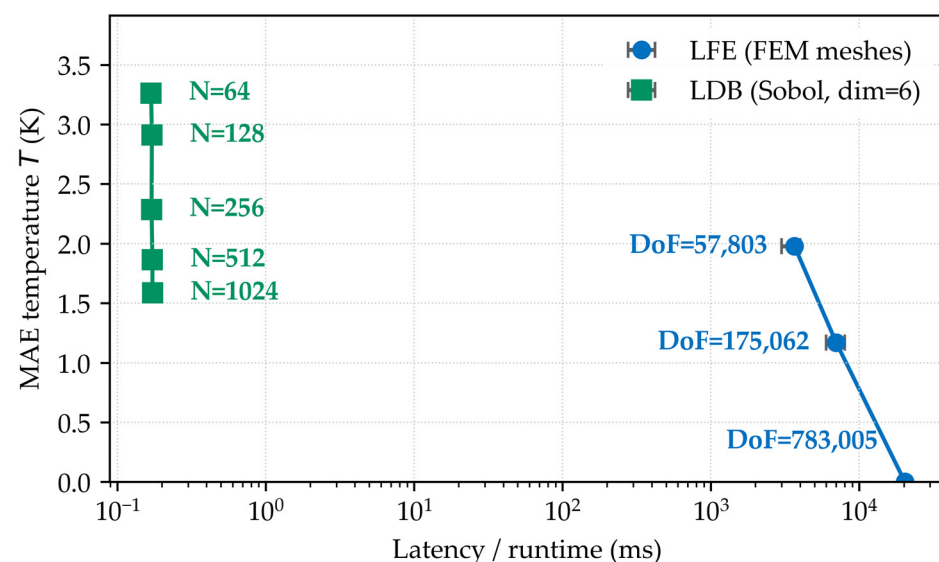


Figure 12. Speed–accuracy trade-off for the CPU case study. Live FEM results are plotted using runtime per query, and LDB results are plotted using per-query latency. The horizontal axis is logarithmic, and timing variability is indicated by percentile-based bars. Bars for LDB are not visible due to low variability.

The corresponding 6D Sobol LDB sensors achieve MAE values ranging from 3.62 K at $N = 64$ to 2.79 K at $N = 1024$. The latency is measured as the time for a single online query, meaning one input vector mapped to one prediction. Under this definition, the LDB query latency is nearly constant across the tested database sizes. Median values lie between 0.169 ms and 0.173 ms, with 5th–95th percentile bounds of approximately 0.160–0.188 ms.

3.3. Scaling Laws and Break-Even Summary

3.3.1. Fitted Scaling Parameters

To summarize the runtime behavior of the two workflows, we fitted the generic models introduced in Section 2.4 to measurements from the case studies. For the live FEM workflow, we used the power-law model in Equation (4) and report the fitted scaling

exponent α . For the LDB workflow, we report measured per-query latency and treat it as approximately constant over the investigated database sizes.

- Case Study A. Using three meshes with 1.6×10^3 , 2.1×10^4 , and 5.9×10^5 DoF and their median wall times of 0.98 s, 19.92 s, and 1172 s, the fitted exponent is $\alpha \approx 1.20$. In contrast, the LDB sensor exhibits an essentially constant per-query latency of $t_{\text{LDB}} \approx 1.64 \times 10^{-4}$ s (about 0.164 ms) across all database sizes considered.
- Case Study B. The three thermal meshes with 57,803, 175,062, and 783,005 DoF require median solver times of 3.67 s, 7.00 s, and 20.33 s, respectively, resulting in $\alpha \approx 0.66$ for this configuration over the tested range. Across the investigated LDB designs and dimensionalities, the measured median per-query latency lies within a narrow range of approximately 0.165–0.173 ms. The 5th–95th percentile interval is approximately 0.160–0.193 ms, and no systematic dependence on N is observed within the studied range.

3.3.2. Empirical Break-Even Points

Given these scaling laws, the break-even point between SVS-LFE and SVS-LDB is defined by the query count Q_{be} , at which the cumulative runtime is equal. In Case Study A, constructing the full database with $N = 159$ design points on the fine mesh requires an offline time of approximately $T_{\text{pre}} = 159 \times 19.92 \text{ s} = 3.17 \times 10^3 \text{ s}$. With an online live FEM runtime of 19.92 s per query and an LDB latency of 1.64×10^{-4} s per query, the break-even count is essentially equal to the database size, $Q_{\text{be}} \approx N$.

For Case Study B, the same calculation yields break-even counts several times larger than the design size when using the fine-mesh live FEM runtime of $T_{\text{LFE}} \approx 7.00 \text{ s}$ as the online baseline. Across the investigated configurations, LHS and OA designs typically yield $Q_{\text{be}} \approx 9 N$ to $14 N$. Sobol designs exhibit higher and more variable amortization factors. For example, the 6D Sobol design with $N = 1024$ reaches break-even after $Q_{\text{be}} \approx 2.0 \times 10^4$ queries (approximately $19 N$). Assuming continuous operation with one query evaluated every t_{LFE} seconds, this corresponds to approximately 4.6 days for $t_{\text{LFE}} = 19.92 \text{ s}$ (or 1.6 days for $t_{\text{LFE}} = 7 \text{ s}$). The largest amortization factor observed in the present set occurs for the 9D Sobol design with $N = 128$, where Q_{be} approaches $43 N$.

4. Discussion

4.1. Synthesis of Results

The two case studies illuminate complementary aspects of the trade-off between live FEM and lookup database workflows and clarify which cost and accuracy terms dominate in practice.

Case Study A provides a controlled low-dimensional setting in which the accuracy and runtime can be related directly to the mesh refinement and database density. The live FEM results show that the stress errors decrease with refinement, while the runtimes increase by orders of magnitude across the tested meshes. The lookup database results show that accuracy improves with increasing database size, and that per-query latency remains effectively constant over the investigated range.

Case Study B extends the comparison to a higher-dimensional thermal problem in which input dimensionality and sampling design become central. Live FEM runtimes remain in the multi-second regime across the tested meshes, and the fitted exponent $\alpha \approx 0.66$ should be understood as an effective scaling over the measured range rather than as an asymptotic solver behavior. For these problem sizes, constant overheads in the scripted workflow and solver setup contribute substantially to the wall-clock time. These overheads can mask the underlying solver scaling. A comparison between the two case studies reveals that the orchestration strategy is a decisive factor for the total

runtime. Case Study A utilizes a lightweight approach. Here, input files are edited directly via Python before a command-line solver call. In contrast, Case Study B relies on ANSYS Workbench journal files. This requires the application to load internal thermal load-mapping routines for each design point. This heavyweight I/O and application overhead results in significantly higher constant costs (c_1) compared to the direct-file-access method. This observation aligns with the established performance characteristics in many-task computing. The literature confirms that middleware initialization and file I/O often dominate the total runtime when individual task durations are short [34]. In contrast, lookup database query latency remains near a case-specific constant across the investigated database sizes and dimensions.

The sampling study demonstrates that surrogate quality depends not only on database size but also on design structure and on how error is estimated. In particular, orthogonal array designs yield very low cross-validation errors while maintaining high ground-truth errors that improve little with additional samples. This behavior indicates that cross-validation can be overly optimistic for highly structured designs and that independent reference validation is necessary when global accuracy is required.

Across both case studies, the fitted timing parameters identify the dominant cost drivers of each workflow. The live FEM cost is governed by per-query runtime, whereas the lookup database online cost is dominated by a small constant overhead and is negligible compared with multi-second simulation runtimes. As a result, the dominant cost component of the lookup database workflow is the offline precomputation time T_{pre} . The break-even analysis confirms that the query count Q_{be} is primarily governed by the offline construction cost relative to the chosen live FEM baseline, with amortization factors ranging from order N in one dimension to several multiples of N in the higher-dimensional CPU study.

To place the performance of the proposed SVS-LDB workflow in the context of the state of the art, we compared the measured latencies with the results reported for alternative surrogate strategies. Deep Learning emulators for physical fields typically exhibit inference times in the range of 5 to 50 ms [35,36], often dominated by framework and data-transfer overheads rather than model depth. Similarly, projection-based Reduced Order Models often necessitate 10 to 100 ms for the online reconstruction step [37]. In contrast, the k-d tree lookup operates consistently below 1 ms. This performance aligns with algorithmic benchmarks for the nearest-neighbor search [38], which identify tree-based spatial indexing as the most computationally efficient query method for moderate dimensions ($m \lesssim 20$) before the curse of dimensionality necessitates approximate methods. This indicates that for the moderate dimensionality and complexity investigated, the LDB approach offers a distinct advantage by avoiding the architectural overheads of heavier algebraic or neural surrogates, effectively eliminating the computational bottleneck of the virtual sensor in the sub-millisecond regime.

These observations form the basis for the practical design guidelines presented next.

4.2. Design Guidelines for Real-Time Virtual Sensors

The results from Sections 3.1–3.3 suggest four practical guidelines for selecting between live FEM and lookup database workflows when designing real-time virtual sensors.

First, the latency budget sets a hard feasibility constraint for live FEM. For the model sizes examined here, live FEM runtimes remain in the seconds regime once meshes reach typical engineering resolution, even when the discretization error is acceptable. In contrast, lookup database queries operate at sub-millisecond latency in both case studies. Therefore, when the required update rate approaches sub-second cadence and especially when budgets

move toward the millisecond range, a lookup database becomes the default choice unless substantial model reduction or solver acceleration is available.

Second, the expected query horizon determines whether offline precomputation can be justified. The lookup database workflow incurs a one-time cost T_{pre} to populate the database and a small per-query cost t_{LDB} during operation, whereas the live FEM workflow incurs a per-query cost t_{LFE} . A simple break-even criterion follows from equating cumulative runtimes. The database becomes advantageous once the number of online queries exceeds Q_{be} , where Q_{be} is governed primarily by the offline investment T_{pre} relative to the selected live FEM baseline. In practical terms, the lookup database approach is most attractive when the operating period includes repeated queries at a fixed configuration, such as continuous monitoring, parameter sweeps in a control loop, or frequent what-if evaluations.

Third, the input dimensionality affects the offline effort and achievable accuracy more than it affects the online latency in the studied range. The query latency remains nearly constant with the database size and dimension, because the dominant contribution is an implementation-specific constant overhead. The main scaling challenge with increasing dimension is the number of samples required to maintain global accuracy across the operating envelope. This places emphasis on careful parameterization, space-filling sampling, and validation against independent reference sets. In moderately higher dimensions, lookup databases remain efficient, if the database size is chosen to meet the accuracy target, and if sampling designs avoid structured gaps that can lead to misleadingly low error estimates.

Fourth, translating accuracy requirements into design parameters follows a two-stage heuristic based on the observed scaling laws. For the Live FEM workflow, the mesh resolution should be refined until the discretization error is approximately half the target sensor uncertainty, as further refinement increases the latency exponentially with diminishing returns for decision-making. For the lookup database, a starting sample density of roughly ten times the input dimension is widely recommended in the literature for initial sensitivity analysis [39]. However, to achieve engineering accuracy with errors below one percent, our results suggest that sample size must typically increase to scale with the power of the dimension, aligning with the recommendations for space-filling designs [40]. Once dimensionality exceeds the range investigated here, static designs become inefficient, and adaptive sampling strategies become necessary to specifically target nonlinear regions.

Together, these guidelines reduce the workflow selection to four questions: What latency budget must be met, how many online queries are expected over the lifetime of a configuration, what database size is required to meet the accuracy target in the chosen parameterization, and which sampling density ensures sufficient coverage of the operating envelope?

Once the appropriate workflow is selected, its integration into a production environment relies on standard edge computing architectures. In a typical usage scenario, the virtual sensor runs as a containerized microservice on an industrial PC directly at the machine. The deployment complexity depends on the chosen method. The lookup database requires only a lightweight runtime environment and the precomputed index file, making it suitable for resource-constrained controllers. Conversely, the Live FEM workflow necessitates hosting the numerical solver executable and managing corresponding license tokens at the edge.

Operationally, the application subscribes to available sensor signals, such as power consumption or rotational speed, via established communication protocols like OPC UA or MQTT. These raw inputs are mapped to the boundary conditions of the internal model to either query the database or trigger the solver. The computed virtual quantity is subse-

quently transmitted back to the machine control system for immediate process regulation or visualized on the human–machine interface. This architecture allows the machine operator to monitor critical thermal or mechanical states in real time without requiring any specific expertise in numerical simulation or data science.

4.3. Sampling Design and Validation Considerations

Case Study B shows that sampling design and validation strategy can strongly influence the apparent quality of a lookup database sensor. Across the tested configurations, space-filling designs such as maximin Latin hypercube sampling and scrambled Sobol sequences exhibit ground-truth errors that decrease as database size increases. This behavior is consistent with the expectation that improved coverage of the input domain reduces interpolation error for local methods.

Orthogonal array designs behave differently in this setting. They produce very low cross-validation errors while ground-truth errors remain high and show little improvement with additional samples. The combination of structured sampling and fold-based reuse can lead the validation procedure to test predominantly near existing design points, which makes the interpolator appear more accurate than it is over the full operating envelope. This effect is particularly relevant for highly structured designs and local interpolation schemes, where neighborhood relationships are strongly shaped by the design geometry.

For deployment-oriented assessment, cross-validation should therefore be treated as a diagnostic rather than as a stand-alone estimate of global accuracy. When a lookup database is intended to support decisions across a wide range of operating conditions, validation against an independent reference set is necessary to quantify the generalization error. In addition, reporting both cross-validation and ground-truth errors provides a useful consistency check. Large discrepancies between these measures should be interpreted as a warning sign that the design does not provide uniform coverage of the input domain or that the validation protocol does not reflect the intended use.

While this study focused on static and steady-state regimes to isolate fundamental scaling laws, the proposed guidelines apply to dynamic systems with specific considerations. For Live FEM, dynamic analysis increases the per-query runtime t_{fem} linearly with the number of time steps required for integration, making the latency constraint (Guideline 1) even harder to satisfy. In contrast, for the LDB workflow, the transition to dynamics primarily manifests as an increase in input dimensionality, as current states or time-history windows must be included as input parameters. Consequently, the trade-off shifts further in favor of the LDB approach regarding online latency but places even greater emphasis on efficient high-dimensional sampling strategies (Guideline 3) to cover the expanded state-space.

4.4. Limitations and Outlook

This study has limitations that bound how far the reported scaling observations can be generalized.

First, the case studies employ simplified geometries and steady-state or quasi-static physics with smooth input-to-output mappings. While industrial digital twins often involve complex assemblies, the stringent latency requirements of virtual sensors (typically $t < 1$ s) inherently limit the complexity of models deployable as live FEM. The chosen benchmarks are representative of the complexity class that is technically eligible for real-time deployment. However, problems with strong transients, contact, or discontinuities might exhibit different error characteristics and require more conservative validation.

Second, the runtime measurements reflect a specific toolchain and orchestration stack. Absolute values depend on the hardware specifications, including the clock speed and

memory bandwidth, as well as the system load. Consequently, the fitted exponents should be interpreted as empirical behavior for the tested configurations rather than as theoretical solver asymptotics. Nevertheless, the relative performance trends remain primarily governed by the underlying algorithmic complexity of the FE solver ($\mathcal{O}(n^\alpha)$) and the k-d tree search ($\mathcal{O}(\log N)$). This algorithmic basis renders the comparative conclusions robust across similar architectures [41]. We acknowledge that specific bottlenecks, such as I/O latency compared to pure computation, may scale differently on other platforms. This variability could potentially shift the quantitative break-even point without altering the qualitative trade-off. To mitigate transient variability including OS noise, all reported timings for both LFE and LDB workflows represent the median of multiple executions, supplemented by 5th and 95th percentiles to demonstrate stability.

Third, the lookup database implementation uses a fixed k nearest neighbors interpolator and a single indexing strategy. This choice provides a transparent baseline, but it does not explore how interpolation method, neighbor selection, or boundary handling affect error and robustness. Future work should compare alternative local and global interpolators, examine sensitivity to hyperparameters, and evaluate domain-aware strategies that reduce boundary degradation without resorting to extrapolation.

Fourth, the scalability of the SVS-LDB approach is fundamentally constrained by the curse of dimensionality, as the number of samples required to maintain a specific sampling density grows exponentially with input dimension m . This introduces two potential bottlenecks: offline computation time and storage capacity. In the presented case studies, the primary limiting factor is the computation time. With simulation runtimes in the multi-second regime (e.g., approx. 20 s per solve), generating the millions of samples required for a dense high-dimensional grid ($m \gg 9$) becomes computationally intractable long before the storage limit of modern hardware is reached. However, this balance depends on the stored data payload. We focused on condensed quantities of interest (scalars or coarse fields). In Case Study B, this resulted in database sizes ranging from approximately 20 kB ($N = 32$) to 304 kB ($N = 1024$). These values are orders of magnitude below typical RAM capacities, ensuring that the lookup speed is not bottlenecked by I/O. If full high-fidelity solution fields were stored for every design point, the storage capacity would become a competing bottleneck much earlier, necessitating data compression techniques like Proper Orthogonal Decomposition (POD) or advanced manifold learning approaches [42] to ensure tractability.

Finally, the CPU case study reaches nine input dimensions, but higher-dimensional parameterizations are common in practice. In such regimes, uniform databases can become inefficient. Adaptive sampling, hierarchical databases, and dimensionality reduction are promising directions and should be evaluated together with independent reference validation to ensure that the reported errors reflect the intended operating envelope.

5. Conclusions

This work compared two implementation paths for simulation-based virtual sensors, a live FEM workflow and a lookup database workflow. Using a nonlinear tension-bar benchmark (Case Study A, low-dimensional nonlinear mechanics) and a steady-state CPU die model (Case Study B, higher-dimensional steady-state thermal analysis), we quantified accuracy and latency trade-offs, fitted empirical scaling parameters, and derived break-even query counts that relate offline precomputation effort to the expected online query horizon.

Live FEM runtimes increase strongly with the problem size in absolute terms. In Case Study A, the fitted exponent is $\alpha \approx 1.2$ over the tested mesh range. In Case Study B, an effective exponent of $\alpha \approx 0.66$ was observed, which reflects dominant constant overheads in the scripted workflow and solver setup rather than inherently sublinear solver behavior.

In conclusion, the performance of a physics-based virtual sensor depends as much on its software architecture as on its numerical solver. Investing effort into lightweight script-based orchestration and avoiding heavyweight application interfaces can yield significant savings in I/O time. This efficiency gain directly translates into a more favorable break-even point for Live FEM or, alternatively, allows for the generation of denser lookup databases in less time. Across both case studies, refined meshes exceed hard/medium real-time budgets once runtimes enter the multi-second regime.

The lookup database query latency remains close to 0.16 to 0.17 ms across the investigated database sizes and dimensionalities; so, the dominant cost component of this workflow is offline precomputation. Break-even occurs after a finite query horizon expressed relative to database size N , approximately $Q_{be} \approx N$ in the tension-bar case and several multiples of N in the CPU case, with the largest observed factor of about 43 N in this dataset. These results provide a compact basis for workflow selection in practice, where deployment decisions must balance accuracy targets, latency budgets, and the anticipated lifetime query load.

The presented framework is intended to integrate naturally into digital twin and condition monitoring pipelines. In such settings, the offline phase can be aligned with calibration and commissioning, while the derived break-even criteria can be used to decide when repeated live simulations should be replaced by a database-backed sensor for sustained operation. Future work can extend this baseline to broader classes of problems and richer parameterizations, while retaining the same decision logic that ties measurable timing data and accuracy targets to deployment-ready choices.

Author Contributions: Conceptualization, A.K.; methodology, A.K.; software, A.K.; validation, A.K.; formal analysis, A.K.; investigation, A.K. and T.R.; resources, A.K. and S.T.; data curation, A.K.; writing—original draft preparation, A.K.; writing—review and editing, T.R., S.T. and F.R.; visualization, A.K.; supervision, T.R., S.T. and F.R.; project administration, T.R., S.T. and F.R.; funding acquisition, F.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Bayerische Transformations-und Forschungsstiftung (BFS) grant number AZ-1392-19. The APC was funded by the Open Access Publishing Fund of the University of Bayreuth.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All scripts are available on reasonable request. All reference data are available via the linked citations.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SVS	Simulation-based virtual sensor
LFE	Live FEM workflow
LDB	Lookup database
FEM	Finite element method
FE	Finite element
DoF	Degrees of freedom
DoE	Design of experiments
MAE	Mean absolute error
QoI	Quantity of interest
CPU	Central processing unit

References

- Martin, D.; Kühl, N.; Satzger, G. Virtual Sensors. *Bus. Inf. Syst. Eng.* **2021**, *63*, 315–323. [\[CrossRef\]](#)
- Kabadayi, S.; Pridgen, A.; Julien, C. Virtual Sensors: Abstracting Data from Physical Sensors. In *Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)*, Buffalo-Niagara Falls, NY, USA, 26–29 June 2006; IEEE: New York, NY, USA, 2006; pp. 587–592, ISBN 0-7695-2593-8.
- Wu, B.; Widanage, W.D.; Yang, S.; Liu, X. Battery digital twins: Perspectives on the fusion of models, data and artificial intelligence for smart battery management systems. *Energy AI* **2020**, *1*, 100016. [\[CrossRef\]](#)
- Liu, C.; Zhang, P.; Xu, X. Literature review of digital twin technologies for civil infrastructure. *J. Infrastruct. Intell. Resil.* **2023**, *2*, 100050. [\[CrossRef\]](#)
- Albertos, P.; Goodwin, G.C. Virtual sensors for control applications. *Annu. Rev. Control* **2002**, *26*, 101–112. [\[CrossRef\]](#)
- Roque Rolo, G.; Dionisio Rocha, A.; Tripa, J.; Barata, J. Application of a Simulation-Based Digital Twin for Predicting Distributed Manufacturing Control System Performance. *Appl. Sci.* **2021**, *11*, 2202. [\[CrossRef\]](#)
- Kritzing, W.; Karner, M.; Traar, G.; Henjes, J.; Sihn, W. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC—PapersOnLine* **2018**, *51*, 1016–1022. [\[CrossRef\]](#)
- Tao, F.; Zhang, H.; Liu, A.; Nee, A.Y.C. Digital Twin in Industry: State-of-the-Art. *IEEE Trans. Ind. Inf.* **2019**, *15*, 2405–2415. [\[CrossRef\]](#)
- National Academies Press (US). *Foundational Research Gaps and Future Directions for Digital Twins*; National Academies Press: Washington, DC, USA, 2024; ISBN 9780309700429.
- Marinkovic, D.; Zehn, M. Survey of Finite Element Method-Based Real-Time Simulations. *Appl. Sci.* **2019**, *9*, 2775. [\[CrossRef\]](#)
- Sun, Z.; Jayasinghe, S.; Sidiq, A.; Shahrivar, F.; Mahmoodian, M.; Setunge, S. Approach Towards the Development of Digital Twin for Structural Health Monitoring of Civil Infrastructure: A Comprehensive Review. *Sensors* **2024**, *25*, 59. [\[CrossRef\]](#)
- Rama, G.; Marinković, D.; Zehn, M. Efficient three-node finite shell element for linear and geometrically nonlinear analyses of piezoelectric laminated structures. *J. Intell. Mater. Syst. Struct.* **2018**, *29*, 345–357. [\[CrossRef\]](#)
- Yoon, S.; Choi, Y.; Koo, J. In situ virtual sensors in building digital twins: Framework and methodology. *J. Ind. Inf. Integr.* **2023**, *36*, 100532. [\[CrossRef\]](#)
- Hoang, P.; Guccione, J. Finite Element Modeling in Left Ventricular Cardiac Biomechanics: From Computational Tool to Clinical Practice. *Bioengineering* **2025**, *12*, 913. [\[CrossRef\]](#)
- Llopis-Albert, C.; Rubio, F.; Devese, C.; García-Hurtado, D. Digital Twin-Based Approach for a Multi-Objective Optimal Design of Wind Turbine Gearboxes. *Mathematics* **2024**, *12*, 1383. [\[CrossRef\]](#)
- Encalada-Dávila, Á.; Vidal, Y.; Tutivén, C. Strain virtual sensor for offshore wind turbine jacket supports: A time series transformer approach validated with Alpha Ventus wind farm data. *Mech. Syst. Signal Process.* **2025**, *231*, 112653. [\[CrossRef\]](#)
- Bashir, A.; Mohsin, M.A.; Jazib, M.; Iqbal, H. MINDTwin AI: Multiphysics Informed Digital-Twin for Fault Localization in Induction Motor Using AI. In *Proceedings of the 2023 International Conference on Big Data, Knowledge and Control Systems Engineering (BdKCSE)*, Sofia, Bulgaria, 2–3 November 2023; IEEE: New York, NY, USA, 2023; pp. 1–8, ISBN 979-8-3503-1324-6.
- Hossain, R.; Ahmed, F.; Kobayashi, K.; Koric, S.; Abueidda, D.; Alam, S.B. Virtual sensing-enabled digital twin framework for real-time monitoring of nuclear systems leveraging deep neural operators. *npj Mater. Degrad.* **2025**, *9*, 21. [\[CrossRef\]](#)
- Kasilingam, S.; Yang, R.; Singh, S.K.; Farahani, M.A.; Rai, R.; Wuest, T. Physics-based and data-driven hybrid modeling in manufacturing: A review. *Prod. Manuf. Res.* **2024**, *12*, 2305358. [\[CrossRef\]](#)
- Kapusuzoglu, B.; Mahadevan, S. Physics-Informed and Hybrid Machine Learning in Additive Manufacturing: Application to Fused Filament Fabrication. *JOM* **2020**, *72*, 4695–4705. [\[CrossRef\]](#)
- Vandewiel, M.R.; Eneyew, D.D.; Awol, A.D.; Capretz, M.A.; Bitsuamlak, G.T. Approximating CFD simulations of natural ventilation: A deep surrogate model with spatial attention mechanism. *J. Build. Eng.* **2025**, *105*, 112425. [\[CrossRef\]](#)
- Ogunniyi, E.A.; Drnek, C.; Hong, S.H.; Downey, A.R.; Wang, Y.; Bakos, J.D.; Avitabile, P.; Dodson, J. Real-time structural model updating using local eigenvalue modification procedure for applications in high-rate dynamic events. *Mech. Syst. Signal Process.* **2023**, *195*, 110318. [\[CrossRef\]](#)
- Zhao, Y.; Zhang, G.; Zang, G.; Zhang, G.; Sang, W.; Zhang, S.; Li, W. Monitoring Bridge Dynamic Deformation Law Based on Digital Photography and Ground-Based RAR Technology. *Appl. Sci.* **2023**, *13*, 10838. [\[CrossRef\]](#)
- Zhang, K.; Qing, J.; Jin, H.; Jin, H. Digital Twin-Enabled Predictive Thermal Modeling for Stator Temperature Monitoring in Induction Motors. *Electronics* **2025**, *14*, 2814. [\[CrossRef\]](#)
- Glamsch, J.; Rosnitschek, T.; Rieg, F. Initial Population Influence on Hypervolume Convergence of NSGA-III. *Int. J. Simul. Model* **2021**, *20*, 123–133. [\[CrossRef\]](#)
- Garud, S.S.; Karimi, I.A.; Kraft, M. Smart Sampling Algorithm for Surrogate Model Development. *Comput. Chem. Eng.* **2017**, *96*, 103–114. [\[CrossRef\]](#)
- Renardy, M.; Joslyn, L.R.; Millar, J.A.; Kirschner, D.E. To Sobol or not to Sobol? The effects of sampling schemes in systems biology applications. *Math. Biosci.* **2021**, *337*, 108593. [\[CrossRef\]](#) [\[PubMed\]](#)

28. Vabalas, A.; Gowen, E.; Poliakoff, E.; Casson, A.J. Machine learning algorithm validation with a limited sample size. *PLoS ONE* **2019**, *14*, e0224365. [[CrossRef](#)] [[PubMed](#)]
29. Rosnitschek, T.; Stierle, C.; Orgeldinger, C.; Seynstahl, A.; Alber-Laukant, B.; Tremmel, S. Dimensional Accuracy and Mechanical Characterization of Inconel 625 Components in Atomic Diffusion Additive Manufacturing. *Appl. Mech.* **2024**, *5*, 376–390. [[CrossRef](#)]
30. DIN EN ISO 6892-1:2020-06; Metallic Materials—Tensile Testing—Part 1: Method of Test at Room Temperature (ISO 6892-1:2019); German Version EN ISO 6892-1:2019. DIN Media: Berlin, Germany, 2020. [[CrossRef](#)]
31. Bathe, K.-J. *Finite Element Procedures*, 2nd ed.; K. J. Bathe: Watertown, MA, USA, 2014; ISBN 978-0-9790049-5-7.
32. Zhang, J.; Sadiqbatcha, S.; O’Dea, M.; Amrouch, H.; Tan, S.X.-D. Full-Chip Power Density and Thermal Map Characterization for Commercial Microprocessors Under Heat Sink Cooling. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2022**, *41*, 1453–1466. [[CrossRef](#)]
33. Bergman, T.L.; Lavine, A.S. *Fundamentals of Heat and Mass Transfer*, 8th ed.; Wiley: Hoboken, NJ, USA, 2019; ISBN 978-1119722489.
34. Raicu, I.; Foster, I.T.; Zhao, Y. Many-task computing for grids and supercomputers. In *Proceedings of the 2008 Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS)*, Austin, TX, USA, 17 November 2008; IEEE: New York, NY, USA, 2008; pp. 1–11, ISBN 978-1-4244-2872-4.
35. Phellan, R.; Hachem, B.; Clin, J.; Mac-Thiong, J.-M.; Duong, L. Real-time biomechanics using the finite element method and machine learning: Review and perspective. *Med. Phys.* **2021**, *48*, 7–18. [[CrossRef](#)]
36. Kasim, M.F.; Watson-Parris, D.; Deaconu, L.; Oliver, S.; Hatfield, P.; Froula, D.H.; Gregori, G.; Jarvis, M.; Khatiwala, S.; Korenaga, J.; et al. Building high accuracy emulators for scientific simulations with deep neural architecture search. *Mach. Learn. Sci. Technol.* **2022**, *3*, 15013. [[CrossRef](#)]
37. Quarteroni, A.; Manzoni, A.; Negri, F. *Reduced Basis Methods for Partial Differential Equations: An Introduction*; Springer: Cham, Switzerland, 2015; ISBN 978-3319154305.
38. Muja, M.; Lowe, D.G. Scalable Nearest Neighbor Algorithms for High Dimensional Data. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2227–2240. [[CrossRef](#)]
39. Loepky, J.L.; Sacks, J.; Welch, W.J. Choosing the Sample Size of a Computer Experiment: A Practical Guide. *Technometrics* **2009**, *51*, 366–376. [[CrossRef](#)]
40. Forrester, A.I.; Keane, A.J. Recent advances in surrogate-based optimization. *Progress. Aerosp. Sci.* **2009**, *45*, 50–79. [[CrossRef](#)]
41. Hoefler, T.; Belli, R. Scientific benchmarking of parallel computing systems. In *Proceedings of the SC15: The International Conference for High Performance Computing, Networking, Storage and Analysis*, Austin, TX, USA, 15–20 November 2015; Kern, J., Vetter, J.S., Eds.; ACM: New York, NY, USA, 2015; pp. 1–12, ISBN 9781450337236.
42. Scheunemann, L.; Faust, E. A manifold learning approach to nonlinear model order reduction of quasi-static problems in solid mechanics: Proof of concept and parameter study. *Comput. Mech.* **2025**, online ahead of print. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.