13th International Conference on
Structural Health Monitoring of Intelligent Infrastructure
DOI: 10.3217/978-3-99161-057-1-094

Smart adaptive triggering strategy for edge intelligence enabled energy-efficient sensing

Shuaiwen Cui[1, 0000-0003-4447-6687], Xiao Yu[1, 0009-0007-8294-5671], Yuguang Fu[1*, 0000-0001-7125-0961]
[1]School of Civil and Environmental Engineering, Nanyang Technological University, 50 Nanyang Ave, Singapore, 639798
email: SHUAIWEN001@e.ntu.edu.sg, XIAO004@e.ntu.edu.sg, yuguang.fu@ntu.edu.sg

ABSTRACT: Achieving both energy efficiency and high triggering accuracy is a critical multi-objective optimization challenge in Structural Health Monitoring (SHM), particularly for power-constrained wireless edge devices deployed in dynamic environments. Traditional empirical and static-threshold-based methods often struggle to simultaneously have low miss trigger and false trigger rate and lack adaptability to varying environmental and operational conditions. This study proposes a multi-stage adaptive triggering strategy built upon a Feedback Control (FC) framework, driven by Bayesian Optimization (BO) as the optimization engine, and accelerated by Digital Twin (DT) for data augmentation and Neural Networks (NN) for real-time contextual understanding and robust inference. The strategy dynamically refines triggering thresholds based on simulated insights and partial observations, enabling rapid adaptation and generalization across scenarios. Experimental validation through simulations and onboard deployments demonstrates that the proposed method improves F-beta performance by over 30% compared to conventional empirical methods. This approach provides a promising pathway toward intelligent, energy-efficient, and sustainable SHM sensing through fast feedback, reduced deployment cost, and minimized missed critical events.

KEY WORDS: Trigger Sensing, Energy-efficiency, Adaptive Sensing, Feedback Control, Bayesian Optimization, Structural Health Monitoring

1 INTRODUCTION

In Structural Health Monitoring (SHM) systems, triggering mechanisms are essential for determining when edge sensor nodes should initiate data acquisition to capture transient structural responses during events of interest [1]. In real-world deployments with limited energy resources, triggering directly affects both responsiveness and long-term system sustainability. A mechanism that is both energy-efficient and accurate allows the system to capture meaningful data while minimizing power consumption [2]. When continuous high-rate sampling is infeasible, intelligent triggering becomes critical for reliable and autonomous monitoring.

Conventional methods typically rely on fixed thresholds or handcrafted rules based on prior knowledge [3]. While easy to implement and low in computational cost, they lack adaptability. Triggering parameters (e.g., threshold and duration) set during deployment often remain static, making it difficult to respond to changes in structural behavior or environmental conditions. In practical deployments, these parameters are often conservatively configured to minimize the risk of missing events of interest. This conservative strategy prioritizes recall at the expense of precision, which, although effective in reducing missed detections, often leads to an increased rate of false triggers [1].

Practical SHM deployments present several challenges that complicate the design of effective triggering mechanisms. Structural events are typically rare, limiting the amount of available data and making it difficult to establish reliable patterns for triggering. The occurrence of events is also highly imbalanced, with most data corresponding to normal or inactive states, which biases learning-based methods and complicates threshold design. In addition, due to energy constraints, sensors operate in low-power sentinel modes and only acquire data when a triggering condition is met. As a result, only responses associated with triggered events are recorded, leading to partial observability, where only a subset of structural behaviors is accessible for analysis [4]. These factors collectively call for a triggering mechanism that is adaptive, context-aware, and lightweight, while also possessing the capability to predict and infer unobserved structural responses.

To address these challenges, this study proposes a multi-stage adaptive triggering strategy primarily built upon a Feedback Control (FC) framework [5], which integrates a digital twin (DT) [6], onboard neural networks (NN) [7] [8], and Bayesian optimization (BO) [9]. FC enables the continuous refinement of triggering behavior through performance-driven feedback; BO operates as an optimization engine that seeks the global optimum and guides trigger parameters toward rapid convergence; DT facilitates data augmentation by simulating real deployment environments; NN provides real-time contextual awareness and accurate prediction, even under partial observability. This integration empowers the triggering system to adaptively and efficiently search for the optimal trigger parameters under uncertain event conditions.

The proposed strategy is validated upon a setup mimicking real-world deployment, showing clear advantages over empirical baselines. Specifically, it achieves over 30% improvement in F_β score, enhancing event detection without increasing unnecessary energy use. These findings demonstrate the effectiveness of the proposed strategy in simultaneously optimizing detection accuracy and energy efficiency, and further underscore the potential of integrating closed-loop control, Bayesian optimization, and digital twin to enable

DOI: 10.3217/978-3-99161-057-1-094

adaptive and intelligent triggering in real-world SHM deployments.

2 SMART ADAPTIVE TRIGGERING

2.1 Basic triggering mechanism and baseline strategy

An example of a typical trigger sensing mechanism is the one implemented in Xnode [2], which combines a low-power, low-resolution sensor with a high-power, high-resolution sensor, as illustrated in Figure 1 and Table 1. The low-power sensor, such as the ADXL362 [10], is responsible for continuously monitoring motion and detecting events of interest based on predefined thresholds and durations. When the signal amplitude exceeds the configured threshold for a specified duration, an event of interest is deemed to have occurred, as shown in Figure 1. At this point, the low-power sensor activates the high-power sensor, transitioning the system from sentinel mode to working mode to capture detailed event data. This fundamental triggering mechanism forms the basis of the smart adaptive triggering strategy proposed in this study.

Table 1. Xnode sensor energy consumption.

Mode \ Sensor	ADXL362 (12 bit)	LIS344 (24 bit)
Sentinel Mode	Working 270nA	Sleeping 1uA
Working Mode	Sleeping 10nA	Working 680uA

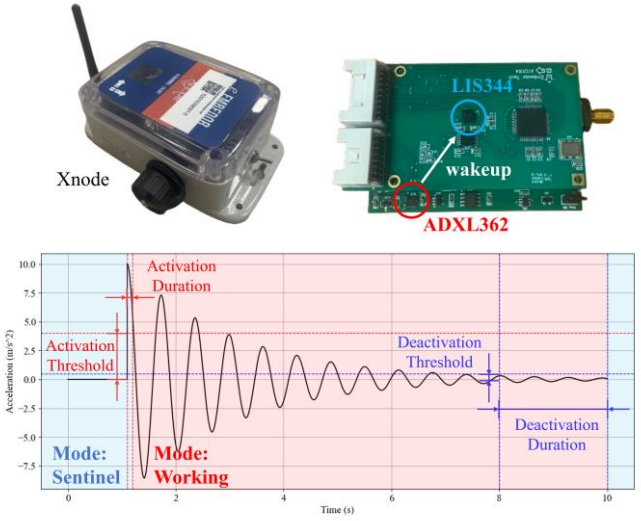


Figure 1. Xnode trigger sensing mechanism.

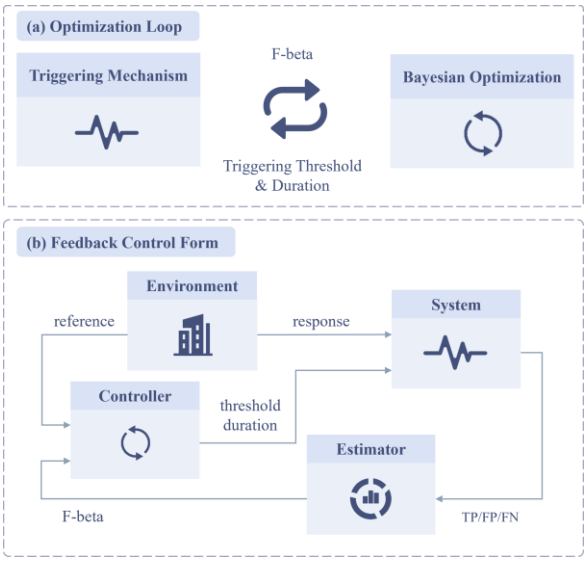
In SHM practice, the application of basic triggering mechanisms is typically accompanied by parameter configuration using empirical approaches. Engineers often perform preliminary sensing to gain insights into the target structure or environment, and subsequently configure the triggering parameters (i.e., threshold and duration) manually in a conservative manner to ensure low miss trigger rate [2]. This empirically tuned method also serves as the baseline for comparison in subsequent sections.

2.2 Feedback control powered by Bayesian optimization

As stated in the introduction, the goal of the triggering mechanism is to minimize missed triggers while keeping the false trigger rate at an acceptably low level, which constitutes a multi-objective optimization problem in its mathematical nature. To facilitate the analysis, the F_β score is introduced to quantitatively and comprehensively evaluate the performance of the triggering mechanism, as defined in Equation (1). Note that with the use of F_β, the original multi-objective problem is transformed into a single-objective optimization problem, where the parameter β controls the relative weighting between missed and false triggers. In SHM context, usually more weight should be put on recall, which means β should be configured greater than 1. For situations where precision is more important than recall, e.g., false triggering can be quite costly, β should be smaller than 1.

$$F_\beta = (1+\beta^2)\frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \quad (1)$$

Based on the analysis presented in the introduction, smart adaptive triggering centers on two key questions: what are the optimal triggering parameters, and how they can be efficiently approached during system operation. The first question represents an optimization problem, while the second constitutes a control problem. A major limitation of most existing triggering mechanisms lies in their lack of adaptivity, which can be effectively addressed through a feedback loop. As illustrated in Figure 2(a), the triggering mechanism outputs a performance metric to the optimizer, which in turn determines the next parameter configuration to explore based on historical observations. To comply with modern control theory [5], the closed loop can be expressed as Figure 2(b). Figure 2(b) depicts the formal structure of feedback control, comprising four main components: the environment, the system, the estimator, and the controller. In the context of trigger-based SHM, the environment refers to the structural responses that sensors aim to observe. The system corresponds to the triggering mechanism itself, as shown in Figure 1. The estimator is responsible for monitoring or estimating the performance of the triggering mechanism, specifically, the F_β score in this study. The controller, in turn, utilizes iterative feedback to search for the optimal parameter configuration and to devise strategies for efficiently approaching the optimal values. In short, feedback control serves as a backbone to integrate necessary tools to provide adaptivity for triggering parameter fine-tuning.



DOI: 10.3217/978-3-99161-057-1-094

Figure 2. Feedback control framework for triggering parameter optimization: (a) optimization loop; (b) feedback control form.

At the core of the feedback control architecture is the controller, which governs the adaptation process through parameter adjustment. While various tools can be used to implement the controller, the choice must be carefully tailored to the characteristics of the target problem. In the case of trigger-based SHM, the controller takes the F_β score as input and outputs the triggering threshold and duration. However, there is no explicit analytical expression linking the input to the output, making it a black-box optimization problem. Moreover, each feedback iteration requires calculating the F_β score based on many events, resulting in considerable computational cost, in addition to the overhead of observation and estimation. Given these considerations, Bayesian optimization [9] is adopted as the controller due to its demonstrated effectiveness in solving black-box problems and its ability to efficiently converge toward the global optimum.

The algorithmic framework of BO is presented in Algorithm 1. As illustrated, the optimization process is driven by the observation dataset $\boldsymbol{\mathcal{D}}$, which comprises input–output pairs, where the inputs are the triggering parameters, and the outputs are the corresponding F_β scores. The process begins with an initial sampling phase to obtain a preliminary understanding of the input–output relationship. Subsequently, the optimization proceeds iteratively, with each iteration updating the surrogate model $\boldsymbol{\mathcal{M}}$ and expanding the dataset with new evaluations. Typically based on Gaussian Process Regression (GPR), the surrogate model provides a non-parametric, probabilistic estimate of the objective function, offering both the predicted mean and associated uncertainty for any given input. This probabilistic nature enables GPR to model complex, non-linear relationships with relatively few samples, while maintaining analytical tractability and useful mathematical properties such as differentiability. To balance exploration of uncertain regions and exploitation of promising areas, an acquisition function $\boldsymbol{\mathcal{S}}$ is employed to determine the most informative point x_i. Common acquisition functions include Expected Improvement (EI), Probability of Improvement (PI), and Upper Confidence Bound (UCB), each offering a different strategy for leveraging the predictive mean and variance to guide sampling. Herein, UCB is used for simplicity. After evaluating the objective function to obtain the corresponding output y_i, the dataset $\boldsymbol{\mathcal{D}}$ is updated. Finally, the input $\hat{x}$ associated with the best observed output in $\boldsymbol{\mathcal{D}}$ is selected as the optimum.

Algorithm 1 Bayesian Optimization Algorithm Framework

Input: Search space $\boldsymbol{\mathcal{X}}$, objective function f, surrogate model $\boldsymbol{\mathcal{M}}$, acquisition function $\boldsymbol{\mathcal{S}}$
Output: Dataset $\boldsymbol{\mathcal{D}}$ (set of sampled points and their evaluations)
Initialize dataset: $\boldsymbol{\mathcal{D}} \leftarrow$ InitSamples(f, $\boldsymbol{\mathcal{X}}$)
for $i = |\boldsymbol{\mathcal{D}}|$ to T **do**
  Fit the model: $p(y|\boldsymbol{x}, \boldsymbol{\mathcal{D}}) \leftarrow$ FitModel($\boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{D}}$)
  Select next point: $x_i \leftarrow \arg max_{x \in \boldsymbol{\mathcal{X}}} \boldsymbol{\mathcal{S}}(\boldsymbol{x}, p(y|\boldsymbol{x}, \boldsymbol{\mathcal{D}}))$
  Evaluate objective function: $y_i \leftarrow f(x_i)$
  Update dataset: $\boldsymbol{\mathcal{D}} \leftarrow \boldsymbol{\mathcal{D}} \cup \{(x_i, y_i)\}$
end for
Pick the best from $\boldsymbol{\mathcal{D}}$ as optimum: $\hat{x} \leftarrow \arg max_{(x,y) \in \boldsymbol{\mathcal{D}}} y$

After introducing the algorithmic details, it becomes evident that the surrogate model $\boldsymbol{\mathcal{M}}$ plays a central role in addressing the black-box nature of the problem by providing a reliable approximation of the true input–output relationship along with favorable mathematical properties. Meanwhile, the acquisition function $\boldsymbol{\mathcal{S}}$ serves as the foundation for efficient convergence toward the global optimum, as it guides the search process in a principled manner rather than relying on random exploration.

As can be inferenced by Algo. 1, the computational efforts required for BO for the following edge deployment is mainly determined by the size of observation dataset $\boldsymbol{\mathcal{D}}$, and each sample in this dataset is only a pair of triggering parameters and performance metric, meaning it can be easily operated on edge devices.

2.3 Digital twin for data augmentation

Feedback control is a powerful tool to address the lack of adaptivity; however, it falls short in tackling another critical challenge—data scarcity, imbalance, and uncertainty. Digital twin technology offers a promising solution to augment data for a variety of purposes, such as simulation, optimization, and neural network training. To effectively construct a digital twin for trigger-based sensing for SHM, it is essential to accurately simulate both the structural response and the triggering mechanism. Specifically, this involves modeling the excitation–structure–response chain as well as the threshold–duration-based triggering logic, as shown in Figure 3.

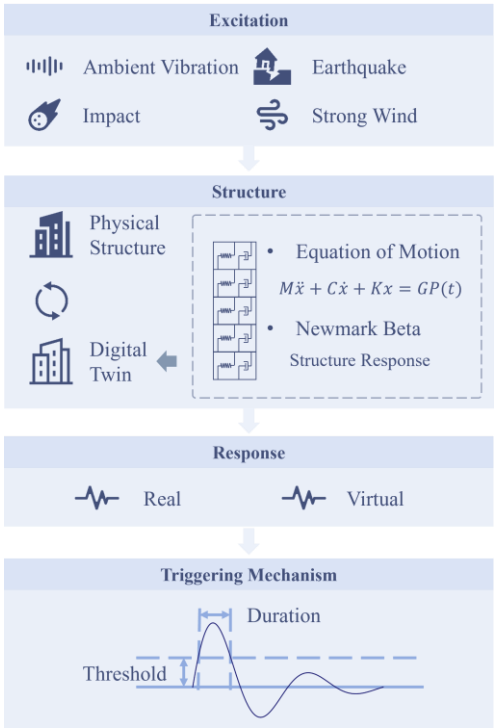


Figure 3. Digital twin for excitation-structure-response flow and triggering mechanism.

Excitation simulation is the first step in structural response modeling. To support the study, several common types of events are considered, including ambient vibrations, earthquakes, impacts, and strong winds. Each event type can be

DOI: 10.3217/978-3-99161-057-1-094

generated using either external databases or simulation techniques. Ambient vibrations are typically modeled using Gaussian white noise, while earthquakes can be synthesized by superimposing harmonic waves based on a given spectrum. Impacts are represented as impulsive loads, and strong winds can be simulated using autoregressive models combined with time modulation.

Structure modeling is another crucial step in response simulation, achieved by formulating the equation of motion, as shown in Equation (2). In this equation, M, C, K represent the mass, damping, and stiffness matrices, respectively. G denotes the force allocation matrix, and $P(t)$ represents the external force vector at time *t*. Equation (2) thus serves as a digital twin of the target structure to be monitored. Following structural modeling is the simulation of structural responses, which involves calculating the displacement, velocity, and acceleration of each degree of freedom in the constructed digital twin. A widely used and reliable numerical method for this purpose is the Newmark-beta method [11], which provides accurate structural response simulation.

$$M\ddot{x} + C\dot{x} + Kx = GP(t) \tag{2}$$

In addition to structural responses, the basic triggering mechanism must also be simulated to complete the digital twin, with implementation details provided in Algorithm 2. With a comprehensive digital twin that includes both the structural model and the triggering logic, data can be flexibly manipulated for various purposes, such as simulation, optimization, neural network training, and more.

Algorithm 2 Triggering Mechanism

Input: signal value s, threshold τ, duration d
Output: trigger flag T (binary: 1 for trigger, 0 for no trigger; initialized as 0)
Internal Variable: counter c (initialized as 0, used to track consecutive time steps)
For each time step t:
if $|s| \geq \tau$ **then**
  Increment counter: $c \leftarrow c + 1$
  if $c \geq d$ **then**
    Set trigger flag: $T \leftarrow 1$
    Reset counter: $c \leftarrow 0$
  end if
else
  Reset counter: $c \leftarrow 0$
  Set trigger flag: $T \leftarrow 0$
end if

2.4 Lightweight neural networks for onboard inference

Feedback from the triggering mechanism is essential for achieving adaptivity; however, its effectiveness is often hindered by challenges in real-world deployment. As shown in Equation (1), the computation of the F_β score relies on both precision and recall, each of which faces practical difficulties. Specifically, precision depends on the identification of true positives and false positives. In the absence of human intervention, ground truth labels for captured signals are typically unavailable, making it impossible to determine whether a triggered signal corresponds to an actual event of interest. This uncertainty undermines the ability to accurately evaluate performance metrics such as precision and, consequently, the F_β score. The situation is even more challenging for recall, which relies on identifying both true positives and false negatives as shown in Equation (4). Estimating false negatives necessitates knowledge of missed events of interest—information that is inherently inaccessible in trigger-based sensing systems. These limitations present fundamental barriers to autonomous online adjustment of triggering parameters, underscoring the importance of developing effective solutions.

$$precision = \frac{TP}{TP+FP} \tag{3}$$

$$recall = \frac{TP}{TP+FN} \tag{4}$$

Edge intelligence, which integrates lightweight onboard computation with AI, offers a promising pathway for achieving online fine-tuning on edge devices, particularly for problems beyond the reach of traditional approaches. To address the issue of precision, the absence of ground truth can be mitigated by deploying an onboard AI classifier to infer signal labels. Similarly, for recall, an AI-based estimator can be employed to directly approximate the recall value, thereby facilitating more accurate performance evaluation. Most importantly, the onboard AI enables autonomous feedback generation without the need for human intervention, thereby streamlining operations and advancing full system automation. To meet the requirements of edge deployment, these neural networks should prioritize lightweight architecture and high computational efficiency.

Table 2. NN parameters summary.

Parameter Type	CNN	DNN
Total Parameters	142(572.00 B)	209 (836.00 B)
Trainable Parameters	44 (176.00 B)	209 (836.00 B)
Non-trainable Param.	8 (32.000 B)	0 (0.00 B)
Optimizer Parameter	90 (364.00 B)	0 (0.00 B)

To effectively classify time series data, a 1D Convolutional Neural Network (CNN) combined with feature engineering is employed. Given an input signal of 6000 samples, the data is first transformed from the time domain to the frequency domain using the Fast Fourier Transform (FFT). Both time- and frequency-domain signals are then downsampled to 64 samples each. The resulting vectors are concatenated into a 128-dimensional feature vector, which serves as the input to a lightweight CNN classifier. Details of the network size and architecture are provided in Table 2 and Table 3, respectively. The training curves and classification results are presented in Figure 4.

Table 3. CNN classifier architecture.

Layer	**Output Shape**	**Para. Number**
Input Layer	128	0
Conv1D	128	16
BatchNorm	128	16
ReLU	128	0
GlobalAvgPooling	4	0
Dense	4	20

DOI: 10.3217/978-3-99161-057-1-094

Table 4. DNN Recall estimator architecture.

Layer	Output Shape	Para. Number
Dense	16	64
Dense	8	136
Dense	1	9

Since false negatives cannot be directly observed in trigger-based sensing, a Dense Neural Network (DNN) is introduced to estimate recall based on noise level, trigger threshold, and duration. Utilizing the dataset generated from the digital twin, the DNN is trained to predict recall under various noise conditions and triggering configurations. The network size is detailed in Table 2, the architecture is provided in Table 4, and the training curves are shown in Figure 4. With both the CNN classifier and the DNN recall estimator trained, the feedback loop is fully established, completing the final component of the closed-loop control system.

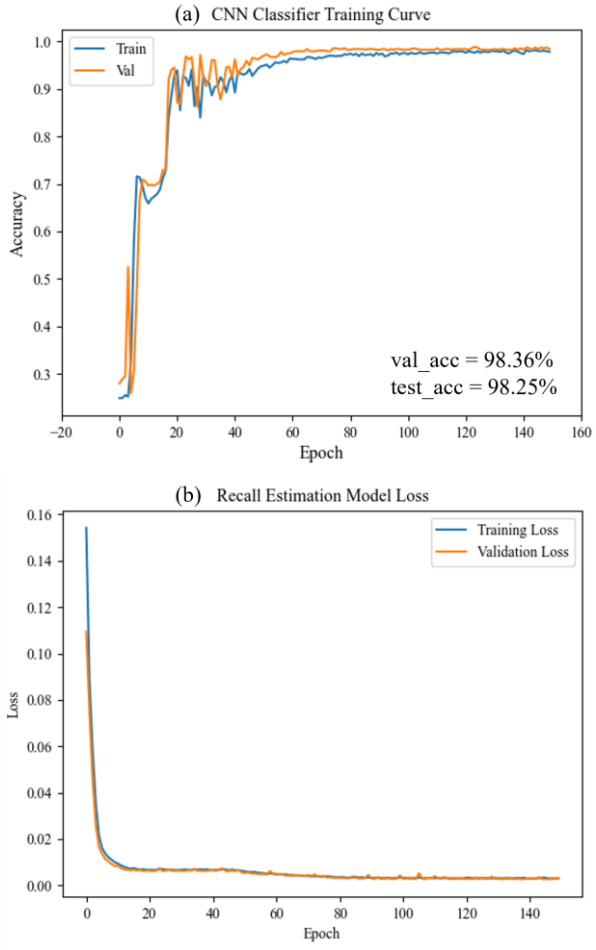


Figure 4. NN training curves: (a) CNN classifier (b) DNN estimator.

3 STAGED DEPLOYMENT

Despite the many advantages of edge intelligence, its most prominent inherent limitation lies in the constrained onboard resources, which restrict the efficiency of triggering parameter optimization. In the previous section, a feedback control approach was introduced for iterative optimization of triggering parameters. To further accelerate this process, a staged deployment strategy is proposed, comprising a pre-deployment optimization stage and an onboard fine-tuning stage. The second stage inherits insights obtained during the first, allowing only lightweight fine-tuning on the device and thus minimizing computational overhead.

Figure 5 illustrates the concept of this staged optimization strategy. Both stages are built upon the same feedback control framework. The key differences lie in two components: the environment component, which refers to the excitation-structure-response flow, and the estimator component, which is responsible for providing performance metrics. In Stage I, the environment is simulated using a digital twin. Although the structural responses are not real, this fully controlled setting provides complete knowledge of all events, enabling accurate performance evaluation. In Stage II, the environment becomes real and uncertain. Since ground truth labels are unavailable, recall cannot be directly computed. To overcome this limitation, onboard neural networks including the CNN classifier and the DNN-based recall estimator are employed to complete the closed-loop feedback control. It is important to note that in the trigger sensing setup, the DT model is not required to be of high fidelity, as its primary role is to generate synthetic data for training the classification model.

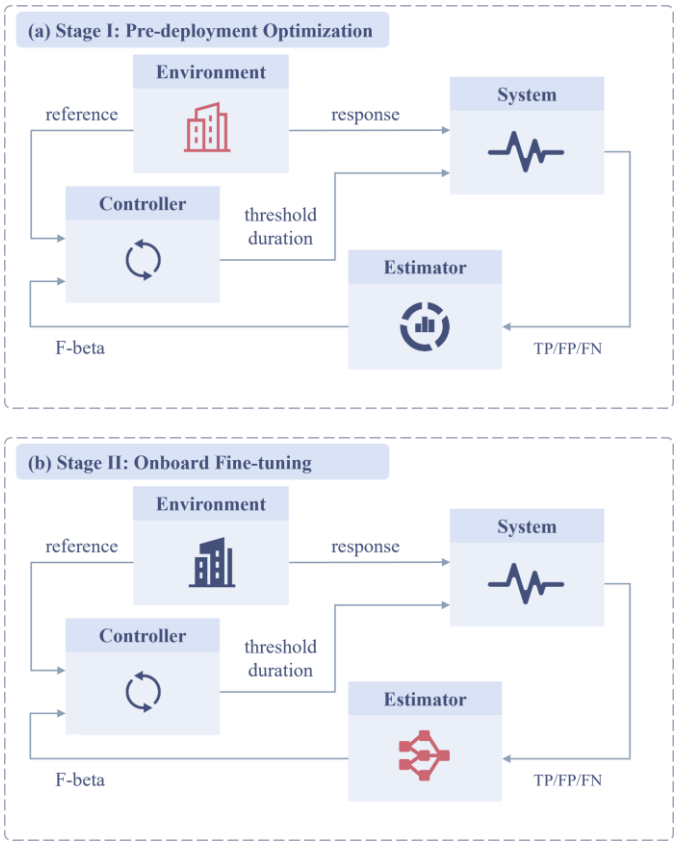


Figure 5. Staged deployment strategy: (a) pre-deployment optimization (b) onboard fine-tuning.

DOI: 10.3217/978-3-99161-057-1-094

4 VALIDATION AND RESULTS

4.1 Dataset generation

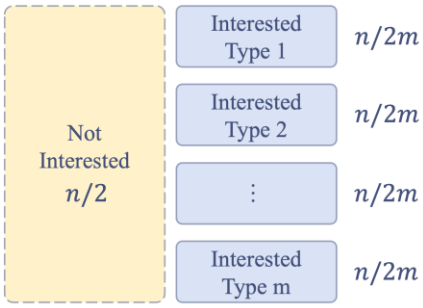


Figure 6. Dataset generation.

As previously discussed, the uncertainty of data distribution in real-world deployment makes it unsuitable for reliable evaluation of triggering mechanism performance. To enable consistent and controlled assessment, a synthetic dataset with a predefined distribution was constructed, as illustrated in Figure 6. Let n denote the total number of event samples, and m represent the number of event types considered as events of interest. Events of no interest, represented by ambient vibration in this context, account for half of the dataset (n/2 samples), while the remaining half is evenly divided among the m types of events of interest. In this study, n is set as 1200 for validation. For ambient vibration signals, data is generated using a Gaussian distribution with a mean of zero and a standard deviation derived from real-world ambient data. This approach aims to closely approximate actual deployment conditions.

In this evaluation setup, three types of events are defined as events of interest: earthquake, impact, and strong wind, resulting in m = 3 and n/6 samples for each type. These signals are synthesized using a uniform distribution based on peak value ranges, allowing for diverse intensity levels within each event type. This structured and realistic data generation strategy ensures a balanced and reproducible dataset, enabling consistent evaluation of triggering mechanisms across a wide range of simulated scenarios.

4.2 Host Devices and Implementation Procedures

According to the proposed staged deployment strategy, the first stage focuses on preliminary optimization of triggering parameters using the digital twin in a resource-rich environment, such as a personal computer or workstation. The second stage is dedicated to real-time fine-tuning in real-world conditions on resource-constrained edge devices. To emulate this setup in the evaluation, Stage I was carried out on a personal computer, while Stage II was implemented on LiftNode, a low-cost microcontroller-based AIoT sensor node developed by the Laboratory of Intelligent Infrastructure at Nanyang Technological University. More specifically, the implementation is based on a dedicated middleware named TinySHM, which is currently under active development and features a hierarchical structure of basic utilities (e.g., time management, communication), mathematical operations, signal processing, and AI support (accelerated by ESP-DL library [12]). The specifications of the host devices, including main frequency and RAM capacity, are summarized in Table 5.

Table 5. Host devices specifications.

Layer	Main Freq.	RAM
PC	2.50 GHz	32 GB
LiftNode-ESP32	480 MHz	8 MB

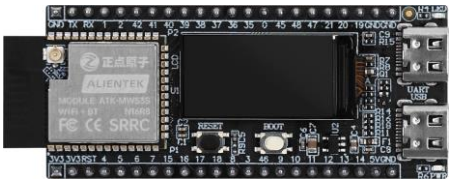


Figure 7. The main control board of LiftNode-ESP32.

ESP32 features the capability for Wi-Fi Connection designed for IoT applications. By combining onboard SD card module for large volume data storage and IoT capabilities, the proposed framework can achieve large dataset storage via Internet streaming and record-by-record onboard processing within limited onboard memory. In short, the implementation adopted a 'more time for less space' strategy, using longer time consumption to address the limits of restrained onboard resources. This is reasonable, as the appearance of interested events can be sparse during monitoring, leaving enough time for edge device to process. Besides, the event records are only used to provide classification label, the core part for adaptive optimization is BO, which only focuses on the observation dataset $\mathcal{D}$, simply data pairs of trigger parameters (threshold and duration) and performance metric (F_β). Usually, it requires hundreds or thousands interested events to update the observation dataset $\mathcal{D}$ once, which means low demand for computational capability on edge devices.

In this research, each record was formulated to 1 min long, assuming most events will not exceed this duration. Theoretically, even the time history data is longer than 1 min, the onboard NN can still easily provide the type of events using partial data, showing the robustness of the proposed framework design. For each triggered and recorded event, the host device determined its type, and update the counting variable. Only when there are enough number of events recorded, the host machine will conduct BO for triggering parameter updating.

4.3 Results

The detailed configurations for the optimization process is listed in Table 6, and the optimization process is recorded and visualized in Figure 8 and Figure 9. As illustrated in Figure 8, red dots represent Stage I optimization results and blue dots represent Stage II results. To accelerate convergence, a bonus factor of 1.1 is applied to the final F_β score for iterations where both precision and recall exceed 90%. The final results including the baseline approach and breakdown results of stage I and II are presented in Table 9.

Table 6. Validation configurations.

Parameter	**Description**	**Value**
noise std	ambient vibration	2.92~3.57x10^{-3} g
EQ peak val.	earthquake	0.1 ~ 3.0 g
IP peak val.	impact	0.1 ~ 3.0 g
SW peak val.	wind (x noise std)	1.5 ~ 2.5
beta	beta in Eq (1)	5
IniNum	# initial observation	15

DOI: 10.3217/978-3-99161-057-1-094

IterNum1	# iteration in stage I	50
IterNum2	# iteration in stage II	20
bonus factor	for p>0.9 & r>0.9	1.1
τ_{lb}	threshold lower bound	0
τ_{ub}	threshold upper bound	0.01706
d_{lb}	duration lower bound	2
d_{ub}	duration upper bound	10

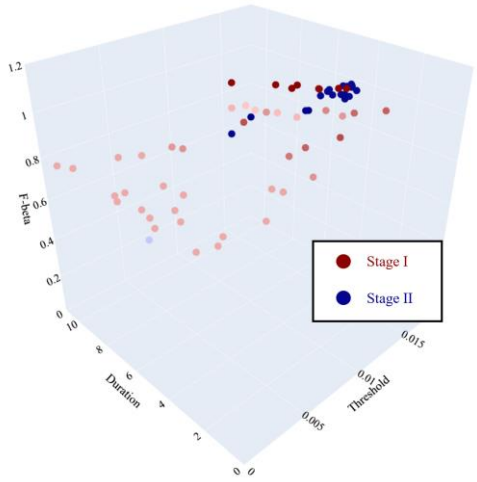


Figure 8. Optimization result overview.

Table 7. Validation results.

Item	F-beta	Precision	Recall
Baseline	0.8025	50%	100%
Stage I	1.0808	96.33%	98.37%
Stage II	1.0511	92.54%	95.72%

As shown in Figure 9 and Table 7, the baseline method, marked in black, reflects a conservative conventional configuration. While it achieves high recall, this comes at the cost of significantly reduced precision, resulting in a high false trigger rate. In contrast, the proposed staged optimization framework effectively addresses this limitation. Several key observations can be drawn from the results.

Superior Performance of SATM. The proposed SATM framework demonstrates clear advantages over the conventional approach, achieving F_β scores of 1.0808 during pre-deployment optimization and 1.0511 during onboard optimization, significantly outperforming the baseline score of 0.8025. These results highlight the effectiveness of SATM in optimizing triggering parameters for SHM applications.

Strong Synthetic-to-Real Transferability. The similarity between data distributions in the pre-deployment phase (based on the digital twin) and the real-world onboard phase indicates strong transferability of the optimization strategy. Although some deviations are observed due to inherent differences between synthetic and real data, such discrepancies are expected to decrease as real-world data continues to accumulate over time.

Highly optimized energy-efficiency ratio. For long-term, battery-powered monitoring, a trigger-based scheme is significantly more energy-efficient than always-on or duty-cycled approaches. By adaptively optimizing the triggering parameters, the overall performance can be quantitatively evaluated using a dedicated performance metric. Depending on the initial parameter settings, energy consumption may decrease if the parameters are overly strict, or increase if they are too loose. However, one thing remains consistent: the energy-efficiency ratio improves, as reflected by the performance metric.

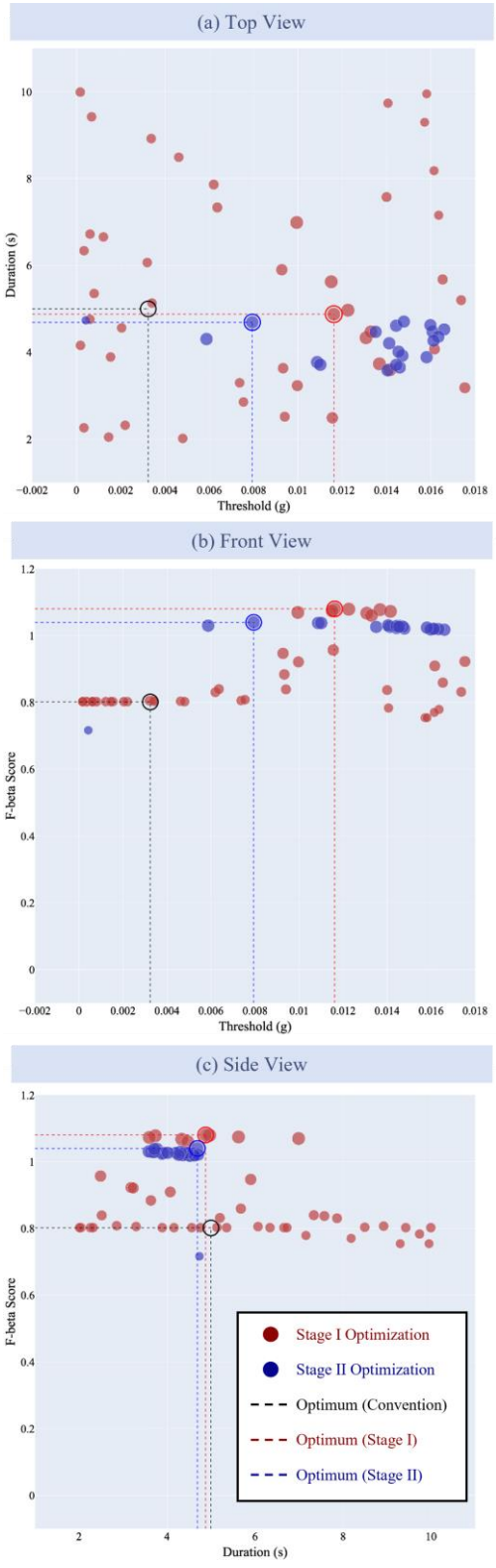


Figure 9. Optimization results: (a) top view, (b) front view, (c) side view.

DOI: 10.3217/978-3-99161-057-1-094

Efficiency of the Surrogate Model. The surrogate model, trained within the digital twin environment, significantly enhances the efficiency of onboard fine-tuning. It enables the optimization process to converge in fewer iterations compared to the initial stage, achieving rapid progression toward high-quality configurations and yielding improved average performance during real-world deployment.

Balanced Exploration and Exploitation. The two-stage deployment strategy effectively balances the trade-off between exploration and exploitation. The first stage emphasizes exploration, exhibiting greater variability in performance metrics to thoroughly investigate the parameter space. In contrast, the second stage focuses on refining configurations based on prior knowledge, resulting in stable and robust system performance.

5 CONCLUSION

This study introduces a smart adaptive triggering mechanism that seamlessly combines feedback control, digital twin modeling, and Bayesian optimization. It is specifically designed to tackle key challenges such as limited adaptivity, multi-objective optimization, unknown event distributions, lack of ground truth, partial observability, and high data acquisition costs. SATM is structured around four core components: the environment, the system, the estimator, and the controller. The mechanism functions through two sequential phases: an initial pre-optimization phase utilizing a digital twin to identify a strong baseline configuration, followed by an onboard fine-tuning phase that adapts parameters under real-world deployment conditions. Evaluations confirm that SATM achieves substantial improvements in triggering performance, delivering approximately a 30% increase in the F_β score compared to traditional approaches. These results underscore the promise of SATM in enabling automatic, adaptive parameter tuning for trigger-based sensing systems across various application domains. In future, the proposed method will be used for long term monitoring uses and incorporate more types of events, e.g., structure failure.

APPENDICES

Excitation-Structure-Response Simulation:
https://github.com/Shuaiwen-Cui/Research-Excitation_Structure_Response.git

Smart Adaptive Trigger Sensing:
https://github.com/Shuaiwen-Cui/Research-Smart_Adaptive_Trigger_Sensing.git

ACKNOWLEDGMENTS

The authors want to gratefully acknowledge the financial support from NTU Start-up Grant (03INS001210C120), CoE Dean's Interdisciplinary Grant 2024 (03INS002320C120), and MOE AcRF Tier 1 Grants (RG121/21).

REFERENCES

[1] Y. Fu, "Sudden-event monitoring of civil infrastructure using wireless smart sensors," Thesis, University of Illinois at Urbana-Champaign, 2019. Accessed: Aug. 03, 2024. [Online]. Available: https://hdl.handle.net/2142/106325

[2] Y. Fu, T. Hoang, K. Mechitov, J. R. Kim, D. Zhang, and B. F. Spencer, "Sudden Event Monitoring of Civil Infrastructure Using Demand-Based Wireless Smart Sensors," *Sensors*, vol. 18, no. 12, Art. no. 12, Dec. 2018, doi: 10.3390/s18124480.

[3] Y. Ni, X. Liu, and C. Yang, "Sensor Scheduling for Remote State Estimation with Limited Communication Resources: A Time- and Event-Triggered Hybrid Approach," *Sensors*, vol. 23, no. 21, Art. no. 21, Jan. 2023, doi: 10.3390/s23218667.

[4] S. Wang and K. Li, "Constrained Bayesian Optimization Under Partial Observations: Balanced Improvements and Provable Convergence," arXiv.org. Accessed: Sep. 04, 2024. [Online]. Available: https://arxiv.org/abs/2312.03212v2

[5] K. Ogata, *Modern control engineering*. in Instrumentation and controls series. Prentice Hall, 2010. [Online]. Available: https://books.google.com.sg/books?id=Wu5GpNAelzkC

[6] R. Ganguli, S. Adhikari, S. Chakraborty, and M. Ganguli, *Digital Twin: A Dynamic System and Computing Perspective*. Taylor & Francis Group, 2023. [Online]. Available: https://books.google.com.sg/books?id=0o9RzwEACAAJ

[7] S. Cui, T. Hoang, K. Mechitov, Y. Fu, and B. F. Spencer, "Adaptive edge intelligence for rapid structural condition assessment using a wireless smart sensor network," *Engineering Structures*, vol. 326, p. 119520, Mar. 2025, doi: 10.1016/j.engstruct.2024.119520.

[8] C. M. Bishop and H. Bishop, *Deep Learning: Foundations and Concepts*. Cham: Springer International Publishing, 2024. doi: 10.1007/978-3-031-45468-4.

[9] T. Agrawal, "Bayesian Optimization," in *Hyperparameter Optimization in Machine Learning*, Apress, Berkeley, CA, 2021, pp. 81–108. doi: 10.1007/978-1-4842-6579-6_4.

[10] AnalogDevices, "ADXL362 Datasheet and Product Info | Analog Devices." Accessed: Dec. 25, 2024. [Online]. Available: https://www.analog.com/en/products/adxl362.html

[11] N. M. Newmark, "A Method of Computation for Structural Dynamics," *Journal of the Engineering Mechanics Division*, vol. 85, no. 3, pp. 67–94, Jul. 1959, doi: 10.1061/JMCEA3.0000098.

[12] Espressif, *ESP-DL*. (Feb. 25, 2025). Accessed: Feb. 25, 2025. [Online]. Available: https://components.espressif.com/components/espressif/esp-dl