BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 72(1), 2024, Article number: e148251
DOI: 10.24425/bpasts.2023.148251

Real-time operational load monitoring of a composite aerostructure using FPGA-based computing system

Waldemar MUCHA*

Department of Computational Mechanics and Engineering, Silesian University of Technology, Konarskiego 18A, 44-100 Gliwice, Poland

Abstract. Operational load monitoring (OLM) is an industrial process related to structural health monitoring, where fatigue of the structure is tracked. Artificial intelligence methods, such as artificial neural networks (ANNs) or Gaussian processes, are utilized to improve efficiency of such processes. This paper focuses on moving such processes towards green computing by deploying and executing the algorithm on low-power consumption FPGA where high-throughput and truly parallel computations can be performed. In the following paper, the OLM process of typical aerostructure (hat-stiffened composite panel) is performed using ANN. The ANN was trained using numerically generated data, of every possible load case, to be working with sensor measurements as inputs. The trained ANN was deployed to Xilinx Artix-7 A100T FPGA of a real-time microcontroller. By executing the ANN on FPGA (where every neuron of a given layer can be processed at the same time, without limiting the number of parallel threads), computation time could be reduced by 70% as compared to standard CPU execution. Series of real-time experiments were performed that have proven the efficiency and high accuracy of the developed FPGA-based algorithm. Adjusting the ANN algorithm to FPGA requirements takes some effort, however it can lead to high performance increase. FPGA has the advantages of many more potential parallel threads than a standard CPU and much lower consumption than a GPU. This is particularly important taking into account potential embedded and remote applications, such as widely performed monitoring of airplane structures.

Keywords: operational load monitoring; real time computations; FPGA; aerostructures; neural networks; artificial intelligence; structural health monitoring; green computing.

1. INTRODUCTION

Structural health monitoring (SHM) stands for the industrial processes that allow early detection of anomalies in mechanical structures in order to make early decisions of required maintenance that reduce cost. The features of SHM systems can include, depending on the needs: (a) damage detection, (b) damage localization, (c) damage classification, and (d) determining the remaining life of the structure [1]. The following paper focuses on the last of the aforementioned features of SHM, which is often referred to as operational load monitoring (OLM). It is an industrial process that allows to make predictions about the remaining operational time of a structure, before fatigue failure will constitute a risk [2]. The most important application area of this process is in the aerospace industry, where the goal is operational life extension of aircraft components and ensuring structural safety [3]. The classic approach was to replace structure elements in planned maintenance actions after a pre-defined period of time. The benefits of performing OLM are extending the in-service life of structures in case they wear out later than expected, and increasing safety if the fatigue happened earlier than expected. OLM is often, but not always, performed simultaneously with other SHM processes such as early stage damage detection or localization [4–6]. In the aerospace industry both processes are handled by health and usage monitoring systems (HUMS) [7, 8]. Health state and life consumption of a structure are continuously monitored using a set of permanently installed sensors and data processing systems. Different sensor types can be utilized in HUMS, including accelerometers and strain sensors [9, 10]. For OLM purposes, either surface-mounted (e.g. strain gauges) [11] or intrinsic (e.g. FBG – fiber Bragg grating optical fiber) [12, 13] strain sensors are usually applied. FBG sensors are considered better for avionic applications than strain gauges as they emit more stable signal – less affected by electric fields occurring at high altitudes caused for example by lightnings. The other advantage is that they can be easily integrated with modern aerospace composite materials such as fiber reinforced polymers that are lightweight but susceptible to barely visible impact damage [14].

Other application areas for the monitoring processes are civil structures (e.g. bridges, pipes) [15, 16], wind turbines [17] and underground vehicles for the mining industry [18].

Machine learning (ML) revolutionized the applications of SHM (including OLM) techniques by automating the processes and limiting the effects of noises [19]. ML is used to build intelligent models that map the input patterns from sensor measurements to output desired targets, e.g. damage assessment or current load value [20, 21].

Different types of artificial neural networks (ANNs) or support vector machines (SVR) are usually used for damage detection and classification. ANN is a bio-inspired algorithm that mimics the operation principles of an animal brain. It is a col-

*e-mail: waldemar.mucha@polsl.pl

Manuscript submitted 2023-07-31, revised 2023-10-19, initially accepted for publication 2023-10-30, published in February 2024.

© 2024 The Author(s). This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)

lection of units called artificial neurons, connected in a hierarchical order. Each artificial neuron is a mathematical model of a brain cell, which receives input signals, processes them and outputs signals to neurons connected to it. The output signal is computed by a linear or non-linear activation function of the weighted sum of the inputs [22]. In the learning process, the weights are adjusted in order for the network to fit the reference data as accurately as possible. Tabian *et al*. [23] implemented a convolutional neural network (CNN) and passive sensing for impact detection and location in composite structures, obtaining 95% accuracy. Damm *et al.* [24] used MEMS and a piezoelectric transducer sensor integrated with a carbon fiber reinforced composite structure and trained CNN. Jung and Chang [25] described CNN for impact localization based on piezoelectric ribbon sensor measurements. Khan *et al.* [26] implemented deep learning (DL) neural networks for vibration-based classification of delamination in composite laminates with the accuracy of 90%. Yu and Kim [27] used DL for damage detection in composite structures with the accuracy of 95%. SVMs are classifiers that map training data to points in space and try to maximize the gap between the categories. When input data are introduced, they map it to the same space and predict their category based on which side of the gap the data fell. This methods were successfully utilized for impact localization from FBG measurements by Datta *et al.* [28]. Mardanshahi *et al.* [29] compared effectiveness of using SVM and two types of ANNs for the detection and classification of matrix clacking in a glass fiber/epoxy composite.

For OLM processes ML techniques are also commonly applied. They often allow to reduce the number of required sensors. In order to perform direct and accurate OLM, sensors should be mounted in every critical point of the structure. Critical point means a place where possible stress and strain concentrations can occur during the service of the structure. As the loads of a structure can be highly variable, including locations of the acting forces, the number of critical points can be too high to handle. For this reason it is a common approach to involve ML prediction models. They allow to obtain required load estimate (e.g. stress at certain location, location and value of acting force, safety factor of the structure under current load) based on limited input data (e.g. discrete strain measurements) [30]. Two most commonly used ML methods for OLM processes are ANNs and Gaussian processes (GP) [4, 31]. Candon *et al.* [32] used ANNs to predict transonic buffet (unavoidable aerodynamic nonlinearity) in the process of track fatigue life reduction using accelerometer measurements as inputs. Wada *et al.* [33] employed ANNs to identify load distributions of a simple plate by using strain measurements from a relatively large number (60) of strain gauges. Gaussian processes, on the other hand, are ML methods that can be used for regression and are based on the Bayesian framework, where the predictive output is provided based on prior conditional probability. It is assumed that the probability distribution for any input values over the output follows the Gaussian distribution. GPs use the whole training data to make predictions based on similarity between points [34]. Holmes *et al.* [35] implemented GP to predict loads of landing gear where the model was trained on experimentally gathered data. Fuentes *et al.* [36] used GP for prediction of aircraft structural loads from recorded flight parameters. Predicted strains were compared with measured values and compliance was achieved.

The novelty of the presented paper is moving OLM process towards green computing by performing high-throughput, true parallel processing of artificial neural networks in real time on a low energy consumption FPGA-based microcontroller. Green computing means a trend of using computing devices in energy-efficient and eco-friendly ways. The awareness of the global need to reduce energy consumption wherever it is possible is constantly rising as fossil fuels supply most of the world's energy [37, 38]. According to Google Trends, 'green computing' was searched 93% more in 2022 than in 2020 (when comparing average week searches).

Due to the applications of OLM processes (for example in aircrafts during flight, together with SHM as part of HUMS), it seems natural to perform all the necessary computations on an embedded system connected to the sensors. In this paper, an FPGA-based embedded computing system is proposed to perform ANN-based OLM in real-time. FPGA stands for field-programmable gate array and it is an integrated circuit with functionality programmable and re-programmable by the user. It contains an array of programmable logic blocks with reconfigurable connections [39]. FPGA computing is definitely classified as green computing [40–42] as the circuits are characterized by low power consumption and high performance and reliability. Other advantages include low cost, no need for the operation system to execute the logic, and flexibility (upgradable logic). FPGAs are commonly implemented in embedded applications, they were even placed on Mars rovers [43, 44]. The specific structure and working principles of FPGAs make them perfect for parallel processing as simultaneous parallel circuits can be running while executing the logic (their number being limited only by the availability of hardware resources) with no CPU time sharing. Modern CPUs have several or several dozen of parallel threads, while in FPGAs the number of independent circuits can reach even hundreds or thousands (like in GPUs but with the power consumption lower by multiple orders of magnitude). Running ANNs on FPGAs is an idea that has been emerging in recent years [45, 46]. Project Brainwave by Microsoft is a deep learning platform for cloud computing with FPGA acceleration, for computer vision and natural language processing applications [47, 48].

FPGA data processing can be definitely considered as green computing as there are many references where energy efficiency between multicore CPUs, GPUS and FPGAs is compared. Betkaoui *et al.* [49] compared a GPU-based system and a heterogeneous system with a CPU and FPGA co-processor and concluded that GPU has got better floating-point performance but FPGA is better for parallel applications requiring low memory bandwidth. Qasaimeh *et al.* [50] compared embedded devices based on ARM CPU, GPU and FPGA for image processing applications in embedded vision solutions. For more complex vision pipelines FPGAs outperformed GPUs and CPUs – reduction by 1.2× to 22.3× of energy/frame was achieved. In papers [51, 52] FPGAs provided the best energy efficiency in al-

most all situations, as compared to GPUs and multicore CPUs, being one or more order of magnitude higher in some cases. Tang *et al.* [53] measured energy efficiency for convolutional neural networks processing on different devices and obtained the following results: 0.119 GFLOPS/W for CPU, 0.522 GFLOPS/W for GPU, and 4.723 GFLOPS/W for FPGA, proving it to be far more superior.

The applicability of the presented research, where FPGAs could be employed to improve OLM/SHM processes, covers different areas – civil structures, vehicles and wind turbines, but first of all it applies to aerospace structures. For this reason electromagnetic interference, which can occur at high altitudes (from lightnings, solar flares, electrostatic discharges, radiated fields of radars etc.) and cause malfunction of electronic devices, must be taken into account in all aspects of avionic design and certification [54]. FPGAs, if not properly protected, are susceptible to SEUs (single-event upsets) which mean changes of state caused by ionizing particle striking an electronic device and leading to errors [55]. Yuan *et al.* [56] measured the influence of electromagnetic behavior on FPGAs. However, modern FPGA devices with certified SEU immunity dedicated to aerospace applications are available in the market. Well-known FPGA companies such as Microchip and Xilinx are producing devices dedicated to avionic applications [57, 58].

Petko and Uhl [59] described smart sensor development for avionic OLM purposes. The smart sensor is built on an FPGA microchip where an ANN was deployed to identify loads of a structure in real time based on measurements from piezoelectric accelerometers. Accuracy of FPGA computations is validated by comparing the results to offline PC calculations. The novelty of the following paper is that not only the accuracy of FPGA algorithm is measured but also its efficiency is evaluated by comparing the computational time of ANN processing on FPGA and a microprocessor of an industrial real-time microcontroller.

In the following paper the OLM process for a composite hat stiffened panel (a structure of typical aerospace application) is described. A relatively small amount of strain gauges was mounted to the structure. An artificial neural network was trained to predict the current Tsai-Wu (failure criterion for composites) value of the panel under a load based on strain measurements. To train ANN, reference data were generated using an accurate (experimentally validated) numerical model, based on the finite element method [60]. Series of experiments were performed where the real-time National Instruments cRIO-9054 microcontroller with Xilinx Artix-7 A100T FPGA was utilized. The trained ANN was deployed to the FPGA algorithm allowing for its high-throughput and true parallel execution in real time. The efficiency and accuracy of executing the ANN using FPGA was measured and compared to standard CPU execution.

Details about running algorithms based on ANNs in FPGAs in a parallel mode are presented in Section 2. Section 3 focuses on the OLM example, including structure and prediction model description, details about converting and deploying the trained ANN to the FPGA, results of benchmarking the high-throughput FPGA version of the algorithm as compared to standard CPU execution, and results of series of real-time OLM experiments.

2. EASE OF USE IN HIGH THROUGHPUT FPGA PROCESSING OF ANNS

ANNs are bio-inspired algorithms that mimic the operation principles of an animal brain. They are collections of connected units called artificial neurons. A model of a single neuron i was presented in Fig. 1. The neuron receives N signals x_j from inputs, processes them with a linear or (usually) nonlinear function of the weighted sum of inputs, called activation function φ, and outputs the processed information y_i. The activation function is the mathematical model of a biological neuron:

$$y_i = \varphi(e) = \varphi\left(\sum_{j=1}^{N} w_{ij}x_j + B\right), \tag{1}$$

where e is the neuron's net value, w_{ij} are weights for each input x_j, and B is bias.

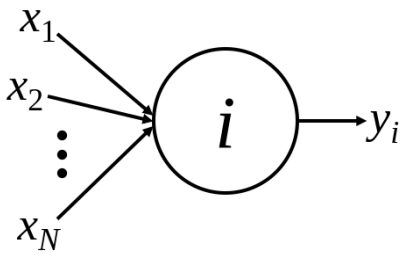


Fig. 1. Model of artificial neuron

In ANNs the artificial neurons are organized in layers. The first is the input layer where the number of neurons is equal to the number of inputs. Signals from the input layer go through one or more hidden layers (the number and size of hidden layers depends on the task the network is performing) to the output layer (the size of the output layer is dependent on the number of outputs). Neural networks can be feedforward if the signals move only to subsequent layers, or recurrent if there are feedback signals. In the training process the weights w_{ij} of all the neurons are adjusted to minimize the error of network predictions [22, 30].

The above-described architecture of an ANN is presented in Fig. 2. The idea of speeding up real-time ANN computations consists in introducing parallelization. Neurons in a given layer

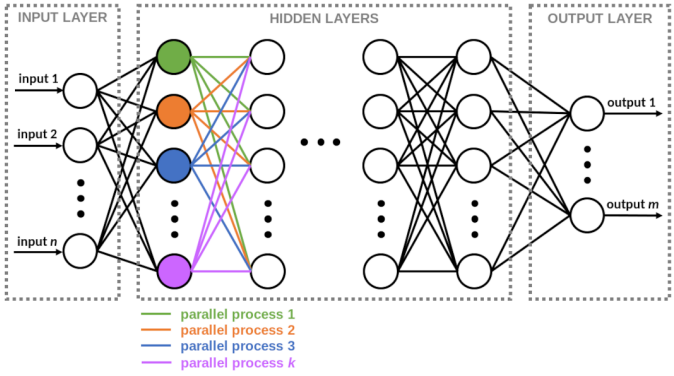
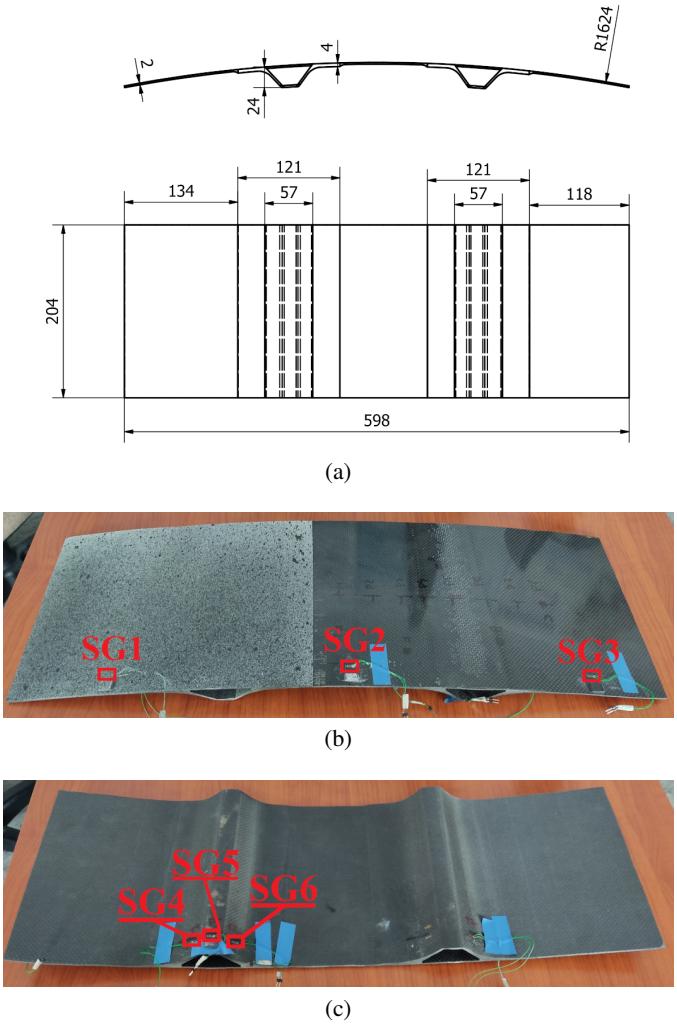


Fig. 2. Parallel processing of ANN computations

are not connected to each other and therefore can process information at the same time when all the inputs are available. For some problems, the hidden layers can have even several dozen or several hundred of neurons – too much to process at the same time by average modern CPUs. The number of threads for graphic processors is significantly higher therefore GPUs are commonly utilized for artificial intelligence processing. However GPUs are usually characterized by high power consumption and are not suitable for embedded applications. These drawbacks can be overcome by FPGAs. The power consumption of FPGAs as compared to GPUs is lower by multiple orders of magnitude. In FPGAs the number of parallel tasks that can be executed is not limited by threads, just by hardware resources on which independent circuits are defined.

In order to maximize the performance of FPGA computations and reduce the utilization of hardware resources necessary to execute the algorithm, it is important to use fixed point arithmetic in neural network processing [61]. It is a method of representing non-integer numbers in computer memory by storing them in a fixed number of bits from which the fractional part of the numbers is assigned to a portion of bits of a fixed size. The number of bits assigned to a variable is called *word length* and the number of bits assigned to its integer portion is called *integer word length*. If the variable can take positive and negative values, an extra bit is assigned to determine the sign of the number [62]. Although implementing fixed point arithmetic increases complexity of the project and development time, it has the advantages of integer arithmetic, allowing to perform operations faster by many orders of magnitude than floating point arithmetic, while maintaining some flexibility of the latter [63,64]. When uploading an ANN algorithm to FPGA, word lengths and integer word lengths should be carefully adjusted. They should be minimal but still cover the whole range of values that the variables can take and ensure the required accuracy of computations. That way the utilized resources of FPGA are minimized.



(a)

(b)

(c)

Fig. 3. Tested aerostructure: (a) dimensions, (b) photograph with top strain gauges positions, (c) photograph with bottom strain gauges positions

3. EXAMPLE

OLM process of a composite aerostructure is considered as an example of real-time parallel data processing by an artificial neural network using FPGA. The aerostructure being considered is a hat-stiffened panel, a typical geometry used for aircraft skin. Dimensions and photographs of the panel are presented in Fig. 3 with marked positions of six strain gauges (SG1-SG6) mounted to the top surface and rib of the structure.

The panel is a laminate made of 10 layers of carbon woven with different angle orientation in epoxy resin. An accurate finite element (FE) model of the structure was created using ANSYS Workbench software with ACP (Ansys Composite Prep/Post) module for advanced designing of composite structures. Successful experimental validation of the numerical model was described in [65] and [66]. In [65], bending stiffness in the linear-elastic range of the panel was measured for different loading points, using an universal testing machine, and compared with the stiffness obtained from the FE model – the obtained differences were between 0.14% and 1.24%. In [66], the composite hat-stiffened panel was loaded at different points using a universal testing machine, and strain measurements from sensors SG1-SG6 were registered together with force value from the load cell of the testing machine. Then the obtained strain-force plots were compared with the results obtained from numerical simulations using the FE model. Satisfying compliance was achieved, mean square errors were calculated and are listed for each sensor for each loading point.

The FE model is presented in Fig. 4. It consists of 50 831 Quad elements of quadratic order and of 50 492 nodes. Left edge of the panel is supported in X and Y direction and right edge – in Y direction only. It is assumed that the panel can be loaded at any point on the top surface. In order to generate enough reference data to train the neural network, 403 small round surfaces were insulated on the top surface, where forces of values 1, 2, 3, . . . , 40 N were applied subsequently at each location. In the locations of the strain gauges, rectangle-shaped surfaces were insulated of the size corresponding to the active part of the sensors, in order to simulate the measurements as average strain values.

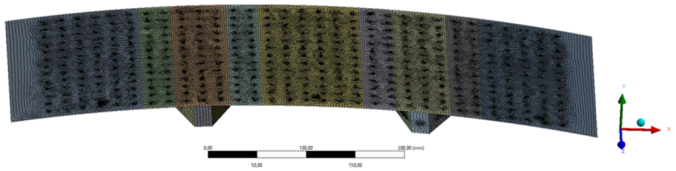


Fig. 4. Finite element model of the structure

3.1. Artificial neural networks

ANN has got six inputs which are strain measurements from strain gauges SG1 – SG6 in [µm/m] and one output that is the Tsai-Wu failure criterion [67] of the whole structure under current load, as illustrated in Fig. 5. It applies to anisotropic composite materials with different strengths in tension and compression. Failure is predicted when the criterion value exceeds 1:

$$F_j \sigma_j + F_{jk} \sigma_j \sigma_k \leq 1, \tag{2}$$

where $j, k = 1, 2, \ldots, 6$, σ_j are stresses in Voight notation, F_i are parameters determined from stress limits and F_{jk} are coupling coefficients. All the coupling coefficients were assumed as -1 and the stress limits are presented in Table 1.

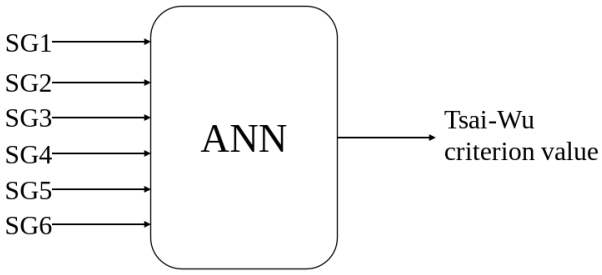


Fig. 5. Inputs and output of ANN

Table 1
Stress limits of the laminate

Load direction	Stress limit, MPa
Tensile 1	200
Tensile 2	200
Tensile 3	12.5
Compression 1	–125
Compression 2	–125
Compression 3	–42.5
Shear 12	30
Shear 23	16
Shear 13	16

The relation between the output and the inputs to ANN is highly nonlinear. To illustrate that, a constant force value of -10 N in the Y direction was assumed and relations between the force location and the Tsai-Wu criterion value and strain measurements are presented in Figs. 6 and 7, respectively.

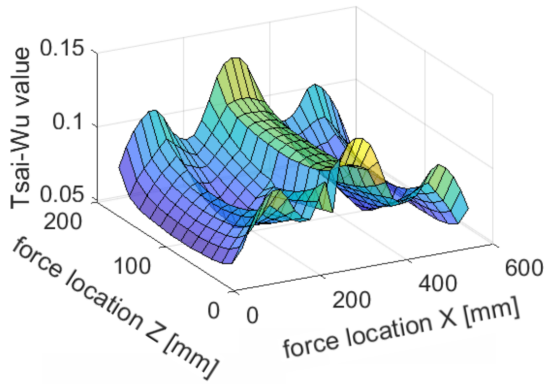
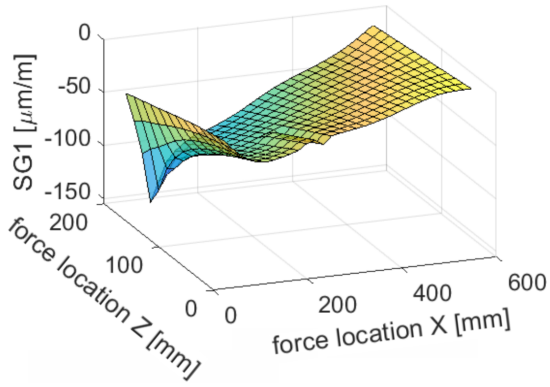
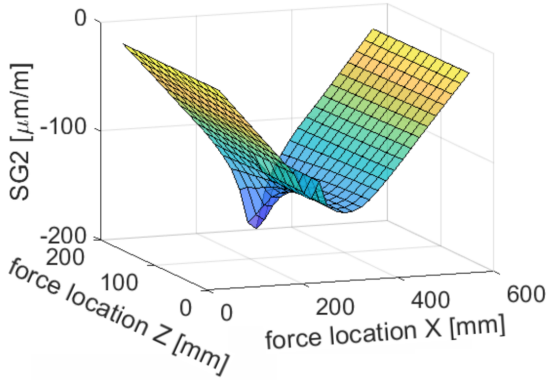


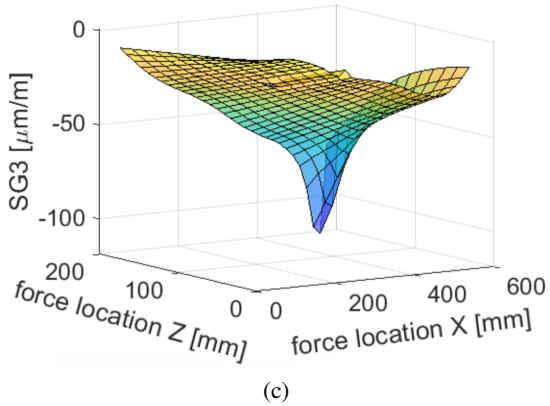
Fig. 6. Tsai-Wu criterion value for different application locations of constant force



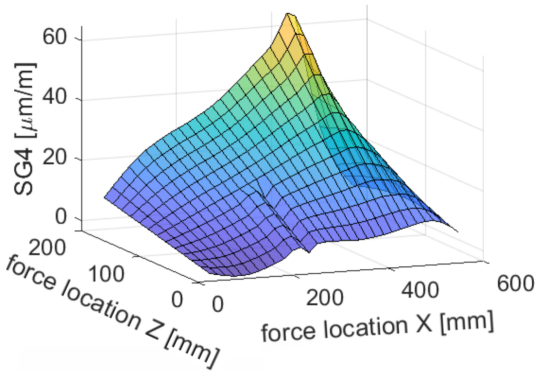
(a)



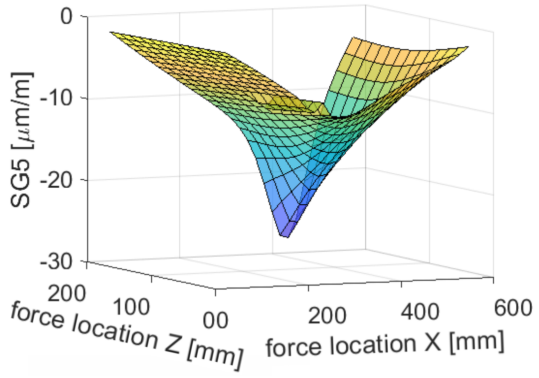
(b)



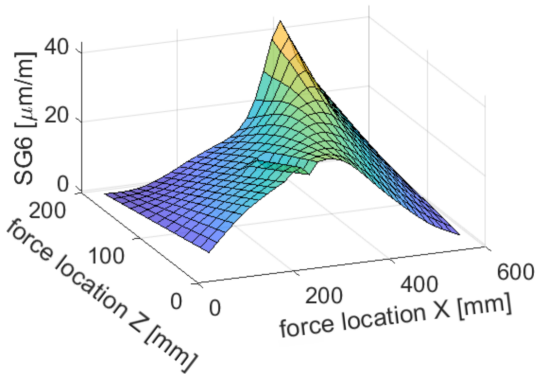
(c)



(d)



(e)



(f)

Fig. 7. Simulated strain gauge measurements for different application locations of constant force: (a) strain gauge SG1, (b) strain gauge SG2, (c) strain gauge SG3, (d) strain gauge SG4, (e) strain gauge SG5, (f) strain gauge SG6

As the network is trained on computationally generated data, to be ultimately working on measured data from strain gauges, white Gaussian noise was added to the reference data from FE simulations. As for data generation, 403 force locations and 40 force values were used, and 16 120 input-output reference pairs were obtained. Random 70% of the data was used for training, 15% for validation and 15% for test. The Levenberg-Marquardt algorithm was used for training.

ANN with one hidden layer, with nine neurons in the hidden layer was found. The activation function of the hidden neurons is sigmoid, equation (3). Training performance of the network is presented in Fig. 8. The best validation performance (MSE) $1.65 \cdot 10^{-4}$ was achieved at epoch 198.

$$\varphi(e) = \frac{2}{1+\exp(-2e)} - 1. \tag{3}$$

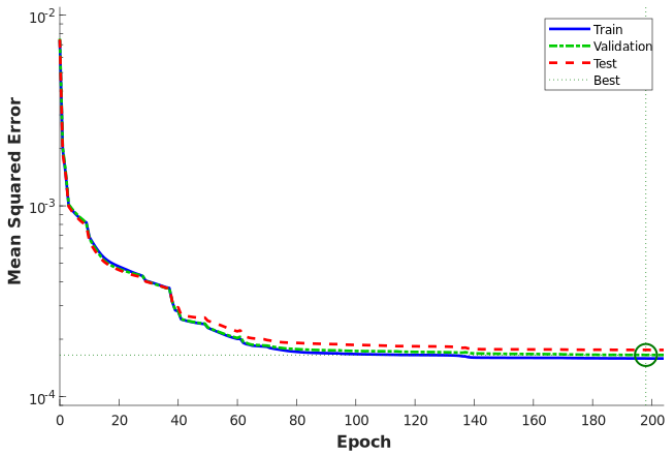


Fig. 8. Training performance of ANN

3.2. FPGA implementation and benchmarking

The algorithm, based on the trained ANN, described in detail in Appendix A, was implemented to the real-time National Instruments cRIO-9054 microcontroller. The microcontroller is equipped with 2-core CPU Intel Atom E3805 and Xilinx Artix-7 A100T FPGA. Extension modules are connected to the cRIO system, and include NI-9235 (8-channel 120 Ω quarter-bridge strain gauge input module) and NI-9201 (8-channel, 12-bit, ±10 V voltage input module). The microcontroller was programmed using National Instruments LabVIEW 20.0.1 software. The thermal design power of the CPU is 3 W and of the FPGA is 0.142 W.

The ANN prediction model was implemented as a function that reads signals from strain gauges and returns predictions of the Tsai-Wu criterion value. For benchmarking purposes, two versions of this function were created. In the first version, all ANN computations were performed using the CPU microcontroller and floating point arithmetic. In the second version, ANN was implemented to the FPGA and fixed-point arithmetic was utilized. In order to adjust the size of fixed-point variables in the ANN function, MATLAB toolbox Fixed-Point Designer™ was utilized. This tool allows to optimize fixed-point algorithms before implementing the logic to embedded targets. It also allows to debug quantization effects like overflows and precision loss. The algorithm was optimized taking into account achieving desired accuracy and minimizing FPGA hardware resources utilization. The maximum value ranges of each variable were defined based on computationally generated reference data. The sizes of all fixed-point variables are listed in Table 2. The sizes concern every element of a given vector / matrix. Moreover, operation (A.6) (the sigmoid activation function of the hidden neurons) was substituted by a lookup table. After careful numerical studies, it has turned out that the input values $h2_i$ to the

Table 2
Optimized parameters of fixed-point variables

Variable	Word length (bits)	Integer word length (bits)	Signed (additional bit)
x	16	11	Yes
offset1	16	11	Yes
i1	16	10	No
gain1	16	–5	No
i2	16	2	No
ymin1	1	1	Yes
i3	16	2	Yes
IW1	16	6	Yes
h1	16	6	Yes
b1	16	4	Yes
h2	16	5	Yes
h3	16	1	Yes
LW2	16	4	Yes
o1	16	5	Yes
b2	16	3	No
o2	16	1	Yes
ymin2	1	1	Yes
o3	16	1	Yes
gain2	16	2	No
o4	16	–1	Yes
offset2	16	–7	No
y	16	–1	No

activation function are always within a small range between –1.5 and 2 (Fig. 9). To speed up the computations and maintain high accuracy, a lookup table with 1024 uniformly distributed values, without interpolation, was implemented. Processing of all the hidden neurons during execution of the function is achieved in true parallel mode as in FPGA they constitute separate circuits. The ANN function utilized 42% of the available arithmetic-logic DSP48 blocks of the FPGA hardware resources.

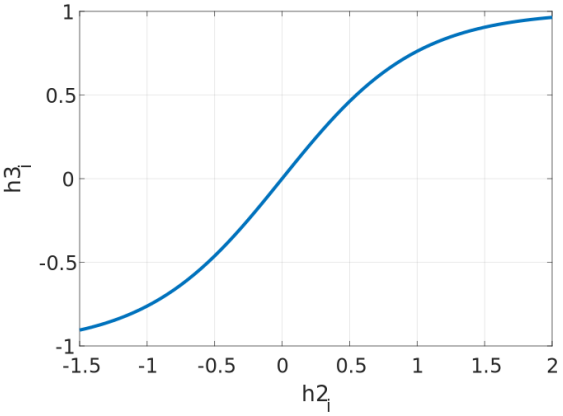


Fig. 9. Sigmoid activation function replaced by lookup table

For benchmarking, both versions of the algorithm (ANN computed in CPU and FPGA) were tested using 10 000 samples of previously taken example measurements (from strain gauges SG1-SG6). The inputs were stored in the memory of the microcontroller as a measurement file. The goal was to perform all the ANN computations as quickly as possible for all the data samples in a loop, without any constant time step and waiting functions. Time of computations was measured as an average from 10 runs. First version of the ANN function (CPU computations, floating point arithmetic) took an average of 1 983 ms to complete the computations with standard deviation of 47.7 ms. In the second version of the ANN function (FPGA computations, fixed point arithmetic), subsequent data samples were read from the measurement file by the CPU, converted to a fixed point number, and sent to the FPGA with a binary flag turned on that new data has arrived. The FPGA algorithm processed the input data to output by ANN and sent interruption signal to CPU when the result was ready to be read. It took 594 ms to complete all the calculations as an average from 10 runs with 16.1 ms as standard deviation. By performing true parallel computations using FPGA, about 70% of computational time reduction was achieved.

3.3. Real-time experimental testing of FPGA-based algorithm

Series of experiments were performed where the FPGA computations were performed in real time and ANN was predicting the Tsai-Wu criterion for the tested structure under load. To load the structure at different points, a universal MTS Insight 10 testing machine, equipped with 10 kN load cell, was utilized. The composite panel was mounted in the testing machine using a specially manufactured steel frame of high stiffness. The real-time microcontroller cRIO-9054 was connected to the testing machine by ±10 V analog signal (load information) and to the six strain gauges of the structure. The microcontroller had two roles: 1) it acted as a DAQ system for the force and strain signals, 2) it processed the measured strain information by ANN to predict the Tsai-Wu criterion value in real-time using an FPGA-based parallel algorithm. Multicore approach was applied to the CPU where the first core was executing deterministic tasks (reading signals and getting Tsai-Wu value predictions from FPGA every constant time step) and the second core was executing non-deterministic tasks (sending measured and predicted values to the PC to be displayed to the user). The time step for deterministic tasks was 5 ms (Fig. 10).

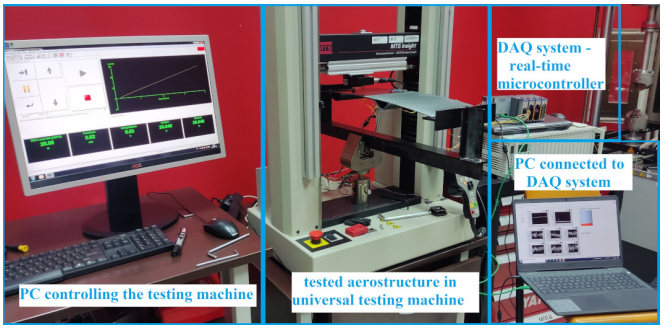


Fig. 10. Experimental setup

During experiments, the tested aerostructure was loaded at four random points P1–P4, not corresponding to any of the 403 sample loading locations (Fig. 4) that were used to generate the reference data to train ANN. The loading locations P1–P4 are presented in Fig. 11. The testing machine was loading the structure at constant speed of 10 mm/min in the range from 0 to 40 N.

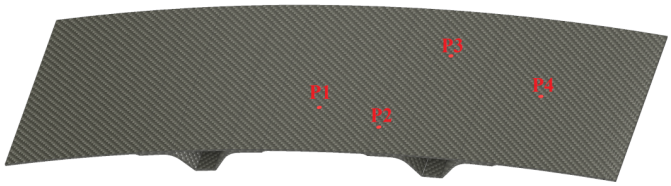


Fig. 11. Loading points P1–P4 locations

The predictions of the Tsai-Wu value by the FPGA-based parallel algorithm with fixed point arithmetic, made in real time during the experiments, were compared against exact values and predictions of ANN using numerically generated strain values as input data, and presented in Fig. 12. Numerically simulated values of the Tsai-Wu criterion using the finite element model and known (measured during experiments) values of applied loads are here considered as exact values. As one can see in Fig. 12, all three plots are compatible in each of the four cases.

The predictions made during real-time experiments were compared to ANN predictions using the more precise floating point version of the algorithm, executed on CPU, and using the acquired measurement data from the experiments. Figure 13 presents this comparison. The differences between both predictions seem to be relatively small. To quantify the accuracy of the predictions, mean squared errors were calculated for every loading point and listed in Table 3. The predictions using the fast parallel FPGA version of the algorithm are slightly less accurate, and for points P1–P3 the MSE is one order of magnitude higher than for the floating point version. However, accuracy of the former version is still high, resulting in MSE of 10^{-4} each time. The MSE between fixed point and floating point representation of ANN is substantially lower, usually of 10^{-6}. The MSEs are slightly greater for point P4, as the stiffness in that point is higher than for the other three, resulting in smaller values of strain measurements, and, therefore, making the measured data relatively noisier. The resulting predictions for P4 are also of greater noise, as observed in Fig. 12 and 13. The predictions are at an acceptable level however, while this inconvenience could be decreased by applying data filtering to the measured data.

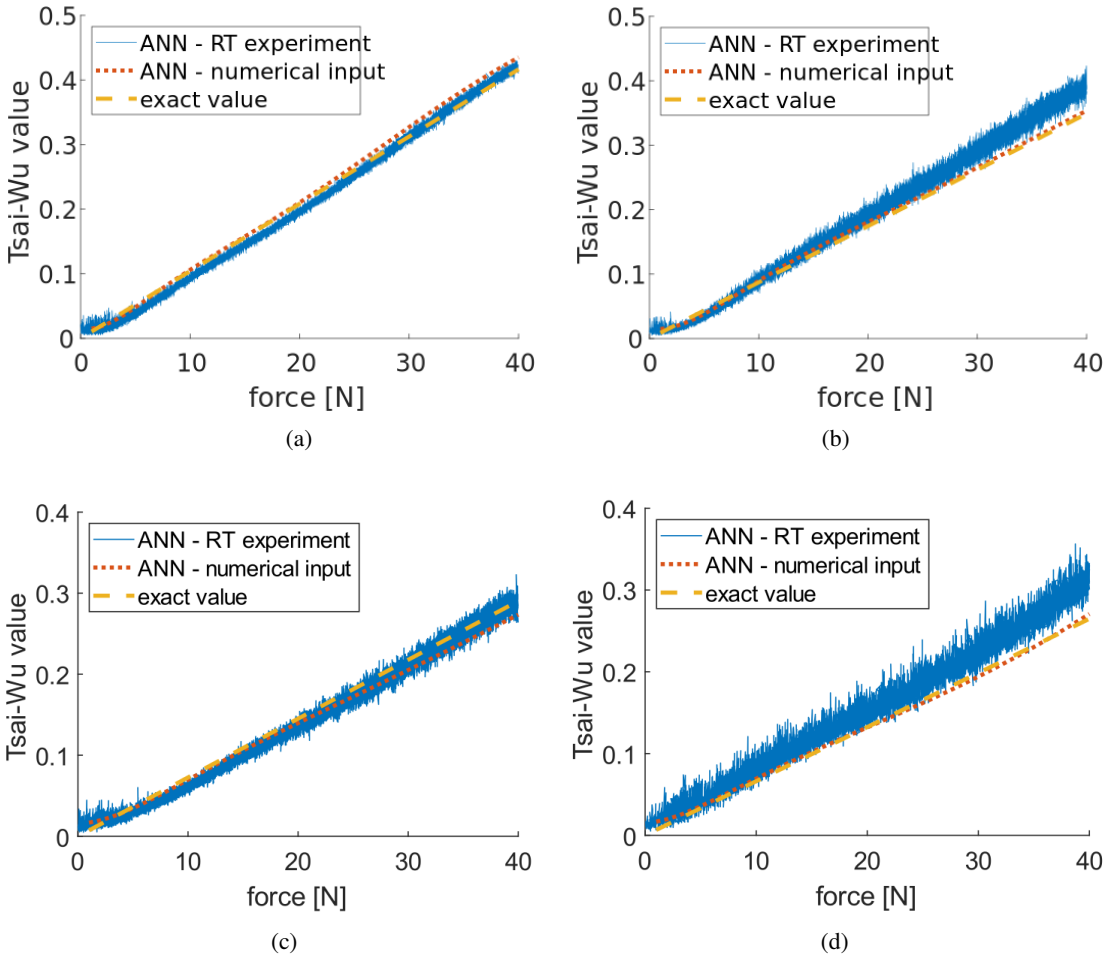


Fig. 12. Tsai-Wu value predictions obtained by ANN during real-time experiments compared against ANN predictions from numerically generated inputs and exact values: (a) loading point P1, (b) loading point P2, (c) loading point P3, (d) loading point P4

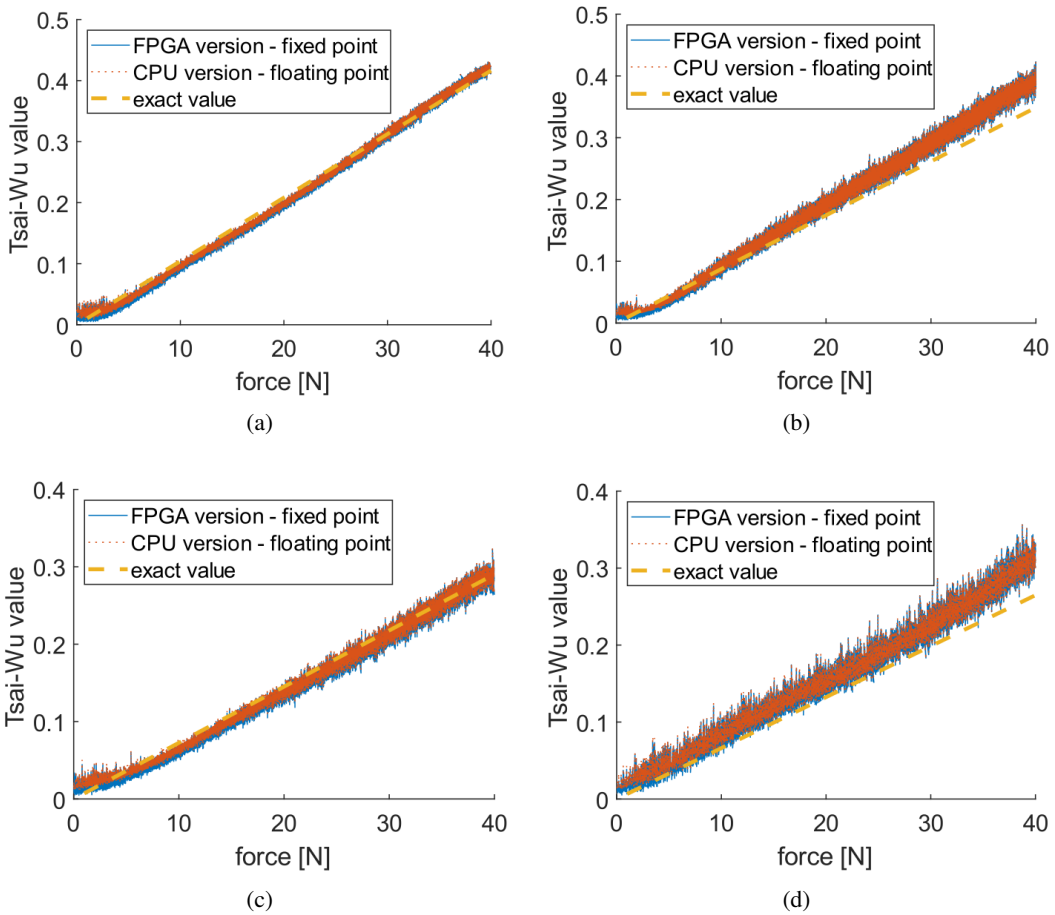


Fig. 13. Comparison between Tsai-Wu value predictions from FPGA version of the ANN algorithm (with fixed point arithmetic) and CPU version (with floating point arithmetic): (a) loading point P1, (b) loading point P2, (c) loading point P3, (d) loading point P4

Table 3

Comparison of mean squared errors (MSE) of Tsai-Wu value predictions for CPU (floating point) and FPGA (fixed point) versions of the ANN algorithm

Loading point	Mean squared error (MSE) of Tsai-Wu value predictions		
	floating point algorithm – exact	fixed point algorithm – exact	floating point algorithm – fixed point algorithm
P1	$7.6934 \cdot 10^{-5}$	$1.0421 \cdot 10^{-4}$	$6.1452 \cdot 10^{-6}$
P2	$6.1998 \cdot 10^{-5}$	$5.9273 \cdot 10^{-4}$	$3.9859 \cdot 10^{-6}$
P3	$7.5060 \cdot 10^{-5}$	$1.0347 \cdot 10^{-4}$	$9.9347 \cdot 10^{-6}$
P4	$7.8658 \cdot 10^{-4}$	$6.9858 \cdot 10^{-4}$	$1.0212 \cdot 10^{-5}$

4. CONCLUSIONS

Green computing, understood as using energy-efficient computing devices, for different applications, is becoming more important than ever. The main focus of the presented research was to move OLM processes towards green computing by utilizing FPGAs in embedded applications. Such applications are natural for sensor-based monitoring processes, as all the necessary operations can be performed in the computing system connected to the sensors. FPGAs are characterized by exceptionally low power consumption and high efficiency at the expense of lower flexibility, as compared to microprocessors. Moreover, FPGAs allow for true parallel processing of data, as there is no need for the operating system to execute the logic, and the number of parallel circuits is only limited to the hardware resources. ANN calculations can be done in a highly parallel mode as all neurons of a given layer can be processed at the same time. Therefore, deploying ANNs to FPGA has the advantages of potentially more parallel threads than in a standard CPU and much lower consumption than for a GPU.

The presented example has proven the abovementioned assumptions. Adjusting the ANN algorithm to FPGA requirements took relatively much effort as all the arithmetic was converted to a fixed point and a lookup table was created in place of the sigmoid activation function. However, benchmarking revealed that this allowed to reduce the time of computations by 70% as compared to standard CPU execution. It is expected that for ANNs with a greater number of hidden neurons and/or a greater number of hidden layers the relative computational time reduction would be even more substantial. The FPGA utilized would not allow to deploy a significantly greater network, as the described one consumed 42% of the available arithmetic-logic DSP48 blocks.

Optimizing the FPGA algorithm is extremely important as the hardware resources are limited and within those resources the required accuracy of computations must be maintained. For OLM processes, having an exact, experimentally verified, finite element model is helpful as it allows to automatically run simulations for every possible load case, without the necessity of performing tedious series of experiments. The possibility to simulate the reference data allowed not only to generate a training dataset for ANN, but also to determine the maximal ranges of all algorithm variables for fixed point conversion. Adding white Gaussian noise to the simulated reference input-output data helps to train ANN to become more resistant to noisy measurements and adds a necessary reserve to the variable ranges.

As for the presented example, the obtained results of OLM can be considered successful. The trained ANN worked correctly in real time, with experimentally acquired measurements from strain gauges, although it was trained only on simulated data. Implementing data filtering to the measurements could reduce the obtained noise of the predictions. However, the noise is already at a relatively low level.

The presented results of high throughput, true parallel and energy efficient FPGA computations of ANN algorithms show the possibilities of expanding such an approach to other SHM applications (not only OLM), as well as other embedded applications of ANNs.

The developed methods and obtained results constitute a starting point for future research on introducing green computing to embedded applications. In the presented example the whole ANN, after conversion to FPGA requirements, was programmed in LabVIEW by manually adding every operation. The next step is to create a programming tool that will allow to automate this process. Also, measuring and comparing energy consumption of FPGA and CPU devices running AI-based OLM algorithms is considered a task for the future.

APPENDIX A: ARTIFICIAL NEURAL NETWORK ALGORITHM

The considered ANN has got one hidden layer with neurons of the sigmoid activation function. The number of inputs $n\_\text{in} = 6$, the number of outputs $n\_\text{out} = 1$ and the number of hidden neurons $N = 9$. In the following description, equations contain mostly single operations, for better understanding of conversion to fixed point representation in FPGA implementation.

In the input layer, the input values (vector x of size $n\_\text{in} \times 1$) are normalized, using three sets of coefficients: **offset1**, **gain1** and **ymin1**, of sizes $n\_\text{in} \times 1$. Subtraction (A.1), Hadamard product (A.2), and addition (A.3) are performed.

$$\mathbf{i1} = \mathbf{x} - \mathbf{offset1}, \tag{A.1}$$

$$\mathbf{i2} = (\mathbf{i1} \odot \mathbf{gain1}), \tag{A.2}$$

$$\mathbf{i3} = \mathbf{i2} + \mathbf{ymin1}. \tag{A.3}$$

In the hidden layer, two constant matrices are used, **IW1** (of size $N \times n\_\text{in}$) and **b1** (of size $(N \times 1)$) to determine the input to the activation function. Matrix operations (A.4) and (A.5) are performed. For every i-th element $\mathbf{h2}_i$ of matrix **h2**, the sigmoid activation function (A.6) is executed, where vector **h3** is obtained as a result, of size $(N \times 1)$.

$$\mathbf{h1} = \mathbf{IW1} \cdot \mathbf{i3}, \tag{A.4}$$

$$\mathbf{h2} = \mathbf{h1} + \mathbf{b1}, \tag{A.5}$$

$$\mathbf{h3}_i = \frac{2}{1 + \exp(-2 \cdot h2_i)} - 1. \tag{A.6}$$

In the output layer, two constant matrices are used, **LW2** (of size $n\_\text{out} \times \text{N}$) and **b2** (of size $(n\_\text{out} \times 1)$) to determine the input to the linear activation function. For this purpose, operations (A.7) and (A.8) are performed. The activation function of the output layer and scaling is performed by operations (A.9)–(A.11), whereas in (A.10) Hadamard division is executed. Vectors **b2**, **ymin2**, **gain2** and **offset2** are all of size $n\_\text{out} \times 1$. Output vector **y** from ANN is obtained.

$$\mathbf{o1} = \mathbf{LW2} \cdot \mathbf{h3}, \tag{A.7}$$

$$\mathbf{o2} = \mathbf{o1} + \mathbf{b2}, \tag{A.8}$$

$$\mathbf{o3} = \mathbf{o2} - \mathbf{ymin2}, \tag{A.9}$$

$$\mathbf{o4} = \mathbf{o3} \oslash \mathbf{gain2}, \tag{A.10}$$

$$\mathbf{y} = \mathbf{o4} + \mathbf{offset2}. \tag{A.11}$$

Values of the constant matrices in the discussed example are as follows:

$$\mathbf{offset1} = \begin{bmatrix} -620.3352 \\ -789.2278 \\ -475.8088 \\ -13.6552 \\ -111.6687 \\ -2.8206 \end{bmatrix},$$

$$\mathbf{gain1} = \begin{bmatrix} 0.00320785737291781 \\ 0.00253863543428045 \\ 0.00414908874413595 \\ 0.00731514312766509 \\ 0.0178318864891182 \\ 0.0113452043988117 \end{bmatrix},$$

$$\mathbf{ymin1} = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix},$$

$$\mathbf{IW1} = \begin{bmatrix} -28.887775 & 6.1646665 & 16.857494 \\ 5.387242 & -0.714431 & -1.456031 \\ 1.860774 & -0.329440 & 0.311148 \\ -3.715395 & 4.398884 & 0.854155 \\ -1.197236 & -0.499655 & -0.609615 \\ -3.666818 & 4.239059 & 0.399583 \\ -0.051716 & -0.557181 & 0.128190 \\ 5.717952 & -1.031718 & -1.658909 \\ 1.805440 & 0.077197 & 1.592871 \end{bmatrix}$$

$$\begin{bmatrix} -5.565801 & 0.217765 & 7.243180 \\ 1.615390 & 1.654311 & -0.647806 \\ -0.324647 & 0.866814 & 0.454322 \\ -2.053143 & 0.009653 & 2.674085 \\ -1.456445 & -1.194856 & -0.322091 \\ -2.244133 & -0.175049 & 2.435752 \\ -2.145632 & -1.307685 & 0.539131 \\ 1.601653 & 1.585608 & -0.920229 \\ 2.959694 & 0.391987 & 1.104751 \end{bmatrix},$$

$$\mathbf{b1} = \begin{bmatrix} 7.160749 \\ -4.246967 \\ -3.474163 \\ -0.996786 \\ 0.751525 \\ -0.800328 \\ 1.927986 \\ -4.427155 \\ 0.719475 \end{bmatrix}, \quad \mathbf{LW2} = \begin{bmatrix} 0.070656 \\ -6.073142 \\ 3.761795 \\ -4.797865 \\ 1.160196 \\ 5.580726 \\ -4.209480 \\ 6.362346 \\ 1.510866 \end{bmatrix}^{\mathrm{T}},$$

$$\mathbf{b2} = \begin{bmatrix} 7.490504 \end{bmatrix}, \quad \mathbf{ymin2} = \begin{bmatrix} -1 \end{bmatrix},$$

$$\mathbf{gain2} = \begin{bmatrix} 3.780953 \end{bmatrix}, \quad \mathbf{b2} = \begin{bmatrix} 0.004871 \end{bmatrix}.$$

ACKNOWLEDGEMENTS

The research was partially financed from the statutory subsidy of the Faculty of Mechanical Engineering, Silesian University of Technology.

The research was partially financed by the National Science Centre, Poland, under grant DEC-2019/03/X/ST8/00405 (*Application of high-throughput parallel real-time computations using FPGAs and AI techniques*).

The author would like to express his gratitude to his grandfather Mr. Ewald Ulfig for many hours of his work on manufacturing parts for the experimental setup.

REFERENCES

[1] F.M. Reis, P.F. da Costa Antunes, N.M. Mendes Maia, A.R. Carvalho, and P.S. de Brito André, "Structural Health Monitoring Suitable for Airborne Components Using the Speckle Pattern in Plastic Optical Fibers," *IEEE Sens. J.*, vol. 17, no. 15, pp. 4791–4796, 2017, doi: 10.1109/JSEN.2017.2715258.

[2] S. Willis, "Olm: A Hands-on Approach," in *ICAF 2009, Bridging the Gap between Theory and Operational Practice*, M.J. Bos, Ed., Dordrecht: Springer Netherlands, 2009, pp. 1199–1214.

[3] N. Aldridge, P. Foote, and I. Read, "Operational Load Monitoring for Aircraft & Maritime Applications," *Strain*, vol. 36, no. 3, pp. 123–126, 2000, doi: 10.1111/j.1475-1305.2000.tb01187.x.

[4] L. Colombo, C. Sbarufatti, W. Zielinski, K. Dragan, and M. Giglio, "Numerical and experimental flight verifications of a calibration matrix approach for load monitoring and temperature reconstruction and compensation," *Aerosp. Sci. Technol.*, vol. 118, p. 107074, Nov. 2021, doi: 10.1016/j.ast.2021.107074.

[5] H. Guo, G. Xiao, N. Mrad, and J. Yao, "Fiber Optic Sensors for Structural Health Monitoring of Air Platforms," *Sensors*, vol. 11, no. 4, pp. 3687–3705, 2011, doi: 10.3390/s110403687.

[6] J. Serafini, G. Bernardini, R. Porcelli, and P. Masarati, "In-flight health monitoring of helicopter blades via differential analysis," *Aerosp. Sci. Technol.*, vol. 88, pp. 436–443, May 2019, doi: 10.1016/j.ast.2019.03.039.

[7] F. Terroba, M. Frövel, and R. Atienza, "Structural health and usage monitoring of an unmanned turbojet target drone," *Struct. Health Monit.*, vol. 18, no. 2, pp. 635–650, Mar. 2019, doi: 10.1177/1475921718764082.

[8] L. Li, "Damage development and lifetime prediction of fiber-reinforced ceramic-matrix composites subjected to cyclic loading at 1300°C in vacuum, inert and oxidative atmospheres," *Aerosp. Sci. Technol.*, vol. 86, pp. 613–629, Mar. 2019, doi: 10.1016/j.ast.2019.01.060.

[9] F. Fang, L. Qiu, S. Yuan, and Y. Ren, "Dynamic probability modeling-based aircraft structural health monitoring framework under time-varying conditions: Validation in an in-flight test simulated on ground," *Aerosp. Sci. Technol.*, vol. 95, p. 105467, Dec. 2019, doi: 10.1016/j.ast.2019.105467.

[10] S.R. Hunt and I.G. Hebden, "Validation of the Eurofighter Typhoon structural health and usage monitoring system," *Smart Mater Struct.*, vol. 10, no. 3. pp. 497–503, 2001. doi: 10.1088/0964-1726/10/3/311.

[11] H. Rocha, C. Semprimoschnig, and J.P. Nunes, "Sensors for process and structural health monitoring of aerospace composites: A review," *Eng. Struct.*, vol. 237, p. 112231, 2021, doi: 10.1016/j.engstruct.2021.112231.

[12] Z. Ma and X. Chen, "Fiber Bragg Gratings Sensors for Aircraft Wing Shape Measurement: Recent Applications and Technical Analysis," *Sensors*, vol. 19, no. 1, p. 55, 2019, doi: 10.3390/s19010055.

[13] M.J. Nicolas, R.W. Sullivan, and W.L. Richards, "Large Scale Applications Using FBG Sensors: Determination of In-Flight Loads and Shape of a Composite Aircraft Wing," *Aerospace*, vol. 3, no. 3, p. 18, 2016, doi: 10.3390/aerospace3030018.

[14] A. Kakei and J.A. Epaarachchi, "Use of fiber Bragg grating sensors for monitoring delamination damage propagation in glass-fiber reinforced composite structures," *Front. Optoelectron.*, vol. 11, no. 1, pp. 60–68, Mar. 2018, doi: 10.1007/s12200-018-0761-9.

[15] D. Anastasopoulos, G. De Roeck, and E.P.B. Reynders, "One-year operational modal analysis of a steel bridge from high-resolution macrostrain monitoring: Influence of temperature vs. retrofitting," *Mech. Syst. Signal Process.*, vol. 161, p. 107951, Dec. 2021, doi: 10.1016/j.ymssp.2021.107951.

[16] D.J. Robert, D. Chan, P. Rajeev, and J. Kodikara, "Effects of operational loads on buried water pipes using field tests," *Tunn. Undergr. Space Technol.*, vol. 124, p. 104463, Jun. 2022, doi: 10.1016/j.tust.2022.104463.

[17] K. Schroeder, W. Ecke, J. Apitz, E. Lembke, and G. Lenschow, "A fibre Bragg grating sensor system monitors operational load in a wind turbine rotor blade," *Meas. Sci. Technol.*, vol. 17, no. 5, p. 1167, 2006.

[18] P. Krot, P. Sliwinski, R. Zimroz, and N. Gomolla, "The identification of operational cycles in the monitoring systems of underground vehicles," *Measurement*, vol. 151, p. 107111, Feb. 2020, doi: 10.1016/j.measurement.2019.107111.

[19] M. Gordan *et al.*, "State-of-the-art review on advancements of data mining in structural health monitoring," *Measurement*, vol. 193, p. 110939, Apr. 2022, doi: 10.1016/j.measurement.2022.110939.

[20] O. Ahmed, X. Wang, M.-V. Tran, and M.-Z. Ismadi, "Advancements in fiber-reinforced polymer composite materials damage detection methods: Towards achieving energy-efficient SHM systems," *Compos. Part B Eng.*, vol. 223, p. 109136, Oct. 2021, doi: 10.1016/j.compositesb.2021.109136.

[21] S.K. Baduge *et al.*, "Artificial intelligence and smart vision for building and construction 4.0: Machine and deep learning methods and applications," *Autom. Constr.*, vol. 141, p. 104440, Sep. 2022, doi: 10.1016/j.autcon.2022.104440.

[22] D.J. Livingstone, "Artificial Neural Networks: Methods and Applications," in *Methods in Molecular Biology*. Humana Press, 2011. [Online]. Available: https://books.google.pl/books?id=eVocYgEACAAJ

[23] I. Tabian, H. Fu, and Z. Sharif Khodaei, "A Convolutional Neural Network for Impact Detection and Characterization of Complex Composite Structures," *Sensors*, vol. 19, no. 22, p. 4933, 2019, doi: 10.3390/s19224933.

[24] A.M. Damm, C. Spitzmüller, A.T.S. Raichle, A. Bühler, P. Weißgraeber, and P. Middendorf, "Deep learning for impact detection in composite plates with sparsely integrated sensors," *Smart Mater. Struct.*, vol. 29, no. 12, p. 125014, Oct. 2020, doi: 10.1088/1361-665X/abb644.

[25] K.-C. Jung and S.-H. Chang, "Advanced deep learning model-based impact characterization method for composite laminates," *Compos. Sci. Technol.*, vol. 207, p. 108713, May 2021, doi: 10.1016/j.compscitech.2021.108713.

[26] A. Khan, D.-K. Ko, S.C. Lim, and H.S. Kim, "Structural vibration-based classification and prediction of delamination in smart composite laminates using deep learning neural network," *Compos. Part B Eng.*, vol. 161, pp. 586–594, Mar. 2019, doi: 10.1016/j.compositesb.2018.12.118.

[27] M.-H. Yu and H.-S. Kim, "Deep-learning based damage sensing of carbon fiber/polypropylene composite via addressable conducting network," *Compos. Struct.*, vol. 267, p. 113871, Jul. 2021, doi: 10.1016/j.compstruct.2021.113871.

[28] A. Datta, M.J. Augustin, N. Gupta, S.R. Viswamurthy, K.M. Gaddikeri, and R. Sundaram, "Impact Localization and Severity Estimation on Composite Structure Using Fiber Bragg Grating Sensors by Least Square Support Vector Regression," *IEEE Sens. J.*, vol. 19, no. 12, pp. 4463–4470, 2019, doi: 10.1109/JSEN.2019.2901453.

[29] A. Mardanshahi, V. Nasir, S. Kazemirad, and M.M. Shokrieh, "Detection and classification of matrix cracking in laminated composites using guided wave propagation and artificial neural networks," *Compos. Struct.*, vol. 246, p. 112403, Aug. 2020, doi: 10.1016/j.compstruct.2020.112403.

[30] W. Mucha, "Comparison of Machine Learning Algorithms for Structure State Prediction in Operational Load Monitoring," *Sensors*, vol. 20, no. 24, p. 87, 2020, doi: 10.3390/s20247087.

[31] W. Mucha, W. Kuś, J.C. Viana, and J.P. Nunes, "Operational Load Monitoring of a Composite Panel Using Artificial Neural Networks," *Sensors*, vol. 20, no. 9, p. 2534, 2020, doi: 10.3390/s20092534.

[32] M.J. Candon, O. Levinski, A. Altaf, R. Carrese, and P. Marzocca, "Aircraft Transonic Buffet Load Prediction using Artificial Neural Networks," in *AIAA Scitech 2019 Forum*, 2019, doi: 10.2514/6.2019-0763.

[33] D. Wada, Y. Sugimoto, H. Murayama, H. Igawa, and T. Nakamura, "Investigation of Inverse Analysis and Neural Network Approaches for Identifying Distributed Load using Distributed Strains," *Trans. Jpn. Soc. Aeronaut. SPACE Sci.*, vol. 62, no. 3, pp. 151–161, 2019, doi: 10.2322/tjsass.62.151.

[34] M. Kuss and C.E. Rasmussen, "Gaussian processes in reinforcement learning," in *Advances in neural information processing systems*, 2004, pp. 751–758.

[35] G. Holmes, P. Sartor, S. Reed, P. Southern, K. Worden, and E. Cross, "Prediction of landing gear loads using machine learning techniques," *Struct. Health Monit.*, vol. 15, no. 5, pp. 568–582, Sep. 2016, doi: 10.1177/1475921716651809.

[36] R. Fuentes, E. Cross, A. Halfpenny, K. Worden, and R.J. Barthorpe, "Aircraft parametric structural load monitoring using Gaussian process regression," in *7th European Workshop on Structural Health Monitoring, EWSHM 2014 – 2nd European Conference of the Prognostics and Health Management (PHM) Society*, 2014, pp. 1933–1940. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84939438509&partnerID=40&md5=8609fbbf7c6af0e6ffd49784da56ac53

[37] A. Chandani, S. Wagholikar, and O. Prakash, "A Bibliometric Analysis of Green Computing," in *Information and Communication Technology for Competitive Strategies (ICTCS 2021)*, A. Joshi, M. Mahmud, and R.G. Ragel, Eds., Singapore: Springer Nature Singapore, 2023, pp. 547–557.

[38] A.O. Ojo, M. Raman, and A.G. Downe, "Toward green computing practices: A Malaysian study of green belief and attitude among Information Technology professionals," *J. Clean. Prod.*, vol. 224, pp. 246–255, Jul. 2019, doi: 10.1016/j.jclepro.2019.03.237.

[39] C. Maxfield, *The Design Warrior's Guide to FPGAs: Devices, Tools and Flows*. Elsevier Science, 2004. [Online]. Available: https://books.google.pl/books?id=dnuwr2xOFpUC

[40] S. Tyurin, "Green Logic: Green LUT FPGA Concepts, Models and Evaluations," in *Green IT Engineering: Components, Networks and Systems Implementation*, V. Kharchenko, Y. Kondratenko, and J. Kacprzyk, Eds., Cham: Springer International Publishing, 2017, pp. 241–261. doi: 10.1007/978-3-319-55595-9_12.

[41] O. Yanovskaya, M. Yanovsky, and V. Kharchenko, "The concept of green Cloud infrastructure based on distributed computing and hardware accelerator within FPGA as a Service," in *Proceedings of IEEE East-West Design & Test Symposium (EWDTS 2014)*, 2014, pp. 1–4. doi: 10.1109/EWDTS.2014.7027089.

[42] A. Shrivastava *et al.*, "VLSI Implementation of Green Computing Control Unit on Zynq FPGA for Green Communication," *Wirel. Commun. Mob. Comput.*, vol. 2021, p. 4655400, Nov. 2021, doi: 10.1155/2021/4655400.

[43] D. Ratter, "FPGAs on Mars," *Xcell Journal*, 2004, [Online]. Available: http://dea.unsj.edu.ar/sda/FPGA_On_Mars.pdf

[44] "FPGA processors keep Mars Rovers moving." [Online]. Available: https://www.militaryaerospace.com/articles/2005/01/ fpga-processors-keep-mars-rovers-moving.html (Accessed: Nov. 26, 2018).

[45] R. Wu, X. Guo, J. Du, and J. Li, "Accelerating Neural Network Inference on FPGA-Based Platforms—A Survey," *Electronics*, vol. 10, no. 9, p. 1025, 2021, doi: 10.3390/electronics10091025.

[46] V. Shymkovych, S. Telenyk, and P. Kravets, "Hardware implementation of radial-basis neural networks with Gaussian activation functions on FPGA," *Neural Comput. Appl.*, vol. 33, no. 15, pp. 9467–9479, Aug. 2021, doi: 10.1007/s00521-021-05706-3.

[47] E. Chung *et al.*, "Serving DNNs in Real Time at Datacenter Scale with Project Brainwave," *IEEE Micro*, vol. 38, pp. 8–20, Mar. 2018.

[48] J. Fowers *et al.*, "A Configurable Cloud-Scale DNN Processor for Real-Time AI," in *Proceedings of the 45th International Symposium on Computer Architecture, 2018*, Jun. 2018. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-configurable-cloud-scale-dnn-processor-for-real-time-ai/

[49] B. Betkaoui, D.B. Thomas, and W. Luk, "Comparing performance and energy efficiency of FPGAs and GPUs for high productivity computing," in *2010 International Conference on Field-Programmable Technology*, 2010, pp. 94–101. doi: 10.1109/FPT.2010.5681761.

[50] M. Qasaimeh, K. Denolf, J. Lo, K. Vissers, J. Zambreno, and P.H. Jones, "Comparing Energy Efficiency of CPU, GPU and FPGA Implementations for Vision Kernels," in *2019 IEEE International Conference on Embedded Software and Systems (ICESS)*, 2019, pp. 1–8. doi: 10.1109/ICESS.2019.8782524.

[51] P. Cooke, J. Fowers, G. Brown, and G. Stitt, "A Tradeoff Analysis of FPGAs, GPUs, and Multicores for Sliding-Window Applications," *ACM Trans Reconfigurable Technol Syst*, vol. 8, no. 1, p. 2, Mar. 2015, doi: 10.1145/2659000.

[52] J. Fowers, G. Brown, P. Cooke, and G. Stitt, "A Performance and Energy Comparison of FPGAs, GPUs, and Multicores for Sliding-Window Applications," in *Proc. ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, *FPGA '12*, New York, USA, 2012, pp. 47–56. doi: 10.1145/2145694.2145704.

[53] Y. Tang, R. Dai, and Y. Xie, "Optimization of Energy Efficiency for FPGA-Based Convolutional Neural Networks Accelerator," *J. Phys. Conf. Ser.*, vol. 1487, no. 1, p. 012028, Mar. 2020, doi: 10.1088/1742-6596/1487/1/012028.

[54] E.N. |A. May 2015, "Controlling the EMI effects of aircraft avionics," Aerospace Manufacturing and Design. [Online]. Available: https://www.aerospacemanufacturinganddesign.com/article/amd0415-aircraft-avionics-emi-effects/ (Accessed: Oct. 17, 2023).

[55] E. Cetin *et al.*, "Overview and Investigation of SEU Detection and Recovery Approaches for FPGA-Based Heterogeneous Systems," in *FPGAs and Parallel Architectures for Aerospace Applications: Soft Errors and Fault-Tolerant Design*, F. Kastensmidt and P. Rech, Eds., Cham: Springer International Publishing, 2016, pp. 33–46. doi: 10.1007/978-3-319-14352-1_3.

[56] S.-Y. Yuan *et al.*, "EMI behavior of FPGA-IP measurement," in *2022 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC)*, Sep. 2022, pp. 270–272. doi: 10.1109/APEMC53576.2022.9888692.

[57] "Aviation With FPGAs | Microchip Technology." [Online]. Available: https://www.microchip.com/en-us/solutions/aerospace-and-defense/aviation/fpga (Accessed: Oct. 17, 2023).

[58] "Avionics & UAV," AMD. [Online]. Available: https://www.xilinx.com/applications/aerospace-and-defense/avionics-uav.html (Accessed: Oct. 17, 2023).

[59] M. Petko and T. Uhl, "Smart sensor for operational load measurement," *Trans. Inst. Meas. Control*, vol. 26, no. 2, pp. 99–117, Jun. 2004, doi: 10.1191/0142331204tm111oa.

[60] R. Grzejda, "Finite element modeling of the contact of elements preloaded with a bolt and externally loaded with any force," *J. Comput. Appl. Math.*, vol. 393, p. 113534, 2021, doi: 10.1016/j.cam.2021.113534.

[61] M. Bečvář and P. Štukjunger, "Fixed-point arithmetic in FPGA," *Acta Polytech.*, vol. 45, no. 2, pp. 67–72, 2005.

[62] R. Yates, "Fixed-point arithmetic: An introduction," *Digit. Signal Labs*, vol. 81, no. 83, p. 198, 2009.

[63] S. Cherubin, D. Cattaneo, M. Chiari, A. Di Bello, and G. Agosta, "TAFFO: Tuning assistant for floating to fixed point optimization," *IEEE Embed. Syst. Lett.*, vol. 12, no. 1, pp. 5–8, 2019.

[64] D. Cattaneo, A. Di Bello, S. Cherubin, F. Terraneo, and G. Agosta, "Embedded operating system optimization through floating to fixed point compiler transformation," *21st Euromicro Conference on Digital System Design (DSD)*, IEEE, 2018, pp. 172–176.

[65] W. Mucha, W. Kuś, J.C. Viana, and J.P. Nunes, "Experimental Validation of Numerical Model of Composite Panel for Aerospace Structural Applications," in *Proceedings of the 26th International Conference Engineering Mechanics 2020,* Czech Republic, 2020.

[66] W. Mucha, W. Kuś, J.C. Viana, and J.P. Nunes, "Comparison of numerical and experimental strains distributions in composite panel for aerospace applications," in *Proceedings of the 6th ECCOMAS Young Investigators Conference,* 7th-9th July 2021, Valencia, Spain, 2022, pp. 403–411, doi: 10.4995/YIC2021.2021.15320.

[67] S.W. Tsai and E.M. Wu, "A General Theory of Strength for Anisotropic Materials," *J. Compos. Mater.*, vol. 5, no. 1, pp. 58–80, Jan. 1971, doi: 10.1177/002199837100500106.