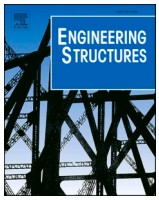
Engineering Structures 326 (2025) 119520

Contents lists available at ScienceDirect

Engineering Structures

journal homepage: www.elsevier.com/locate/engstruct



Adaptive edge intelligence for rapid structural condition assessment using a wireless smart sensor network

Shuaiwen Cui [a], Tu Hoang [b], Kirill Mechitov [c], Yuguang Fu [a,*], Billie F. Spencer Jr. [d]

[a] School of Civil and Environmental Engineering, Nanyang Technological University, 50 Nanyang Ave, 639798, Singapore
[b] Geocomp, 125 Nagog Park, Acton, MA 01720, USA
[c] Embedor Technologies, 1800 S Oak St, Ste. 202B, Champaign, IL 61820, USA
[d] Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 N. Matthews Ave, Urbana, IL 61801, USA

ARTICLE INFO

Keywords:
Structural health monitoring
Wireless smart sensor network
Edge intelligence
Anomaly detection
Gaussian process regression
Reference-free displacement estimation

ABSTRACT

Combining artificial intelligence and edge computing, edge intelligence is a promising computing paradigm for the Internet-of-Things-based Structural Health Monitoring (SHM), showing great potential to improve system responsiveness by reducing communication latency. Previously, very limited studies proposed, optimized, or verified edge intelligence approaches for SHM applications, where the overhead and efficiency of algorithms to manage limited onboard resources are the main gaps. In this study, an adaptive edge intelligence strategy is proposed to facilitate autonomous structural condition assessment, involving reference-free displacement estimation algorithm, Gaussian Process Regression, and stochastic process control. To facilitate algorithm deployment, both effective single-node independent computing and multi-node coordination are explored to deal with the limited onboard resources, utilizing the computing capacity of each node to speed up computation. Using the Xnode, a MEMS-based wireless sensor platform, lab tests and full-scale applications in railroad bridge monitoring were conducted to verify the proposed strategy, demonstrating the potential and suitability of the developed approach for rapid adaptive structural condition assessment in SHM practice.

1. Introduction

For structure condition assessment, engineers usually identify suitable indicators and then select suitable devices to monitor [1]. One of the objectives of using SHM is to capture pertinent information when damage occurs [2], in which the response speed is critical. In most cases, abrupt structural damage is indicated by anomalies [3]. Therefore, for structural condition assessment, an effective idea is to determine appropriate indicators and use suitable approaches to capture anomalies indicating potential damage [2,4] and then to trigger alarms.

Given the prevalence of accelerometers [5], acceleration is a common measurand [6], while displacement is widely recognized as an important indicator as it is quite intuitive to make informed decisions [7]. Therefore, efficient approaches to convert acceleration to displacement are desirable. Integration-based conversion approaches require a reference to determine the constants, making them less useful for real-world applications. A promising reference-free dynamic displacement estimation algorithm to be introduced below [8] makes this conversion much more practical, which can be reduced to a Finite Impulse Response (FIR) filter design problem. With the filter generated, raw acceleration can be converted to dynamic displacement. Subsequently, anomaly detection algorithms/strategies are desired to enable rapid condition assessment based on obtained displacement. Unlike model-based anomaly detection methods, data-driven ones are less demanding in terms of knowledge and are straightforward to express explicitly [3,9,10]. For cases where the available dataset is of small or moderate size, the Gaussian Process Regression (GPR) method is ideal, of which the objective is to use machine learning to find a function that can fit the observed data well and make high-quality predictions. GPR features many advantages for anomaly detection [11–14]: firstly, GPR is flexible and can fit functions of any shape; secondly, GPR can provide a probabilistic distribution to quantify the uncertainty; thirdly, GPR is robust to noise and outliers. Since the GPR model can give a probability distribution along with the prediction value, by comparing to the ground truth, stochastic process control (SPC) can be leveraged to trigger alarms. Given the reasons above, reference-free dynamic displacement estimation and GPR can be an ideal combination for anomaly detection, including edge computing cases which have been rarely explored, e.g.,

* Corresponding author.
E-mail address: yuguang.fu@ntu.edu.sg (Y. Fu).

https://doi.org/10.1016/j.engstruct.2024.119520
Received 24 June 2024; Received in revised form 5 November 2024; Accepted 13 December 2024
Available online 18 December 2024

onboard anomaly detection in wireless sensor networks (WSN) or Internet-of-Things (IoT) setups.

While theoretical foundations are important, their practical implementation in SHM systems is even more critical for engineering applications [15]. By significantly reducing power consumption and latency, the emergence of the micro-electromechanical-system (MEMS) WSN [16–18] has profoundly changed SHM, laying down the foundation for edge intelligence which can harness the advantages of both edge computing [10] and Artificial Intelligence (AI). On the one hand, compared to the conventional cloud or centralized computing, edge computing features decentralized computing capability, enabling more onboard computation and lower communication bandwidth and therefore lower overall power consumption and communication latency [19]. On the other hand, compared to model-based approaches, AI significantly lowers the barrier of modeling by enabling the model to learn from data automatically [20]. This paper aims to leverage edge intelligence in WSN to achieve rapid structural condition assessment adaptable to the changing conditions. With edge intelligence, the anomaly detection algorithm can utilize the computing resources of the whole network, speeding up the computation process significantly. Meanwhile, the communication payload can be condensed from the bulky raw data to compact actionable information, significantly reducing power consumption and associated latency. In short, with the proper arrangement, edge computing can lead to a higher level of autonomy, longer lifespan and better responsiveness of deployed WSN.

Though featuring attractive merits, edge computing faces significant challenges at the same time, with constrained resources being the biggest challenge [19,21], including power, memory, storage, processing speed, etc. For wireless sensing, communication is generally the most power-hungry task [22]. Therefore, reducing communication overhead can significantly help to save energy and prolong the lifespan [19], which comes at the cost of increased onboard computation for data compression and information distillation, constrained by processing speed, memory, and storage. Compared to the energy saved, the trade-off, i.e., more onboard computation, is considered worthy and necessary. In this study, a series of novel measurements are taken to tackle the aforementioned challenges so that the limited computational resources can be properly utilized to achieve adaptive onboard structural condition assessment.

Compared to prior work, the novelty and contributions of this study reside in the following points: reference-free dynamic displacement estimation and GPR are combined for lightweight adaptive onboard anomaly detection and hence rapid condition assessment; the proposed strategy is streamlined for onboard realization, using effective measures to deal with the limited resources, involving both single-node computation and multi-node coordination; the proposed strategy and its realization on sensors were validated with lab tests and full-scale field applications data. The remainder of this paper is structured as follows: Section 2 elaborates the proposed structural condition assessment strategy with sensitivity analyses; Section 3 presents technical details of an edge computing framework for the proposed strategy along with verifications; Section 4 demonstrates the application and validation details using full-scale railroad bridge data; Section 5 concludes this paper.

2. Structural condition assessment strategy

2.1. Reference-free dynamic displacement estimation

This algorithm aims to convert acceleration into dynamic displacement without requiring a reference, achieved by convolving the acceleration data with a derived FIR filter. On the one hand, accelerometers are widely used, but displacement data is generally more intuitive; on the other hand, traditional double integration approaches often encounter challenges to determine the constants. The reference-free displacement estimation algorithm used here transforms the problem of calculating dynamic displacement from one of integration into an optimization problem [8,23].

As shown in Eq. (1), the optimization function consists of two terms: the first term represents the least-squares error between the second-order derivative of the estimated displacement and the measured acceleration data, while the second term is the Tikhonov regularization (or Ridge Regression), which involves the square of the displacement scaled by a balancing factor β. By minimizing Eq. (1) [8,23], the equation forces the second-order derivative of displacement $u(t)$ to align closely with the measured acceleration data $\bar{a}$, while the second term modulates the effect of noise or boundary conditions by adjusting β, thereby suppressing the influence of large values caused by noise or boundary conditions. Eq. (1) is an extended, high-order form of the original optimization function, designed to address practical issues related to differentiation and smoothness in low-order form [8,23].

$$\Pi(u) = \frac{1}{2}\int_{T_1}^{T_2} \frac{d^{n-2}}{dt^{n-2}}\left(\frac{d^2u}{dt^2} - \bar{a}\right)^2 dt + \frac{1}{2}\beta^n \int_{T_1}^{T_2} u^2 dt \tag{1}$$

Where u stands for the estimated displacement, $\bar{a}$ stands for measured acceleration, β stands for the Tikhonov factor to balance the degree to suppress noise and boundary effect. T_1 and T_2 are the starting and end points of the period for displacement estimation. To obtain the minimum of formula (1), first-order variation can be calculated and set to zero, leading to formula (2).

$$\frac{d^{2n}u}{dt^{2n}} + (-\beta)^n u = \frac{d^{2n-2}\bar{a}}{dt^{2n-2}}, \quad T_1 < t < T_2 \tag{2}$$

Applying the Fourier Transform to formula (2), an accuracy function can be derived as formula (3), where ω stands for frequency and i is the imaginary unit. Details can be found in the literature [8,23].

$$H^{accu}(\omega) = |(i\omega)^2 H_{u\bar{a}}(\omega)| = \frac{\omega^{2n}}{\omega^{2n} + \beta^n} \tag{3}$$

In practice, a process of parameter optimization is necessary, including the $H^{accu}(\omega)$, the regularization factor β, and filter order n [7]. After that, to guarantee the stability of the filter, a digital FIR filter type I [8,23] with a general linear phase is used. Taking the relationship among the discretized estimated displacement, measured acceleration data, and the accuracy function, a function for the coefficients of the FIR filter can be derived as formula (4).

$$c_{p+k+1} = \frac{f_s}{2\pi^2}\int_0^{f_s/2} \frac{f^{2n-2}}{f^{2n} + \lambda^{2n} f_T^{2n}} \cos(2\pi p f \Delta t) df \tag{4}$$

Where p is an arbitrary integer within the range from 0 to k, λ is a balancing factor, and the window length $N_w = 2k+1$ depends on the target frequency f_T :

$$N_w = \frac{2kf_T}{f_s} \tag{5}$$

By conducting the process above, each element c_{p+k+1} of the FIR filter can be derived for the convolution operation in dynamic displacement estimation. Applying this filter to the acceleration signal allows for the calculation of the corresponding displacement data. The maximum values from these displacement datasets serve as a foundation for the GPR discussed in the next section for anomaly detection.

2.2. Gaussian process regression and stochastic process control

GPR is a non-parametric Bayesian machine learning technique [24] that can effectively model observed data and make accurate predictions, forming the basis for anomaly detection in this context. The term 'non-parametric' indicates that the model automatically learns from the

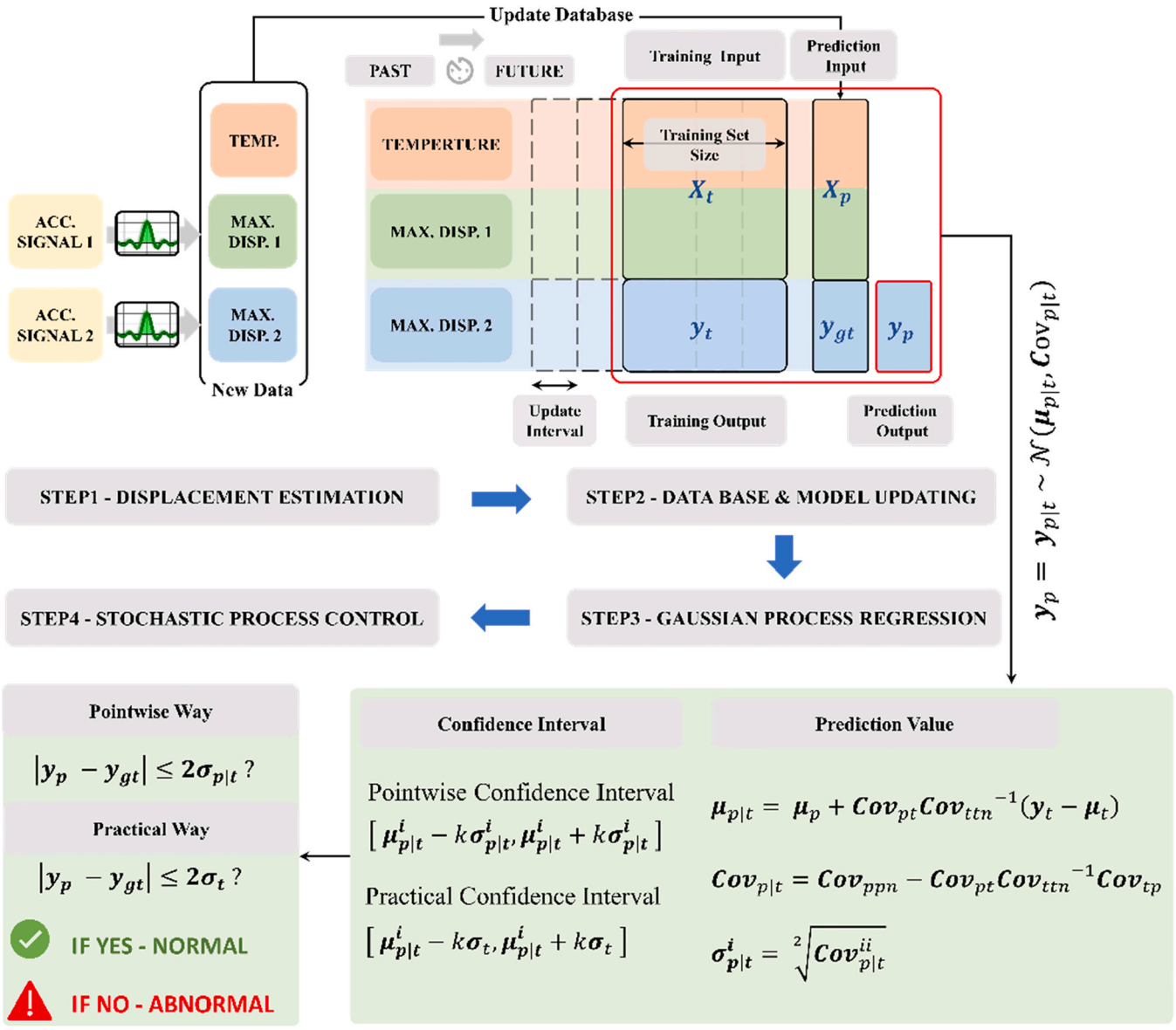


Fig. 1. Schematic illustration of the proposed anomaly detection strategy.

data and can adapt to fit functions of any shape. 'Bayesian' implies that it uses Bayesian inference, providing quantified uncertainty [25]. These characteristics make GPR an excellent choice for data-driven structural condition assessment on edge devices in SHM. Following this, SPC can be applied to compare the residuals—i.e., the difference between predictions and ground truth—against the derived confidence interval to assess potential damage.

A Gaussian Process (GP) is defined as a collection of random variables, any finite number of which form a joint Gaussian distribution [14]. On a function space, the input variable can be denoted as $\boldsymbol{x} \in R^d$, then its corresponding output can be denoted as $f(\boldsymbol{x})$. Gaussian process $f(\boldsymbol{x})$ can be fully specified with a mean function $mean(\boldsymbol{x})$ and a covariance function $\mathrm{cov}(\boldsymbol{x}, \boldsymbol{x}')$:

$$mean(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})] \tag{6}$$

$$\mathrm{cov}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f(\boldsymbol{x}) - mean(\boldsymbol{x}))(f(\boldsymbol{x}') - mean(\boldsymbol{x}'))] \tag{7}$$

$$f(\boldsymbol{x}) \sim \mathcal{GP}(mean(\boldsymbol{x}), \mathrm{cov}(\boldsymbol{x}, \boldsymbol{x}')) \tag{8}$$

The formula (8) means that the random variables represent the values of function $f(\boldsymbol{x})$ at location $\boldsymbol{x}$, and follow a Gaussian process specified by a mean function $mean(\boldsymbol{x})$ and a covariance function $\mathrm{cov}(\boldsymbol{x}, \boldsymbol{x}')$. Note that $\boldsymbol{x} \in R^d$ here is a vector, standing for a point in a d dimensional space. Usually, for prediction, there are datasets for training and for prediction: for training, there are input data $\boldsymbol{X}_t \in R^{n_t \times d}$, output data $\boldsymbol{y}_t = f(\boldsymbol{X}_t) \in R^{n_t \times 1}$; for prediction, there are input data $\boldsymbol{X}_p \in R^{n_p \times d}$, predicted output data $\boldsymbol{y}_p = f(\boldsymbol{X}_p) \in R^{n_p \times 1}$, and ground truth output $\boldsymbol{y}_{gt} \in R^{n_p \times 1}$. Note that only $\boldsymbol{y}_p$ is to be determined. Leveraging the properties in the definition of GP, the joint distribution of $\boldsymbol{y}_t$ and $\boldsymbol{y}_p$ is a multivariate Gaussian distribution. To make the model more robust, the noise is considered, yielding the following formula. Note that noise level ε is a hyperparameter to be fine-tuned.

$$\begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{y}_p \end{bmatrix} \sim N\left(\begin{bmatrix} \boldsymbol{\mu}_t \\ \boldsymbol{\mu}_p \end{bmatrix}, \begin{bmatrix} \boldsymbol{Cov}_{ttn} & \boldsymbol{Cov}_{tp} \\ \boldsymbol{Cov}_{pt} & \boldsymbol{Cov}_{ppn} \end{bmatrix} \right) = N\left(\begin{bmatrix} \boldsymbol{\mu}_t \\ \boldsymbol{\mu}_p \end{bmatrix}, \begin{bmatrix} \boldsymbol{Cov}_{tt} + \varepsilon^2 \boldsymbol{I}_t & \boldsymbol{Cov}_{tp} \\ \boldsymbol{Cov}_{pt} & \boldsymbol{Cov}_{pp} + \varepsilon^2 \boldsymbol{I}_p \end{bmatrix} \right) \tag{9}$$

The formula above stands for the a priori. However, what we are really interested in is the a posteriori after incorporating the knowledge from observation, i.e., the conditional distribution $\boldsymbol{y}_p | \boldsymbol{X}_t, \boldsymbol{y}_t$,

$\boldsymbol{X}_p$ calculated based on the observed training set $\{\boldsymbol{X}_t^i, \boldsymbol{y}_t^i | i = 1, 2, 3, \ldots, n_t\}$. With proper assumption for the noise level ε and the values for $\boldsymbol{\mu}_p$ ($\boldsymbol{\mu}_p^i = \overline{\boldsymbol{\mu}_t} = \mu, i = 1, 2, 3, \ldots, n_p$), the joint distribution of $\boldsymbol{y}_t$ and $\boldsymbol{y}_p$ can be considered as known. For conditional distribution with known joint distribution, the Bayesian Theorem can be used, yielding subsequent equations [14]. More specifically, for displacement-based anomaly detection: the prior is that the distribution of monitoring point displacements follows a Gaussian distribution; the evidence (or observations) refers to the displacements of the monitoring points used for baseline model fitting (or training), i.e., $(\boldsymbol{X}_t, \boldsymbol{y}_t)$; the posterior is the prediction based on the test input and the model, namely, $\boldsymbol{y}_p | \boldsymbol{X}_t, \boldsymbol{y}_t, \boldsymbol{X}_p$. Note that $\boldsymbol{Cov}_{ppn}$ can be reduced to $\boldsymbol{Cov}_{pp}$ if the noise for prediction is not considered.

$$\boldsymbol{y}_p | \boldsymbol{X}_t, \boldsymbol{y}_t, \boldsymbol{X}_p \sim N\Big(\boldsymbol{\mu}_p + \boldsymbol{Cov}_{pt}\boldsymbol{Cov}_{ttn}^{-1}(\boldsymbol{y}_t - \boldsymbol{\mu}_t), \boldsymbol{Cov}_{ppn} - \boldsymbol{Cov}_{pt}\boldsymbol{Cov}_{ttn}^{-1}\boldsymbol{Cov}_{tp}\Big) \tag{10}$$

Formula (10) represents the conditional distribution of $\boldsymbol{y}_p$. For short, note $\boldsymbol{y}_p | \boldsymbol{X}_t, \boldsymbol{y}_t, \boldsymbol{X}_p$ as $\boldsymbol{y}_{p|t}$, $\boldsymbol{\mu}_p + \boldsymbol{Cov}_{pt}\boldsymbol{Cov}_{ttn}^{-1}(\boldsymbol{y}_t - \boldsymbol{\mu}_t)$ as $\boldsymbol{\mu}_{p|t}$, $\boldsymbol{Cov}_{ppn} - \boldsymbol{Cov}_{pt}\boldsymbol{Cov}_{ttn}^{-1}\boldsymbol{Cov}_{tp}$ as $\boldsymbol{Cov}_{p|t}$. $\boldsymbol{\mu}_{p|t}$ stands for the mean values for $\boldsymbol{y}_{p|t}$, and can be used as prediction for $\boldsymbol{y}_{p|t}$; $\boldsymbol{Cov}_{p|t}$ is the covariance matrix for $\boldsymbol{y}_{p|t}$, where the diagonal is the square of standard deviation of $\boldsymbol{y}_{p|t}$, as shown in formula (11).

$$\boldsymbol{\sigma}_{\boldsymbol{p}|\boldsymbol{t}}^{\boldsymbol{i}} = \sqrt[2]{\boldsymbol{Cov}_{p|t}^{ii}}, \quad i = 1, 2, 3, \ldots, n_p \tag{11}$$

With the short notation, formula (10) can be expressed as the following formulae.

$$\boldsymbol{y}_p = \boldsymbol{y}_{p|t} \sim N\Big(\boldsymbol{\mu}_{p|t}, \boldsymbol{Cov}\Big) \tag{12}$$

$$\boldsymbol{\mu}_{p|t} = \boldsymbol{\mu}_p + \boldsymbol{Cov}_{pt}\boldsymbol{Cov}_{ttn}^{-1}(\boldsymbol{y}_t - \boldsymbol{\mu}_t) \tag{13}$$

$$\boldsymbol{Cov}_{p|t} = \boldsymbol{Cov}_{ppn} - \boldsymbol{Cov}_{pt}\boldsymbol{Cov}_{ttn}^{-1}\boldsymbol{Cov}_{tp} \tag{14}$$

So far, the last missing piece is the kernel function, where the true power of GPR resides [26]. By manipulating the type, parameters, and the combination of kernels, engineers can control the behavior of GPR. For simplicity, the classic RBF kernel is adopted, as shown in Formula (15).

$$\operatorname{cov}(\boldsymbol{x}_i, \boldsymbol{x}_j) = a^2 e^{-\frac{1}{2}\left[\sum_{m=1}^{d}\left(\frac{x_i^m - x_j^m}{l^m}\right)^2\right]}, m \in \{1, 2, \ldots, d\} \tag{15}$$

Formula (15) defines the kernel value between any two entries from the input data set $\boldsymbol{X}$; a is a hyperparameter controlling average distance away from the mean function; $\boldsymbol{l}$ stands for the length scale, determining the reach of influence of each point on neighbors. One can maximize the marginal log-likelihood function or try a grid search to optimize these two hyperparameters.

The key idea of SPC is to keep the variable within a specified range. For anomaly detection, the ground truth values falling into the confidence interval $\left[\boldsymbol{\mu}_{p|t} - k\boldsymbol{\sigma}_{p|t}, \boldsymbol{\mu}_{p|t} + k\boldsymbol{\sigma}_{p|t}\right]$ are considered as normal, otherwise abnormal, where typically $k = 2 or 3$. This check is equivalent to check whether $|\boldsymbol{y}_{gt} - \boldsymbol{y}_{p|t}| < k\boldsymbol{\sigma}_{p|t}$ is true. It should be noted that, in practice, the standard deviation of the training output $\boldsymbol{\sigma}_t$, usually the averaged value, is used to replace $\boldsymbol{\sigma}_{p|t}$ for simplicity.

2.3. The anomaly detection and condition assessment strategy

2.3.1. Strategy overview

In essence, the structural condition assessment strategy involves using GPR and SPC (as introduced in Section 2.2) to detect anomalies in the maximum displacement estimated from acceleration data (with the reference-free approach from Section 2.1). As shown in Fig. 1, the first step is to estimate dynamic displacement from different sources and obtain the ambient temperature; the second is to update the database for the subsequent training of the GPR model as baseline; the third is to conduct GPR and input test data to obtain the prediction and confidence interval; and the final is to conduct SPC to determine whether any points fall beyond the control limits. For illustrative purposes, the system is assumed to have two acceleration signal sources and one temperature signal source.

2.3.2. Dynamic displacement estimation and adaptive model updating

This subsection aims to provide details of the first two steps of the proposed strategy. The first step, displacement estimation, is to convert the raw acceleration data into dynamic displacement. Note that temperature data is also obtained in this step because it is highly related to the performance of the structure [15], though it is optional. For the second step, the database is updated, and the part for GPR model training and inference can be updated when required.

As shown in Fig. 1, the input data is assumed to be temperature and acceleration signals from several independent sources. With the filter calculated from the algorithm in Section 2.1, acceleration data can be converted to dynamic displacement through convolution. Afterwards, the maximum value of the displacement data is extracted and used to update the database and the GPR model.

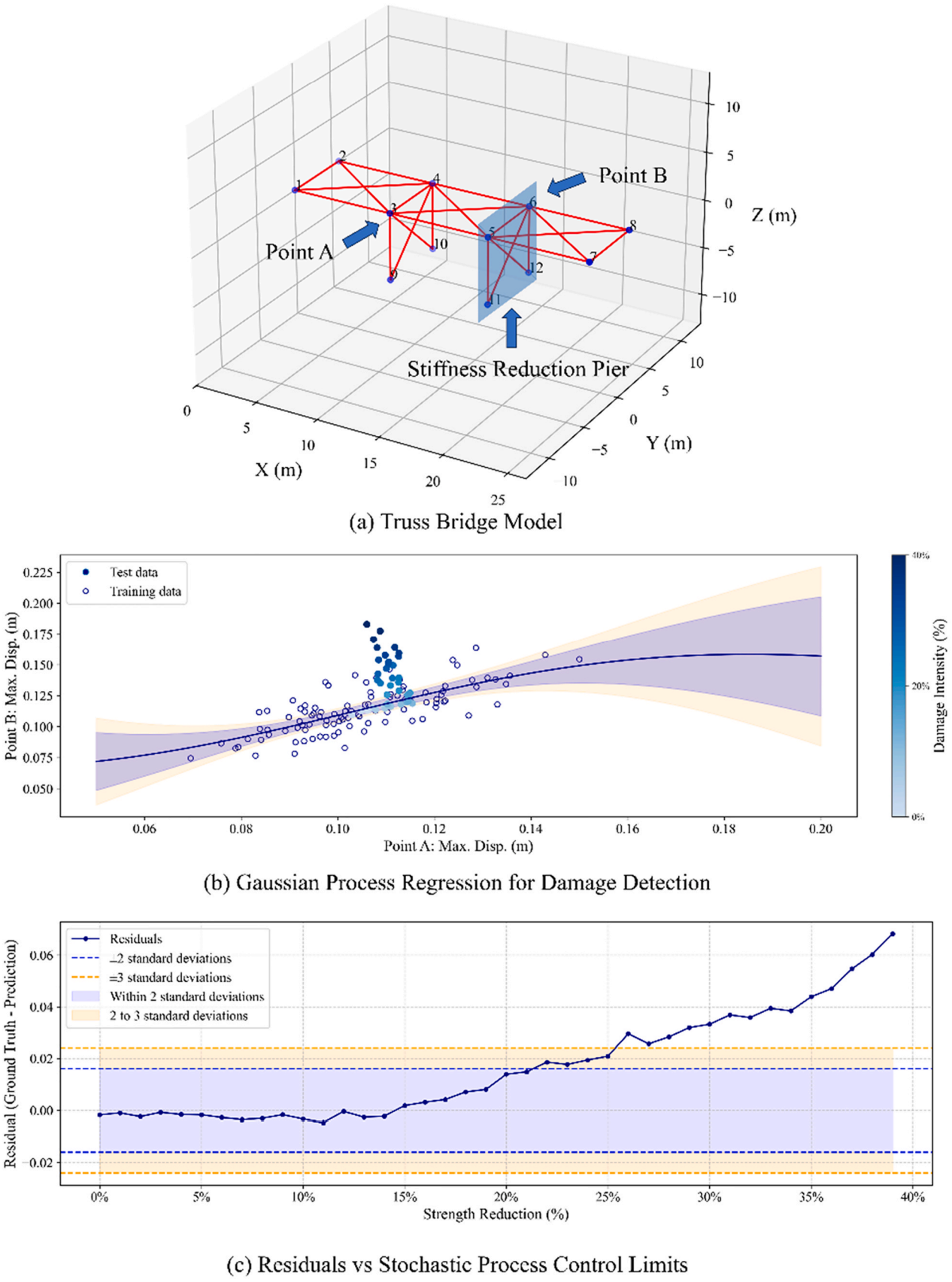
In the database, for the training part, temperature data and maximum displacement from source 1 are selected as input data, denoted as $\boldsymbol{X}_t$, and maximum displacement from source 2 is set as the output data, denoted as $\boldsymbol{y}_t$. For the prediction part, temperature, and the maximum displacement from source 1 are used as input, denoted as $\boldsymbol{X}_p$, and the maximum displacement of source 2 is the data to predict, denoted as $\boldsymbol{y}_p$ (from the Bayesian inference view, $\boldsymbol{y}_p = \boldsymbol{y}_{p|t}$). Meanwhile, the ground truth data for $\boldsymbol{y}_p$ is denoted as $\boldsymbol{y}_{gt}$. To consider structure changes as time goes on, a model updating scheme is incorporated to achieve adaptivity. As shown in Fig. 1, the dashed rectangles stand for obsolete GPR training datasets, and the solid rectangle stands for the current. Every time the total number of samples reaches an engineer-determined value (e.g., 50), the training set head moves forward by that number. Note that, by default, the head of the history dataset is the data to predict, i.e., $\boldsymbol{y}_p$, and the training set is always behind the history record head. To avoid the cases where the target structure slowly degrades and the anomaly detection mechanism fails to detect potential damage, hard limits can be set for the measurand as a fail-safe, i.e., acceleration in this case.

2.3.3. GPR and SPC for anomaly detection

This subsection covers the last two steps of the proposed strategy. GPR calculation is to compute the prediction value ($\boldsymbol{y}_p$) and associated confidence interval ($\left[\boldsymbol{\mu}_{p|t} - 2\boldsymbol{\sigma}_{p|t}, \boldsymbol{\mu}_{p|t} + 2\boldsymbol{\sigma}_{p|t}\right]$), while the SPC calculation is to compare the difference between the prediction value and the ground truth value ($|\boldsymbol{y}_p - \boldsymbol{y}_{gt}|$) to determine whether there is an anomaly.

For GPR prediction computation at desired points, the mean value ($\boldsymbol{\mu}_{p|t}$) of the expected output can be taken as the prediction value, as shown in formula (13). For GPR confidence interval computation, its nature is to calculate the standard deviation at the corresponding points. The pointwise standard deviation ($\boldsymbol{\sigma}_{\boldsymbol{p}|\boldsymbol{t}}^{\boldsymbol{i}}$, $i = 1, 2, 3, \ldots, n_p$) can be obtained from formula (14) and formula (11) in sequence. Therefore, the prediction values $\boldsymbol{\mu}_{p|t}$ and their associated confidence interval $\left[\boldsymbol{\mu}_{p|t} - k\boldsymbol{\sigma}_{p|t}, \boldsymbol{\mu}_{p|t} + k\boldsymbol{\sigma}_{p|t}\right]$ can be obtained, typically, $k = 2$ or 3, standing for confidence levels of 95.45 % or 99.73 %, respectively. In practice, for simplicity in calculation, engineers also use the averaged standard deviation of the training set ($\boldsymbol{\sigma}_t$) as a practical value for anomaly detection.

The key idea of SPC is to compare the difference between the ground



(a) Truss Bridge Model

(b) Gaussian Process Regression for Damage Detection

(c) Residuals vs Stochastic Process Control Limits

Fig. 2. Truss bridge model and sensitivity analyses using the proposed method.

Table 1
Truss bridge model for the proposed strategy sensitivity analyses.

Properties	Parameters	Values
Dimension	Number of Span	3
	Deck Span Length	8 m
	Deck Width	6 m
	Pier Height	7 m
Degree of Freedom	Translational	3
	Rotational	0
Material	Young's Modulus	12000 MPa
	Section Area	0.1 m^2
Boundary Condition	Fixed	Nodes 1–2, 7–12
Loading	Points	Nodes 3–6
	Node Lump Mass	1000 kg
	Gravity	9.81 N/kg
Monitoring Points	A	3
	B	6
	Monitoring Direction	Vertical
GPR Hyperparameters	Kernel Type	RBF
	Amplifying Factor	1.0
	Length Scale	0.05
	Noise Level (standard deviation)	0.75

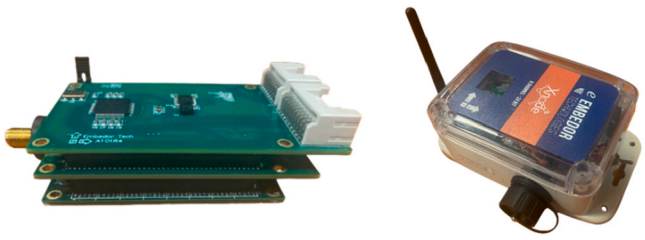


Fig. 3. Next-generation wireless smart sensor platform – Xnode.

truth and the predicted value: if the gap is above a specified threshold, it is considered abnormal; otherwise, normal. Theoretically, one can use the pointwise standard deviation to compute the threshold, i.e., check whether $|\boldsymbol{y}_p - \boldsymbol{y}_{gt}| \leq k\boldsymbol{\sigma}^{\boldsymbol{i}}_{\boldsymbol{p}|\boldsymbol{t}}$. Alternatively, in practice, one may also choose to use the standard deviation of the training output, i.e., check whether $|\boldsymbol{y}_p - \boldsymbol{y}_{gt}| \leq k\boldsymbol{\sigma}_t$.

2.3.4. Sensitivity analyses

To explore the relationship between damage severity and the difference between predictions and ground truth, a truss bridge model was developed to conduct finite element analyses using OpenSeesPy [27] and evaluate the proposed strategy, as shown in Fig. 2(a). Details of the model and simulation parameters are provided in Table 1. Fig. 2(b) and Fig. 2(c) illustrate examples of GPR and SPC for anomaly detection, respectively. In this typical setup, with $|\boldsymbol{y}_p - \boldsymbol{y}_{gt}| \leq 2\boldsymbol{\sigma}_t$ as the standard, a stiffness reduction of over 20 % in the vertical direction can be readily identified. Notably, in practical applications, adjusting the SPC threshold allows users to customize sensitivity based on specific requirements.

One key issue related to anomaly detection sensitivity is sensor placement. Proper placement enhances detection by positioning sensors near critical structural members to capture relevant signals, e.g., bridge pier in the study. Placement should also consider ease of installation and maintenance, as well as signal quality, particularly for wireless sensors, e.g., radio performance was examined in different bands were examined in the study. Ensuring access to a reliable power source, such as a wall outlet or solar energy, e.g., solar panel oriented to the east direction in the study, is also important for consistent operation.

Table 2
Xnode key features.

	Xnode Smart Sensor
Sensing channels	8
Sample rate	1–16 kHz
A/D resolution	24-bit
Time sync error	+ /−10us
Acquisition schemes	periodic/trigger sensing
LOS transmission range	> 1 km
Data rate	250–1000kbps
Transmission protocol	IEEE 802.15.4
Clock speed	12–204 MHz
Volatile data memory	32MB
Permanent storage	128MB NAND Flash + 4 GB SD card
Power draw (sensing)	~220 mA
Power draw (sleep)	~0.5 mA

3. Adaptive edge intelligence framework for wireless smart sensors

3.1. Next-generation wireless smart sensor platform

This study uses the Xnode, a high-performance wireless smart sensor (WSS) platform, as the testbed of the displacement estimation algorithm and the GPR-SPC-based anomaly detection framework. As shown in Fig. 3, Xnode is composed of three printed circuit boards: the processor board, the radio/power board, and the sensor board. All three boards along with other accessories are encompassed by an environmentally-hardened enclosure. More specifications are listed in Table 2.

3.2. The edge intelligence framework

Fig. 4 demonstrates the framework for the proposed anomaly detection strategy on a WSN. As depicted, a typical IoT-based monitoring system features sensor nodes, the gateway node, cloud server and end user interface. Either based on schedule or demand, the anomaly detection workflow starts from the gateway with the request for maximum displacement values from the sensor nodes. After receiving the request, the sensor nodes start to load acceleration data, then load or compute the filter coefficients, then conduct convolution to obtain displacement estimation. Afterwards, each sensor sends the maximum value of displacement back to the gateway node. On the Xnode, the communication between gateway node and sensor nodes is implemented with a 'Remote Procedure Call (RPC)' mechanism to be introduced in Section 3.4.1. Apart from displacement, the gateway node obtains local temperature requested from the internet through an API or a built-in thermometer. Next, the gateway node appends the maximum displacement values from node 1 and node 2, together with temperature to the record history file. Afterwards, the configurable parameters are updated on demand according to the model updating mechanism. The SD card serves to decouple the displacement estimation and GPR-SPC for anomaly detection. For GPR-SPC anomaly detection, the program loads data (training input/output, prediction input/ground truth output) from the record history file according to the associated configuration parameters. Then, GPR is conducted according to formulae (11−15) to obtain the prediction value and associated confidence interval. For SPC, anomaly detection is performed by comparing the difference between the prediction value and the ground truth. Note that, in practice, the control limit is usually replaced by two- or three-times standard deviation of training output. If an anomaly is detected, an alarm will be raised and relayed to the end user.

3.3. Independent computation for dynamic displacement estimation on sensor nodes

In the computing framework introduced in Section 3.2, apart from communication, the entire process of dynamic displacement estimation

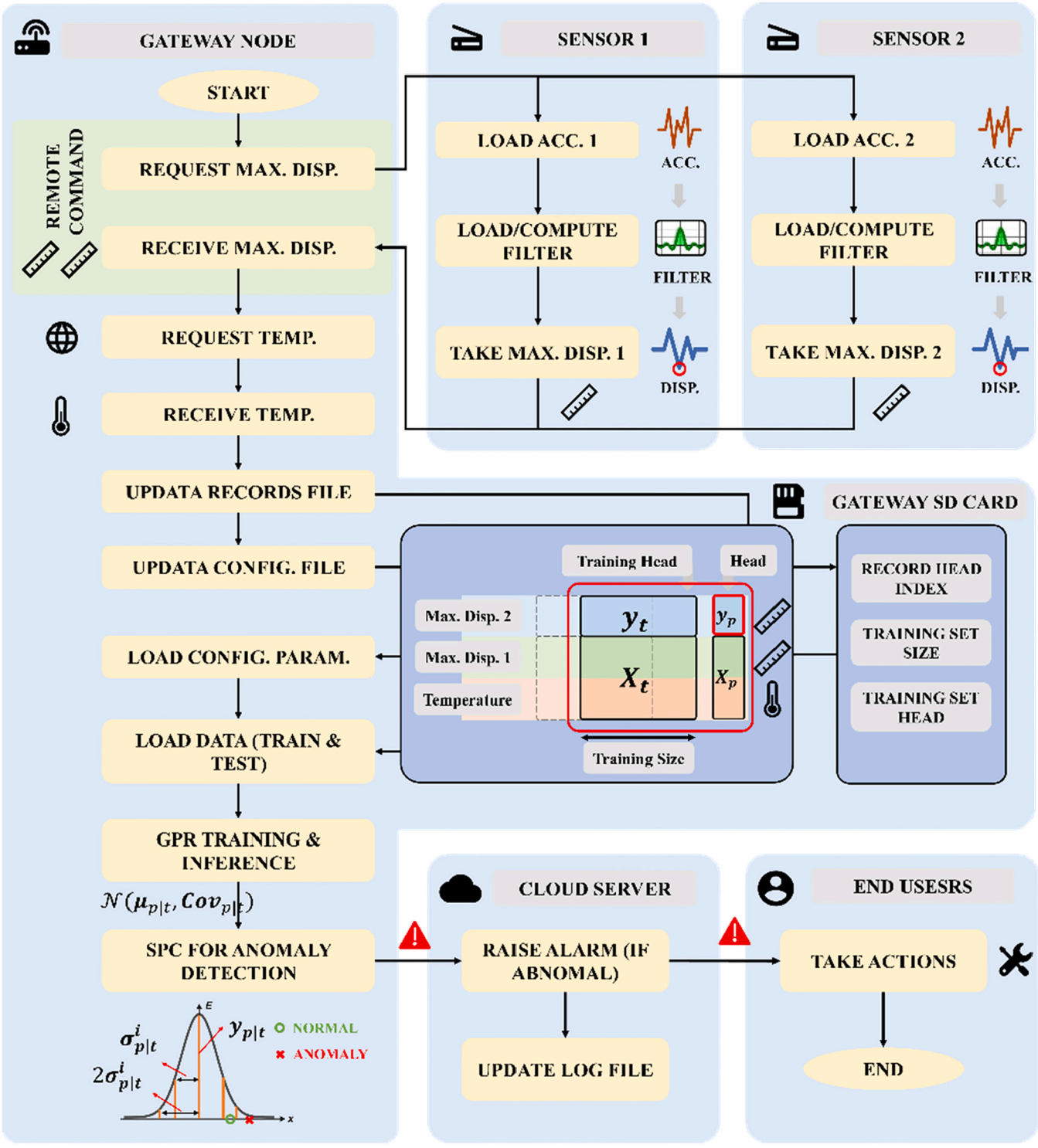


Fig. 4. Edge intelligence framework for anomaly detection.

is completed on sensor nodes alone, as shown in Fig. 4.

3.3.1. *Dynamic displacement estimation*

In this study, to estimate the dynamic displacement is in essence to compute the convolution of the filter and the acceleration data. To facilitate onboard realization, several measures are investigated based on a benchmark setup (10 minutes, 1 kHz acceleration data). In Table 3, each column is an approach, standing for different combinations of the measures in each row, and the selected measures are marked with an 'X'. By comparing the approaches, the effectiveness of each measure in Table 3 can be revealed, as shown by the time consumption in the last two rows in Table 4.

1. Use External FFT Library. As a baseline, for filter calculation, KissFFT [28] is investigated. Using this library, the total time consumption of approach 1 in Table 4 is 3353.35 s.
2. Decimate Data. Decimating data refers to reducing data by applying a low-pass filter and down-sampling with a factor of 10, bringing a

Table 3
Details of implementation approaches.

	Approach							
	1	2	3	4	5	6	7	8
Use External FFT library	X	X	X	X				
Decimate data		X		X	X	X	X	X
CMSIS DSP for filtering			X	X	X	X	X	X
CMSIS DSP for the decimation				X	X	X	X	X
Precompute and save coefficients					X	X	X	X
Save raw data	X	X	X	X	X			
Save binary data							X	X
Onboard processing with RTOS								X

5.29x speedup, as shown in Table 4. ('x' stands for 'times', same for the rest)

3. CMSIS DSP. CMSIS DSP (Common Microcontroller Software Interface Standard, Digital Signal Process Library) is a library designed for ARM Cortex series processors to take advantage of hardware-accelerated floating-point calculations and hardware-specific optimizations. Compared to previous approaches, the speedup factors for CMSIS DISP filtering and data decimating are 2.27x and 5.37x, respectively, proving its efficiency. For filtering, the theoretical output length is the sum of the filter and the input signal minus one.
4. Precomputation and Saving Coefficients. In the calculation, the filter coefficients can be saved and loaded when required instead of being calculated each time, bringing a 1.06x speedup.
5. Save Raw Data and Transitional Data. In fact, raw data and the transitional data can be discarded after use, which can save the time of data saving, leading to a 4.4x speedup.
6. Save Binary Data. Saving the data in binary form can be more efficient than human-readable form, which can lead to a 4.29x speedup.
7. Onboard Processing with RTOS. This measure means the processing is performed onboard, concurrent with sensing, utilizing the real-time capability of the FreeRTOS real-time operating system while not interfering with the sampling task. With concurrent processing, the entire process can be completed within 30 ms, leading to a significant incremental speedup factor of 87.00x.

The SHM community may be particularly interested in the overall performance of key tasks enabled by edge computing, as shown in Table 5. The benchmark results in Table 5 use 10 min of 1 kHz acceleration data, tested on a laptop and Xnode. For the laptop to simulate a central server, computations are executed on a Lenovo Legion R9000P equipped with an AMD R9 7945HX processor (boost frequency up to 4.7 GHz) and 32 GB of 5600 MHz memory. For edge computing, the computations are performed on Xnode, as detailed in Section 3.1. In the traditional approach, all raw data is transmitted to the server for processing, whereas the edge intelligence method processes data onboard and transmits only the final results. This approach notably reduces computing time, primarily due to FreeRTOS, which facilitates concurrent computation and sensing. Additionally, the power consumption values in Table 5 include communication overhead, such as 4 G module initialization and connection setup. As shown in Table 5, the edge intelligence approach requires less time for both computing and transmission compared to the traditional method. While local computing requires additional power, the reduced transmission overhead leads to a substantial decrease in overall consumption, highlighting the advantages of edge computing.

Table 5
Comparison of displacement estimation using traditional and edge intelligence methods.

Item	Sub-item	Traditional SHM	Edge Intelligence SHM
Time Consumption	Computing	0.63 s	0.03 s
	Transmission	3.88 s	0.04 s
Power Consumption	Computing	8.5 mAh (Server)	0.019 mAh (Edge)
	Transmission	0.74 mAh (Edge)	0.26 mAh (Edge)

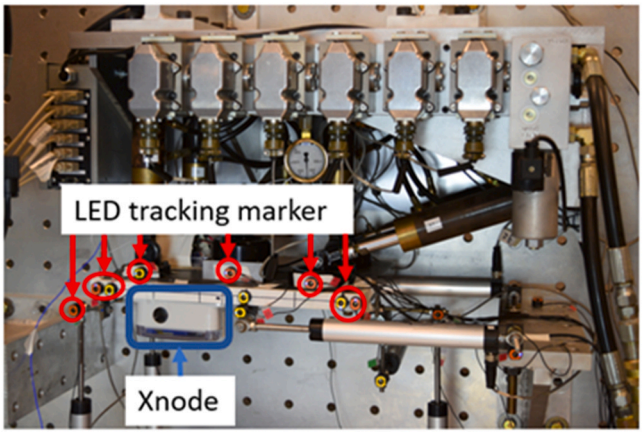


Fig. 5. Physical setup of the displacement tracking test.

Table 4
Breakdown of runtimes for the implemented approaches.

	Approach							
	1	2	3	4	5	6	7	8
Read data	38.05	38.05	38.05	38.05	38.04	0	0	0
Obtain coefficients	190.80	190.80	190.80	2.93	0.04	0.04	0	0
Filtering	3085.65	396.68	12.08	2.59	2.59	2.59	2.59	0
Save data	38.86	8.57	38.86	8.57	8.58	8.57	0.03	0.03
Total time	3353.35	634.10	279.78	52.14	49.24	11.20	2.61	0.03
Speedup factor (incremental)	N/A	5.29x	2.27x	5.37x	1.06x	4.40x	4.29x	$8.70e^{1}$ x
Speedup factor (cumulative)	N/A	5.29x	$1.20e^{1}$x	$6.43e^{1}$x	$6.81e^{1}$x	$2.99e^{2}$x	$1.28e^{3}$x	$1.12e^{6}$x

Notes:
1. All the runtimes are measured in seconds.
2. The read data in the first row indicates 10 minutes of data or 600,000 samples.
3. The third row, filtering includes decimation and displacement estimation.
4. The incremental speedup factor is to compare the approach in each column with the approach in the previous column, and the cumulative speedup factor is to compare with the first column.

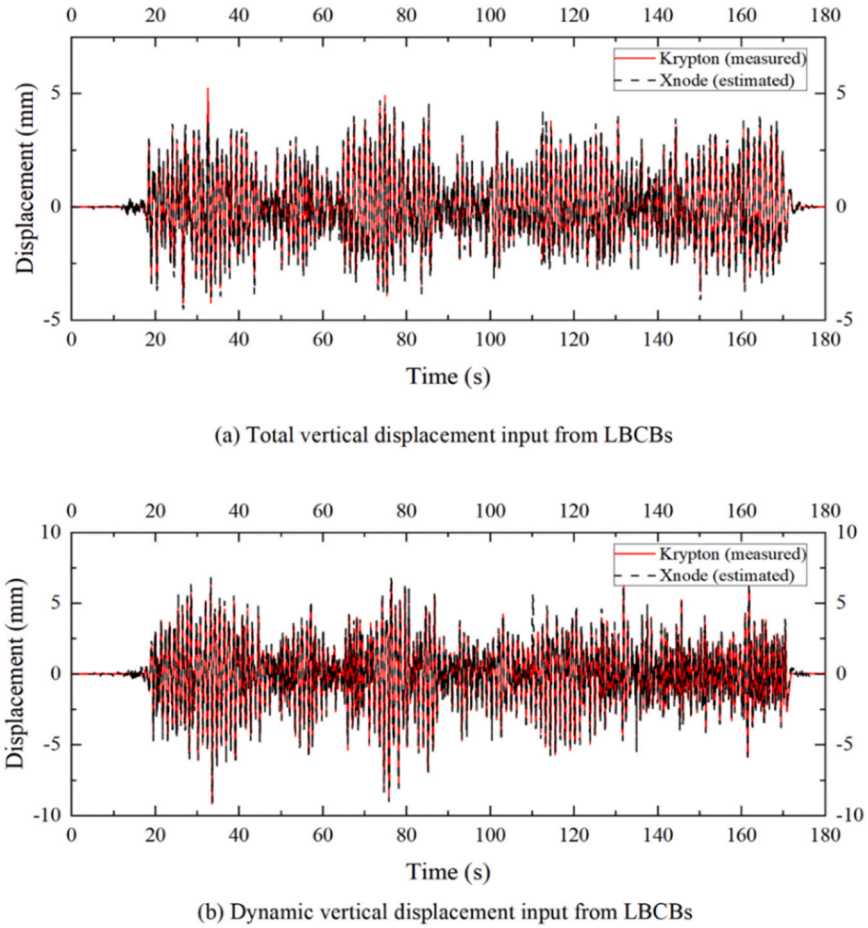


Fig. 6. Comparison between measured dynamic displacement from the Krypton camera and the Xnode: (a) total vertical displacement; (b) dynamic vertical displacement.

3.3.2. *Validation using lab test data*

To validate the onboard displacement estimation implementation, a lab test comparing the ground truth and the estimated displacement was conducted. As shown in Fig. 5, the setup reproduced a test record by using the Load and Boundary Condition Boxes (LBCB), and the dynamic displacement was estimated using an attached Xnode. For comparison, a Krypton K600 camera was utilized to track the displacement of the sensor node by tracking the LED markers, and the camera estimation was used as ground truth. In terms of sampling rate, the original sampling rates were 1000 Hz, 100 Hz, 1000 Hz for the actuators of LBCBs, the Krypton camera, and the Xnode, respectively. As shown in Figs. 6 and 7, the estimated data showed a good match with the results from the camera, for both total displacement and the dynamic displacement.

3.4. *Coordinated computation for anomaly detection handled by the gateway node*

The gateway node handles the coordination of multiple sensor nodes for anomaly detection. It retrieves the maximum dynamic displacement from each node and then retrieves temperature data from the internet or a built-in sensor. Afterwards, anomaly detection can be conducted.

3.4.1. *Remote procedure call mechanism for coordination within WSN*

The computation and result retrieval on sensor nodes rely on a proper communication mechanism. To facilitate development, the RPC mechanism [16] was adopted, offering an effective means for nodes to collaboratively perform a task. The RPC mechanism adopts a master-slave structure and consists of two-stage operations. The first stage is for command registration, stating the involved functions and data. The second stage is for command execution for targeted tasks, involving four user-customized functions denoted as 'sent', 'func', 'resp', 'exec', each standing for a critical point in the communication process. As shown in Fig. 6, function 'sent' stands for the actions to be taken on the master node when the command has been sent to the slave nodes; function 'func' encapsulates the actions to be executed remotely on slave nodes when receiving the commands; function 'resp' contains the actions to be conducted on slave nodes when the response is sent back to the master node; function 'exec' describes the actions to be taken on master nodes when receiving the response from slave nodes. Given the fact that the displacement estimation may not be finished within a single round of the remote command call mechanism, the whole process was separated into two rounds RPC. The first round is designed to conduct displacement estimation while the second is designed to retrieve the results, denoted as CALCULATION and RETRIEVAL, respectively. Note that, between the two rounds, there is a user-defined delay to wait for the sensor nodes to finish the computing tasks.

For a WSN, the gateway acts as the master node, and the sensor nodes act as the slave nodes. As described above, to use RPC, the commands are required to be registered first. Afterwards, the two rounds of remote procedure call are executed. In fact, apart from the functions 'func' and 'exec', the others mainly serve as utility functions for checking and reporting progress. For the two rounds of remote procedure call, functions closely related to the computation are only 'calculation_func', 'retrieval_func', 'retrieval_exec'. In 'calculation_func', the sensor node is commanded to load the prescribed data records, and then the data is reshaped into the appropriate structure. Afterwards, the filter is

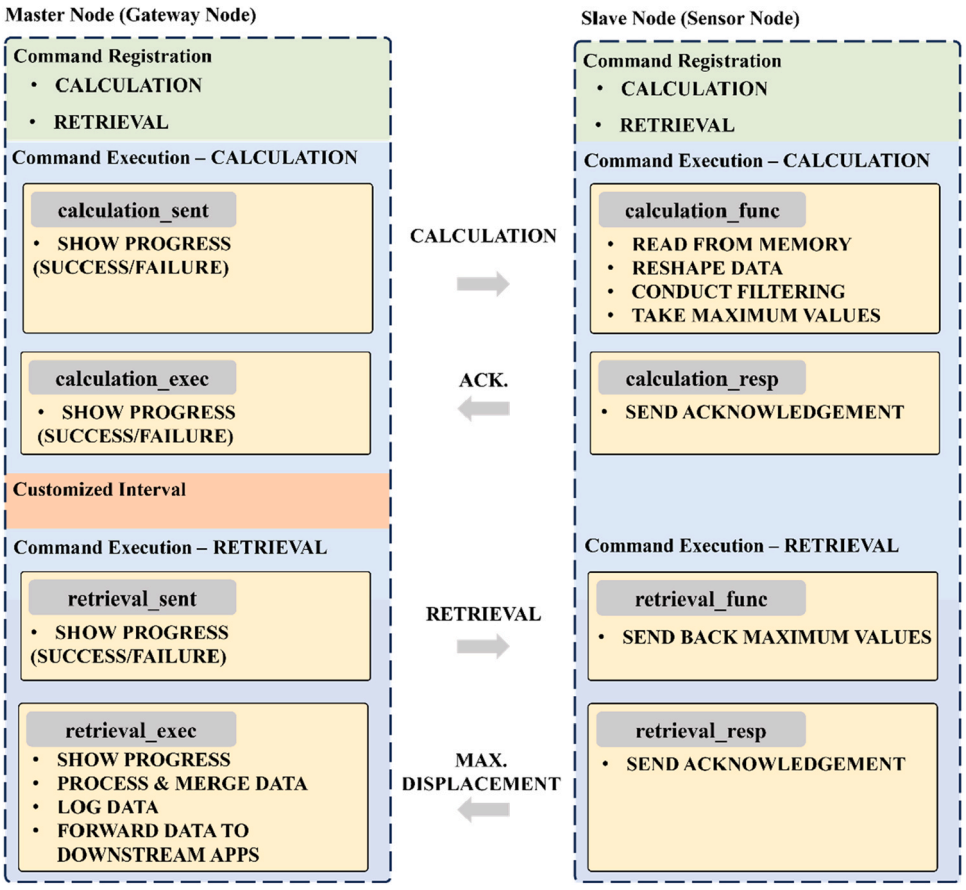


Fig. 7. Schematic illustration of remote procedure call for displacement estimation.

calculated or loaded to be applied to the acceleration data to obtain the displacement estimation, and then the maximum displacement is calculated. In 'retrieval_func', if the calculation is successfully conducted in round 1, then the maximum displacement values can be set up for data retrieval in this step. Function 'retrieval_exec' defines the behaviors of the gateway node after the maximum displacement values are received, i.e., dataset updating and GPR model updating.

3.4.2. GPR and SPC for anomaly detection

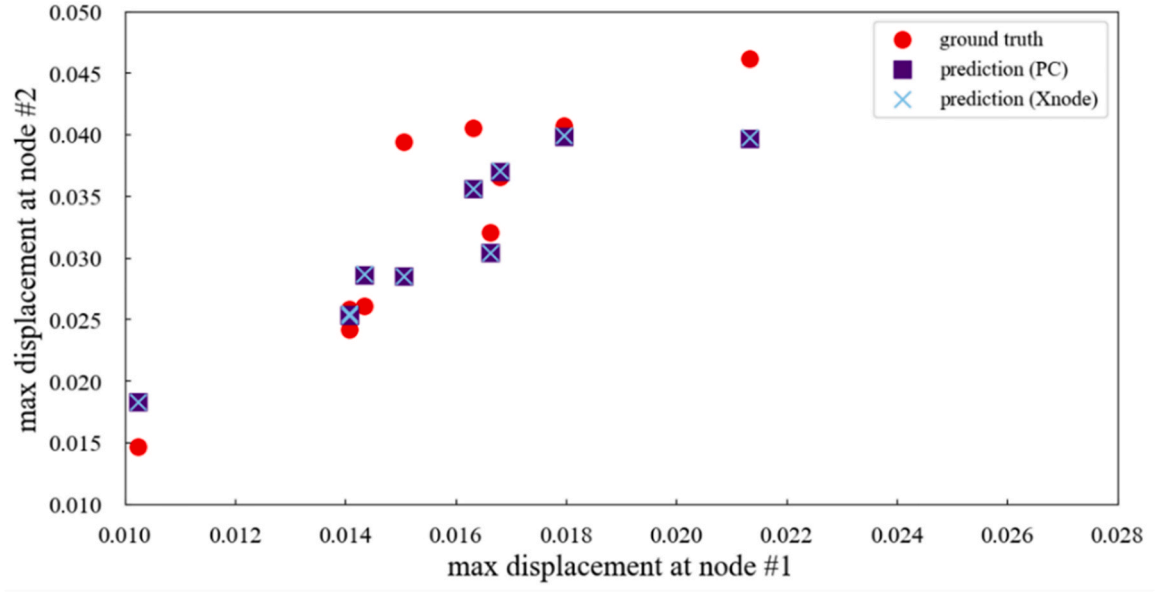
As shown in Fig. 4, for GPR and SPC, the input data are temperature and maximum displacement from one sensor node, while the output data is the maximum displacement from the other. Moreover, to facilitate computation, two files were used on the SD card: one is to store history record data and serves as a database, including temperature and maximum displacement from two sensors, and the other is to store configurable parameters, including the index of the training dataset head, the size of training dataset, and the current head of the record history. To facilitate related operations, a lightweight file system, FATFS [29], was used.

For GPR and SPC, one can conduct model training and inference at the same time, as implied by formulae (11−15). As shown in Fig. 4, the first step is to read configurable parameters indicating where and how many to load and then load the data for training and inference accordingly from the history record file, including training input and output, prediction input and ground truth output. The next step is the calculation for prediction value and confidence interval. As introduced in Section 3.3.1, the CMSIS DSP library is used to harness the hardware-specific optimizations to accelerate floating-point calculations for computationally intensive tasks. To use CMSIS DSP functions, one should dynamically allocate memory for source data and destination data, where source data provides input and destination data is to store output data. As implied by formulae (11−15), there are many transitional results that can be stored and loaded when required rather than being computed every time. Another implication is that the numerical errors propagate in the matrix computation, therefore, when designing the program, it is important to be aware of the accumulation of numerical errors. The last step is the anomaly detection using SPC, of which the key is to compare the difference between the prediction value and the ground truth value, and to determine whether the gap is greater than a prescribed threshold, e.g., 2σ. In practice, users can alternatively use the standard deviation of the whole training output to derive the threshold.

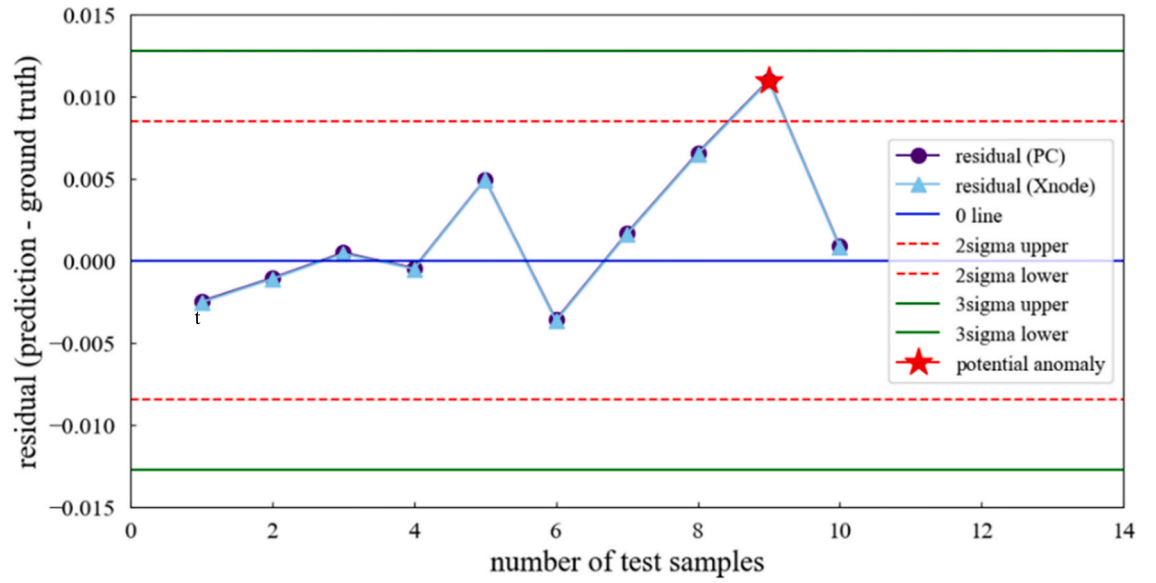
For each record of an event of interest, the program tracks and updates the location of the head of the history file, simultaneously, the program also tracks the head of the training set. If the difference between the file head and the training set head reaches the user-customized value, for example, 50, the index of the training set head moves forward for 50 entries. This user-customized value determines how frequently to update the training set. In this way, the model can adapt to the latest status.

3.4.3. Validation using field test data

The validation of the proposed edge computing framework for anomaly detection was conducted by comparing the results from a PC and Xnode. For anomaly detection, the key idea for validation is to compare the consistency of the GPR-SPC model output values from the PC and Xnode. In this section, the two major tasks performed on the gateway, i.e., GPR for output prediction and SPC for anomaly detection were tested. Note that for anomaly detection, the lower and upper control limits for SPC were calculated based on the averaged standard deviation over the training output data rather than using the individual confidence interval associated with each prediction output value. The



(a) GPR prediction – computer vs Xnode



(b) SPC anomaly detection - computer vs Xnode

Fig. 8. Comparison of PC and Xnode implementations.

reasons were two-fold: using training output average is easier and has been proved reliable in practice; the computation for individual confidence interval is trickier and introduces numerical errors, resulting in higher inconsistency.

The PC was installed with a 64-bit Windows operating system and the computation was conducted using Python script; the 32-bit Xnode was running a FreeRTOS system, and the computation was conducted using the CMSIS-DSP library. The data for GPR-SPC-based anomaly detection validation was from a real railroad bridge monitoring project to be introduced in Section 4. The data includes maximum displacement values from two sensors together with temperature data. Maximum displacement from one sensor and the temperature are used as input for prediction, the actual value of the max displacement from the other sensor is used as the ground truth for the GPR-SPC model prediction value. To be short, this validation is to compare the result consistency between the PC and Xnode, if they are close enough, the edge computing will be considered as reliable. In the validation, 30 samples were used for GPR training, and 10 samples were used for the prediction test.

For the first task, i.e., computing the prediction value for the output, as shown in Fig. 8(a), the results from PC and Xnode are highly consistent with each other, showing considerable potential for edge deployment in engineering practice. For the second task, checking whether the residual values fell inside the lower and upper control limits, as shown in Fig. 8(b), the residuals from PC and Xnode also showed great agreement. Note that the residual is calculated by subtracting the prediction output value from the ground truth output value. The lower and upper control limits adopt 2σ rule or 3σ rule, where σ is the averaged standard deviation of the training output. The 2σ rule stands for narrower tolerance band and more rigorous anomaly detection standard while the 3σ rule stands for wider tolerance band and less

Table 6
General information of the steel and timber bridges.

No.	Type	Height	Length	Number of Sensor
I	Steel	22	134	6
II	Steel	25	266	8
III	Mixed	43	209	3
IV	Mixed	33	226.5	2
V	Mixed	33	165.5	3
VI	Mixed	49	298.58	2
VII	Mixed	18	92.83	2
VIII	Mixed	36	211	2
IX	Timber	33	139.75	3
X	Timber	16	91.5	2
XI	Mixed	29	165.67	2

rigorous anomaly detection standard. The standard deviation of the training output set can be easily calculated on edge devices, which introduces almost zero numerical errors as the operation is quite simple.

Note that, compared to using average standard deviation for control limits, using individual confidence interval associated with the prediction value for SPC can lead to larger numerical errors, requiring extra efforts to suppress the errors. Key sources of numerical errors can be identified as follows. The first is the computation of matrix inverse where the computation is intense, especially when the matrix is ill-conditioned. To calculate the standard deviation values, the diagonal elements of the covariance matrix should be taken out and then applied square root operation. When these elements are close to zero, the square root operation can significantly amplify the errors. Therefore, in practice, the simple and effective average standard deviation is recommended for control limits computation.

4. Full-scale applications

In this section, the proposed strategy and edge intelligence framework were validated using full-scale short-span railroad bridges, offering a reliable reference for typical civil structures. However, for large-scale structures, i.e., long-span bridges, further investigation may be needed to address potential challenges.

4.1. Railroad bridges and WSS setup

In this application, the WSS monitoring system enabled with onboard displacement estimation was installed on steel truss and timber trestle railroad bridges. In this study, two steel and nine timber bridges were selected for assessment as shown in Table 6.

For steel bridges, six to eight sensors were installed at the midspan of the bridge. For each timber trestle bridge, two to three sensors were mounted on the tall piers for measuring noticeable vibration. The sensors were installed at the pier caps to capture vibration while not affecting the structural integrity. Each sensor node directly communicates with the gateway node, forming a star topology.

Table 7
Displacement comparison of the East and West sides of the two steel truss bridges.

	Bridge I		Bridge II	
	Vertical	Lateral	Vertical	Lateral
Mean difference (%)	318.11	4.34	12.16	2.42
Max difference (%)	416.61	9.6	30.82	3.14

For onboard computing, the hyperparameters were set up based on trials or grid search to optimization estimation or regression performance, and the values can be different from case to case. Given that bridge II and bridge IV are selected as examples in Sections 4.2 and 4.3 respectively, the associated hyperparameters are also provided here. For displacement estimation for bridge II, the regularization factor β was set to 0.317, and the filter order n was set to 4; for GPR for bridge IV, noise level ε was set to 0.0004, the amplifying factor a was set to 1.0, length scale vector $[l^1, l^2]$ was set to [5, 0.2].

4.2. Bridge condition assessment using pure displacement estimation

Two steel bridges (I and II) in Indiana, US, were monitored for six weeks. Due to low traffic frequency (about 1 train/day), the amount of data is too limited to apply the GPR-based anomaly detection, but the results can be used without GPR. Fig. 9(a) gives an example of the bridge and sensors instrumented and Fig. 9(b) gives an example of the data. Comparing measurements on the two sides of the bridge can reveal abnormal conditions. Table 7 shows the comparison between the West and East sides under train loading. It can be observed that for lateral displacement, a good agreement between the displacement of the trusses can be observed for both bridges (Fig. 9(b)). The same behavior is observed in the vertical direction at bridge I. This result, however, is not observed in bridge II in the vertical direction, where the maximum displacement on the West side is three times higher than the East. In a bridge inspection, it was found that on the West side, there were multiple members with section losses, while the East side showed no section loss, explaining the data anomaly. This test marks a successful application of the displacement estimation algorithm.

4.3. Bridge condition assessment using gaussian process regression

As the application of the proposed anomaly detection approach, both

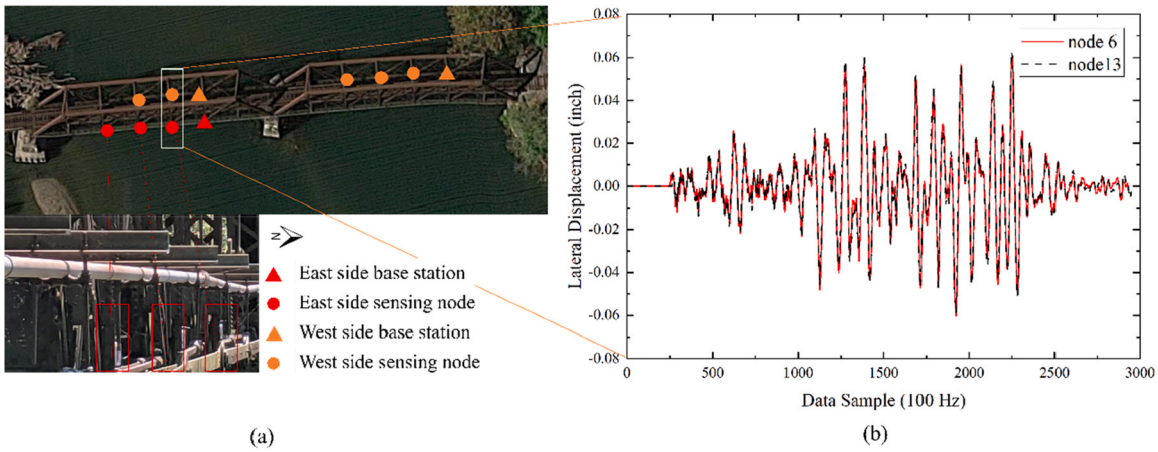


Fig. 9. Illustration of steel truss bridges monitoring: (a) instrumented sensors; (b) example data captured on sensing nodes.

Fig. 10. Examples of the instrumented timber trestle bridges.

short- and long-term timber trestle bridge monitoring was conducted, examples are given by Fig. 10. In the short-term monitoring case, the GPR model is established by using the data from the first 3–4 days. Four bridges (VII, VIII, X, XI) were selected for short-term change monitoring. In the long-term case, five bridges (III, IV, V, VI, IX) were selected for monitoring in two separate campaigns.

Figs. 10 and 11 together illustrate the step-by-step application of the proposed GPR-SPC anomaly detection framework, using Bridge IV as an example. The first step is to collect data and build the training dataset, represented by the black dots in Fig. 11(a). Next, curve fitting is performed using this training data, shown by the black curve and the associated purple confidence interval band in Fig. 11(a) and Fig. 11(b). The black curve provides continuous predictions across the range, and the difference between the ground truth and the prediction at each horizontal position represents the residual, as shown in Fig. 12(a). The third step is to utilize the built-up model, namely the fitted curve and its associated confidence interval band, to determine whether the new data is normal or not by comparing the residual with the control limits. In this case, the input data is the horizontal location, as shown in Fig. 11(b),

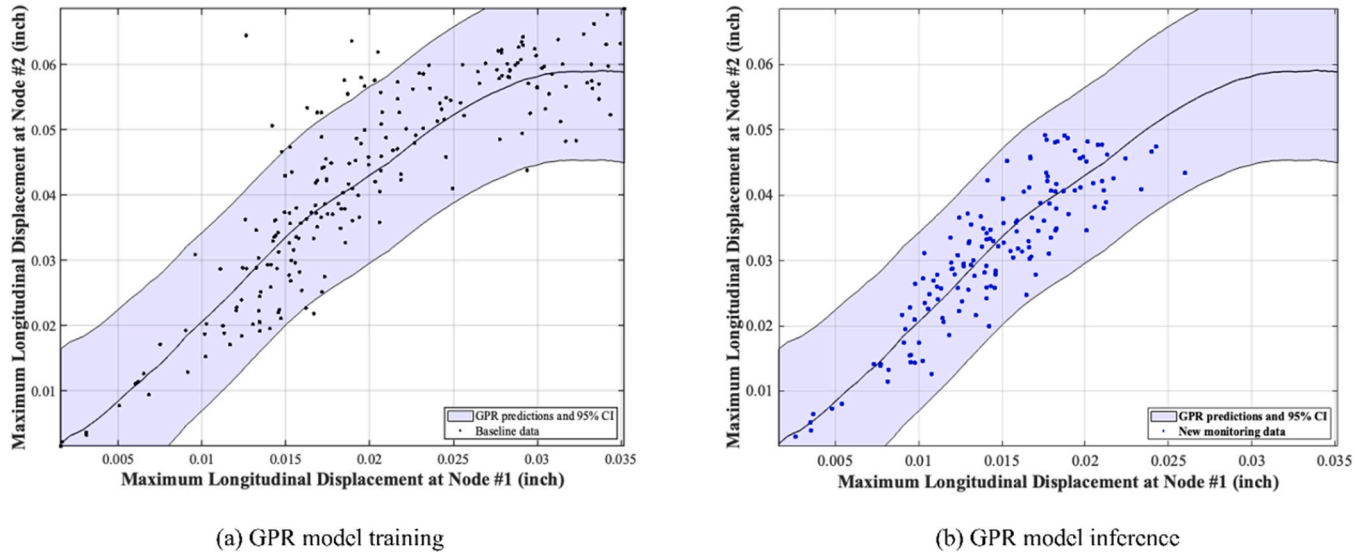


Fig. 11. Gaussian process regression anomaly detection: (a) model training; (b) inference.

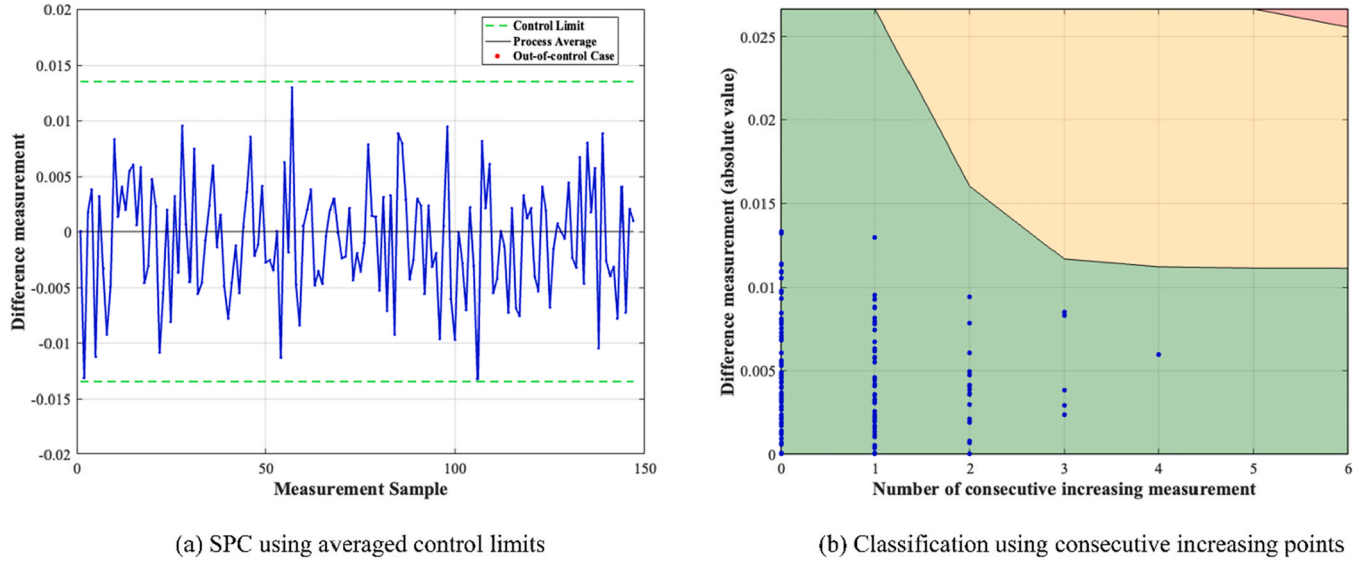


Fig. 12. Stochastic process control and condition assessment classification.

Table 8
Classification of timber bridge performance based on short-term monitoring.

No.	Node	Lateral Direction		Longitudinal Direction	
		Max Yellow Indicator (%)	Max Red Indicator (%)	Max Yellow Indicator (%)	Max Red Indicator (%)
VII	1	0.98	0	1.48	0
VII	2	0	0	1.72	0
VIII	1	1.35	0	3	0
VIII	2	0	0	0	0
X	1	0.38	0	2.55	0
X	2	0.77	0	0	0
XI	1	1.25	0	0	0
XI	2	2.33	0	0	0

Table 9
Classification of timber trestle bridge performance based on long-term monitoring.

No.	Node	Lateral Direction		Longitudinal Direction	
		Max Yellow Indicator (%)	Max Red Indicator (%)	Max Yellow Indicator (%)	Max Red Indicator (%)
III	1	1.27	0	0	0
III	2	0.83	0	1.21	0
III	3	0.83	0	1.83	0
IV	1	0	0	0	0
IV	2	0	0	0	0
V	1	2.14	0	0.94	0
V	2	0.28	0	0.5	0
V	3	0.85	0	0	0
VI	1	0	0	1.59	0
VI	2	0.61	0	0	0
IX	1	3.79	0	7.38	0
IX	2	3.92	0.61	7.89	0.88
IX	3	0.73	0	6.61	0.59

and the output is the vertical location of the new incoming data. If the new incoming data falls in the confidence interval band, it is considered as normal. As can be seen, for bridge IV, all new incoming data fall in the confidence interval band and are considered normal. Note that, in practice, engineers prefer to use the averaged standard deviation of training output to derive the control limits, including both the upper and lower boundaries, as shown in Fig. 12(a). Finally, values from the SPC regarding difference measurement and the number counting consecutive increasing points in difference measurement are used to track the anomaly detection, as shown in Fig. 12(b). In Fig. 12(b), as well as in Tables 8 and 9, color coding indicates status: green represents data within the 2σ range (normal); yellow indicates values between the 2σ and 3σ limits (attention required); and red highlights values beyond the 3σ limit, signaling action is required [23]. Note that the temperature data is also collected, however, for better visualization, the temperature data is not shown.

For short-term change detection monitoring, Table 8 summarizes the findings of the four timber bridges. All of the analysis results show that only 0 – 2.55 % of the data points are in the yellow regions, indicating they experienced no change during the short-term monitoring campaign.

For long-term change detection monitoring, since there was no information about the ground truth of the health of the structure, all possible regressions have been done to make full use of data, and the worst case (more data points lying in the yellow/red regions) was used for the classification. Table 9 summarizes the findings of the long-term change monitoring of the five timber bridges. From the data in Table 9, bridge IX shows a significantly higher number of measurements in the red region. Bridge IX was neither known for having any previous performance issues nor scheduled for maintenance. Nevertheless, around the bridge, researchers found multiple steel rail plates and spikes that had not been observed in the first deployment, indicating possible maintenance work being conducted.

The proposed anomaly detection strategy is theoretically applicable to a wide range of civil and mechanical structures and can be extended to incorporate multiple sensors and diverse data types beyond temperature and displacement, enabling more robust and reliable outputs. The current study specifically targets and has been validated on short-span bridges, demonstrating strong potential for effective application to typical civil structures. For larger-scale structures, such as long-span bridges, further research may be needed to address potential challenges and broaden the strategy's applicability in structural health monitoring.

5. Conclusions

This paper proposes an adaptive onboard anomaly detection strategy combining a reference-free displacement estimation algorithm, GPR, and SPC. The strategy is designed for and implemented using the edge computing paradigm, enabling the SHM system to harness advantages that centralized computation lacks, such as reduced power consumption and low latency, which is the principal novelty of this work. To facilitate the onboard realization of the proposed strategy, which includes displacement estimation and anomaly detection using GPR and SPC, strategies were explored and implemented to address the limited onboard computational resources. For instance, the CMSIS DSP library and the real-time operating system FreeRTOS were employed, resulting in significant efficiency gains, such as up to six orders of magnitude improvement in displacement estimation. The onboard realization was validated through lab tests and offboard computations, both demonstrating excellent agreement. A model updating mechanism was also incorporated to enhance adaptivity. Full-scale applications to short-term and long-term railroad bridge monitoring provided robust practical support for applying the proposed anomaly detection algorithm and framework to normal-sized civil engineering structures. However, for more challenging scenarios, such as large-scale structures or long-span bridges, further research is needed to extend the applicability of the strategy. These challenges also present valuable directions for future exploration.

CRediT authorship contribution statement

Yuguang Fu: Writing – review & editing, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Kirill Mechitov:** Writing – review & editing, Software, Methodology, Conceptualization. **Billie F. Spencer:** Writing – review & editing, Supervision, Resources, Investigation, Funding acquisition. **Tu Hoang:** Writing – review & editing, Visualization, Validation, Software, Resources, Methodology, Formal analysis, Data curation, Conceptualization. **Shuaiwen Cui:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The authors want to gratefully acknowledge the Federal Railroad Administration (FRA) for the financial support of this research under contract DTFR53-17-C-00007, and ZJU-UIUC Institute Research under Grant #ZJU083650, NTU Start-up Grant 021323-00001, MOE AcRF Tier 1 Grants, No. RG121/21.

Data availability

Data will be made available on request.

References

[1] Lynch JP, Sohn H, Wang ML. Sensor technologies for civil infrastructures, volume 1: Sensing hardware and data collection methods for performance assessment. Elsevier Science; 2014.

[2] Raza Hung VDang TB-TMohsin, Nguyen Tung V, Nguyen HX. Deep learning-based detection of structural damage using time-series data. Struct Infrastruct Eng 2021; 17:1474–93. https://doi.org/10.1080/15732479.2020.1815225.

[3] Abdeljaber O, Avci O, Kiranyaz MS, Boashash B, Sodano H, Inman DJ. 1-D CNNs for structural damage detection: verification on a structural health monitoring benchmark data. Neurocomputing 2018;275:1308–17. https://doi.org/10.1016/j.neucom.2017.09.069.

[4] Bock C., Aubet F.-X., Gasthaus J., Kan A., Chen M., Callot L. Online Time Series Anomaly Detection with State Space Gaussian Processes. CoRR 2022;abs/2201.06763.

[5] Fu Y, Billie F, Spencer J. Sudden-event monitoring of civil infrastructure using wireless smart. sensors. 2022;53.

[6] Peng C., Fu Y., Spencer B.F. Sensor fault detection, identification, and recovery techniques for wireless sensor networks: a full-scale study. Proceedings of the 13th international workshop on advanced smart materials and smart structures technology, 2017, p. 22–23.

[7] Gomez F, Fu Y, Hoang T, Mechitov K, Spencer BF. Estimation of dynamic interstory drift in buildings using wireless smart sensors. J Struct Eng 2024;150:04024044. https://doi.org/10.1061/JSENDH.STENG-12482.

[8] Gomez F, Park J-W, Spencer JrBF. Reference-free structural dynamic displacement estimation method. Struct Control Health Monit 2018;25:e2209. https://doi.org/10.1002/stc.2209.

[9] Mondal TG, Chou J-Y, Fu Y, Mao J. A hybrid deep neural network compression approach enabling edge intelligence for data anomaly detection in smart structural health monitoring systems. Smart Structures and Systems 2023;32:179–93.

[10] Wang X, Wu W, Du Y, Cao J, Chen Q, Xia Y. Wireless IoT monitoring system in Hong Kong–Zhuhai–Macao bridge and edge computing for anomaly detection. IEEE Internet Things J 2024;11:4763–74. https://doi.org/10.1109/JIOT.2023.3300073.

[11] Amer A, Kopsaftopoulos F. Gaussian process regression for active sensing probabilistic structural health monitoring: experimental assessment across multiple damage and loading scenarios. Struct Health Monit 2023;22:1105–39. https://doi.org/10.1177/14759217221098715.

[12] Li M. SHM-based condition assessment of bridges using gaussian process regression 2017.

[13] Teimouri H, Milani AS, Loeppky J, Seethaler R. A Gaussian process–based approach to cope with uncertainty in structural health monitoring. Struct Health Monit 2017;16:174–84. https://doi.org/10.1177/1475921716669722.

[14] Rasmussen CE, Williams CKI. Gaussian processes for machine learning. MIT Press; 2005.

[15] Hoang T., Billie F. Spencer J. Autonomous Wireless Smart Sensor for Monitoring of Railroad Bridges. Report #54 2022.

[16] Sim S.-H., Spencer B.F. Decentralized Strategies for Monitoring Structures using Wireless Smart Sensor Networks. Newmark Structural Engineering Laboratory Report Series 019 2009.

[17] Fu Y, Zhu Y, Hoang T, Mechitov K, Spencer BF. xImpact: intelligent wireless system for cost-effective rapid condition assessment of bridges under impacts. Sensors 2022;22:5701. https://doi.org/10.3390/s22155701.

[18] Fu Y, Hoang T, Mechitov K, Kim JR, Zhang D, Spencer Jr BF. xShake: Intelligent wireless system for cost-effective real-time seismic monitoring of civil infrastructure. Smart Struct Syst 2021;28:483–97.

[19] Lea P. IoT and Edge Computing for Architects. Packt Publishing, Limited; 2020.

[20] Alajlan NN, Ibrahim DM. TinyML: enabling of inference deep learning models on ultra-low-power IoT edge devices for AI applications. Micromachines 2022;13. https://doi.org/10.3390/mi13060851.

[21] Zhou Z, Chen X, Li E, Zeng L, Luo K, Zhang J. Edge intelligence: paving the last mile of artificial intelligence with edge computing. Proc IEEE 2019;107:1738–62. https://doi.org/10.1109/JPROC.2019.2918951.

[22] Khan JA, Qureshi HK, Iqbal A. Energy management in wireless sensor networks: a survey. Comput Electr Eng 2015;41:159–76. https://doi.org/10.1016/j.compeleceng.2014.06.009.

[23] Hong YH, Kim H-K, Lee HS. Reconstruction of dynamic displacement and velocity from measured accelerations using the variational statement of an inverse problem. J Sound Vib 2010;329:4980–5003. https://doi.org/10.1016/j.jsv.2010.05.016.

[24] Zhang B, Ni Y. A data-driven sensor placement strategy for reconstruction of mode shapes by using recurrent Gaussian process regression. Eng Struct 2023;284:115998. https://doi.org/10.1016/j.engstruct.2023.115998.

[25] Fu W, Sun B, Wan H, Luo Y, Zhao W. A Gaussian processes-based approach for damage detection of concrete structure using temperature-induced strain. Eng Struct 2022;268:114740. https://doi.org/10.1016/j.engstruct.2022.114740.

[26] Yu Z, Xie W, Yu B, Cheng H. Probabilistic prediction of joint shear strength using Gaussian process regression with anisotropic compound kernel. Eng Struct 2023; 277:115413. https://doi.org/10.1016/j.engstruct.2022.115413.

[27] Zhu M, McKenna F, Scott MH. OpenSeesPy: python library for the opensees finite element framework. SoftwareX 2018;7:6–11. https://doi.org/10.1016/j.softx.2017.10.009.

[28] Borgerding M. KISSFFT: a Fast Fourier Transform (FFT) library that tries to Keep it Simple, Stupid n.d. ⟨https://github.com/mborgerding/kissfft⟩ (accessed May 25, 2024).

[29] Nishiura T. FatFs - Generic FAT Filesystem Module n.d. http://elm-chan.org/fsw/ff/ (accessed May 25, 2024).