Neural network impact vibration characterization for mobile additive manufacturing

Calem Young, Murat Muradoglu, Chern Ferng Chung, Tuck Wah Ng*

Laboratory for Optics and Applied Mechanics, Department of Mechanical & Aerospace Engineering, Monash University, Clayton VIC3800 Australia.

Abstract

Mobile additive manufacturing (AM) offers a way to ameliorate the long build time problem and benefits greatly if the print job can be abandoned and restarted anew as soon as a significant defect is anticipated. This study proposes a monitoring system using accelerometer data to analyze impact vibrations experienced when vehicle-mounted printers encounter speed bumps. A neural network (NN) Gaussian model method was developed to process these vibration signals, and demonstrated superior performance when compared to the traditional damped least squares approach. The NN method achieved more favorable loss distributions, which were characterized by higher density of loss values near the origin as well as loss distributions that were narrower and skewed towards lower values. The NN approach also achieved good performance when applied on continuous experimental data, achieving throughput of up to 10,133 samples per second. These findings portend the feasible use of the NN method to support the strategy of expeditiously deciding print job cessation based on impact vibration characteristics in mobile AM.

* Corresponding author: engngtw@gmail.com

Keywords: Machine learning; 3D printing; defect; deformation; accelerometer;

1. Introduction

Additive manufacturing (AM), or 3D printing, has emerged as one of the most significant recent technological breakthroughs in production. It promises strong cost savings due to factors such as reduced labor, transportation, raw material, and inventory demands, whilst offering process simplification and flexibility [1]. The ability of AM to enhance sustainability in manufacturing is well established [2] which serves as a crucial component towards the desired creation of urban smart factories [3]. One of the key challenges facing AM is the relatively long build time incurred. The manufacturing of 3D printed products en-route in vehicles, commonly referred to as mobile AM, has been suggested as a way to ameliorate the build time problem [4, 5]. Logistical models have been devised to assess the feasibility of this approach [6]. These models, however, assume that the part build can be achieved flawlessly. In reality, defects appear to the extent that the 3D printed part can at times be deemed unusable. Considering the long build times that are typically involved, abandoning a print job as soon as a significant defect is detected and restarting the print afresh is a logical approach to adopt to conserve time and material for sustainability. One method to do this is by monitoring the printing process in real time using camera recording coupled with a detection scheme that utilizes machine learning [7]. However, this monitoring approach is only viable for routine manufacturing processes where extensive image training datasets are available, making it unsuitable for unique parts. Vibration is one of the possible sources of build defects in AM that has been widely researched [8, 9]. Commercial 3D printers typically include vibration isolation systems that produce acceptable print quality for parts with moderate tolerance requirements. However, these standard isolation features may be insufficient for mobile AM applications, where vehicles encounter significant road disturbances such as speed bumps [10]. The impact vibration encountered can engender significant build part defects notwithstanding the vibration isolators that have already been incorporated into the 3D printer. It is useful, hence, to be able to identify and characterize the impact vibrations in mobile AM as a first step towards relating them to potential build defects.

A feature that will be useful in mobile AM is for any impact vibration to be characterized rapidly. This allows for shorter periods between sampling and for more timely decisions to be made on whether to abort the printing operation. Regression strategies, often described as data curve-fitting, attempt to model the observational data as functions using model parameters that depend on one or more independent variables. As this approach requires the data to be matched through a series of successive approximations, the process can be slow even with a processor with good computational power, since it does not take advantage of parallel operations that are possible with graphics processing units (GPUs). Neural network approaches have been adopted to improve the accuracy of data curve-fitting [11]. The capacity to improve processing speed has also been claimed [12], albeit there has been admittedly limited evidence to support this claim with contemporary processors.

In this work, the signals that are derived from an accelerometer mounted on a 3D printer subjected to significant impact forcing as a vehicle travelling over speed bumps will be processed using a lightweight neural network Gaussian modelling

scheme. Its performance is assessed against a typical nonlinear regression method in terms of immunity to noise as well as the speed involved in revealing the impact vibration characteristics. The results will then be analyzed to evaluate the system's ability to support rapid decision-making for print job cessation when significant defects are anticipated.

2. Methods

2.1 Accelerometer measurements of vehicle hump interaction

Consider a 3-axis accelerometer that is originally oriented to measure from the x, y, and z axes to produce outputs of A_x, A_y and A_z respectively (see Figure 1A). The accelerometer can generally be angularly displaced such that θ is the angle between the gravity vector (1g field) and the *z*-axis, φ is the angle between the horizon plane (0g field) and the *x*-axis, and ψ is the angle between the horizon plane and the *y*-axis (see Figure 1B). Suppose that the accelerometer is placed on the 3D printer (which in turn is placed inside a vehicle) such that φ and ψ are zero, the interaction of the vehicle with a pothole or speed bump causes changes in θ, and the direction of travel of the vehicle is in the x-axis direction. It will not be difficult to establish that the readings of A_x and A_z will change according to θ via

$$A_x = g\ sin\ sin\ \theta \tag{1a}$$

$$A_z = g(1 - cos\ cos\ \theta\ ) \tag{1b}$$

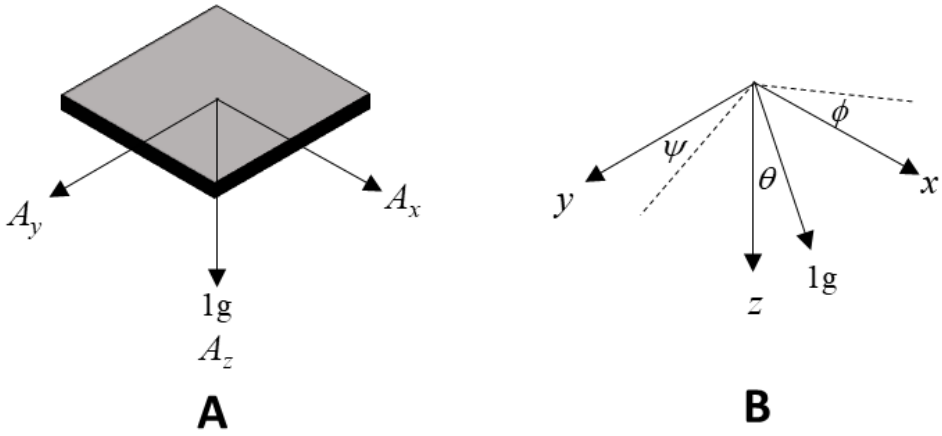

Figure 1. A 3-axis accelerometer can be oriented (A) such that Az produces readings corresponding to the gravity vector. The accelerometer can be angular displaced (B) such that its orientation is deviated from the gravity vector and horizon plane by the angles θ, φ and ψ.

When the vehicle encounters the speed bump under constant speed, the A_z distribution would logically be the most dominant. The underlying mechanics behind this interaction comprises three phases; (a) an initial impact tending to lift the wheel upwards, (b) an inversion of travel direction from rising to falling after the wheel has traversed the top of the bump, and (c) a second upward acting reaction impact when the wheel contacts the speed bump again (see Figure 2). A time trace of A_z against time would reveal maximal values of acceleration (A, B, C of Figure 2 inset) corresponding to this interaction. The acceleration versus time traces

that encompass the phases (a), (b) and (c) manifest as a sequence of Gaussian -like distributions. Such a normal function *f* can be generated in relation to the variable *x* using the parameters *a*, *b*, and *c* in which

$$f(x) = a\ exp\left(-\frac{(x-b)^2}{2c^2}\right) \tag{2}$$

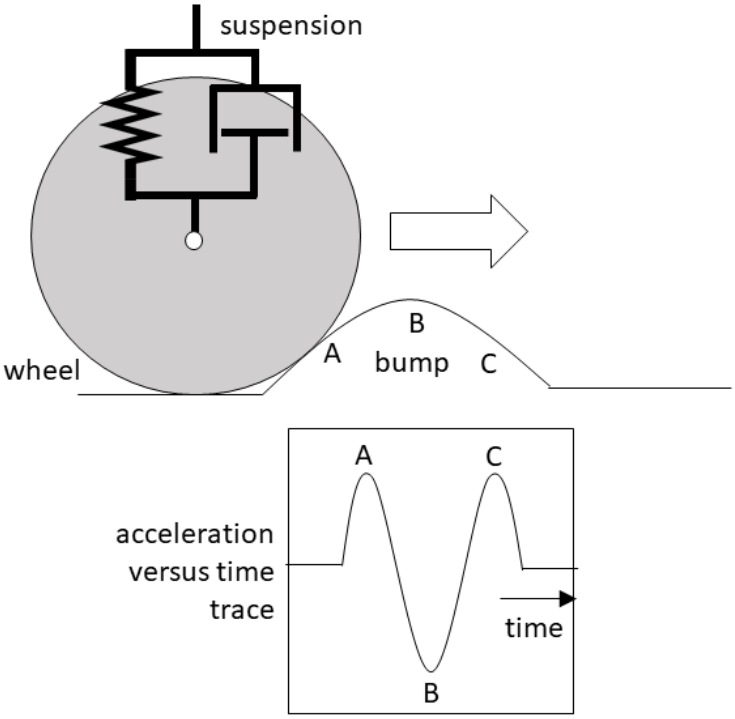

Figure 2. The interaction mechanics of a vehicle wheel with a speed bump in which the suspension is modeled as a spring damper combination. The inset depicts the ideal z-axis acceleration with time in which A denotes the first maximal upward impact after the wheel hits the bump, B represents the maximal downward acceleration when the wheel starts to fall after lifting off the bump, and C denotes the maximal upward impact when the wheel contacts the bump again.

The ability to characterize the impact vibration will thus involve finding the normal functions that best fits the data so that parameters *a*, *b*, and *c* in Equation (2) can be derived. It is also important to note that this normal distribution will typically be overlaid over by higher frequency fluctuations.

In order to determine the influence of the speed bumps, it is necessary to locate the accelerometer on a non-moveable part of the 3D printer. In addition, its location should also be equally sensitive to inclinations that the 3D printer would undergo in the clockwise or anticlockwise sense (relative to the value of θ at rest). In the test setup, a smartphone (Samsung Galaxy S22) was attached to the 3D printer (Creality, Ender 3) at the midspan of the top horizontal frame. The acceleration values were obtained by running an app (Sensor Logger) that extracted data from the inertial measurement unit (IMU) present in the smartphone. In a real-world situation, the 3D printer would be located inside a vehicle from which the acceleration data is collected en-route and processed [13] (see Figure 3).

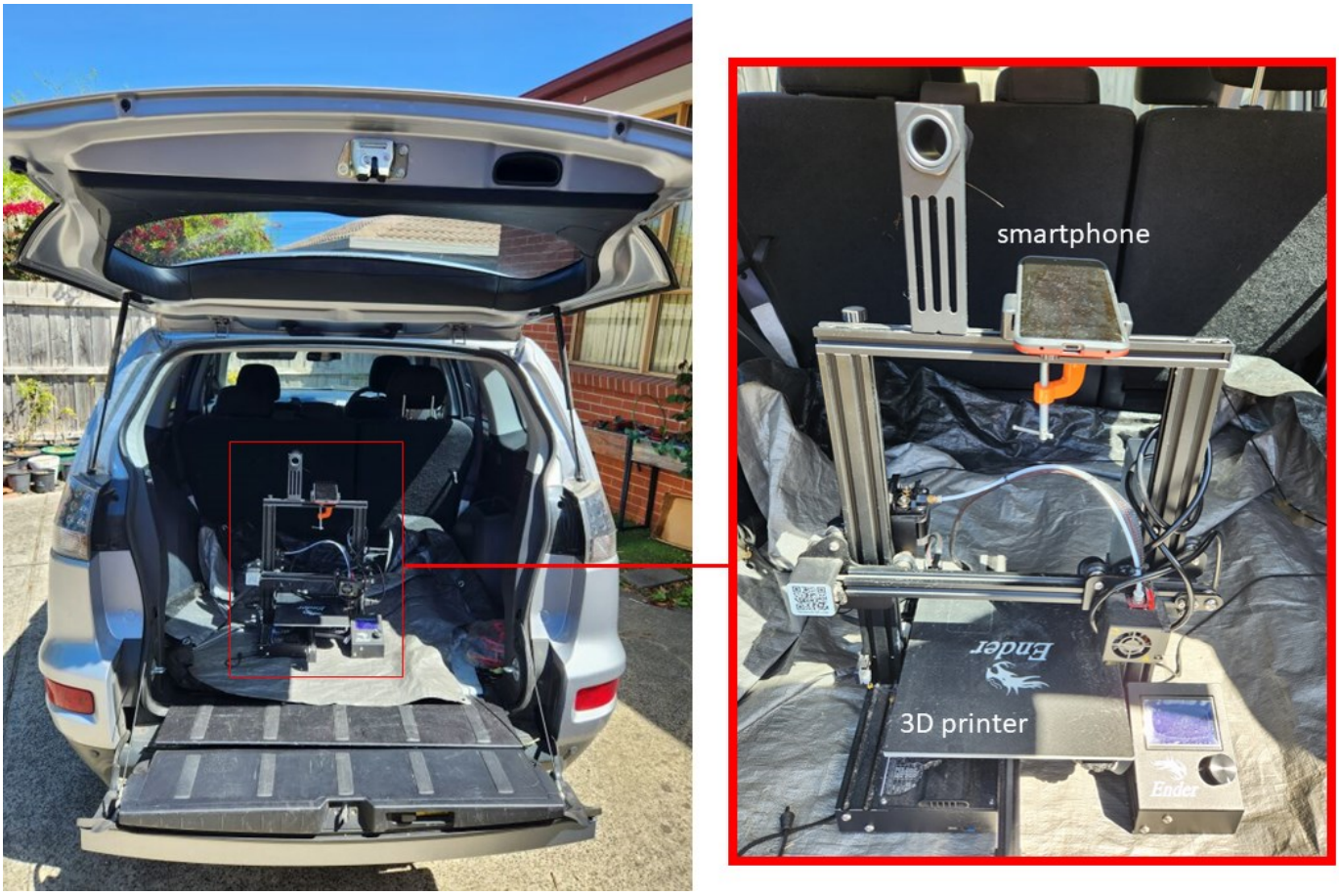

Figure 3. Depiction of how acceleration data could be obtained from a 3D printer located inside a vehicle when used to achieve mobile additive manufacturing. A smartphone, with an in-built inertial measurement unit, attached at the midspan of the top horizontal frame of the 3D printer is used to provide the acceleration data using an app.

2.2 Curve fitting using normal distribution parameterization

One method to obtain the parameters a, b, and c in Equation (2) is through a curve fitting operation on the signal versus time readings taken from the accelerometer. This operation can be carried out via regression using the non-linear least squares method based on the Levenberg–Marquardt algorithm [14, 15]. The algorithm seeks to minimize the objective function $J(p) = \sum d_i^2(p)$ where $d_i(p)$ represents the residual error between the observed data and the model's prediction at the parameter vector p [16]. To achieve this, the algorithm iteratively updates p by approximating $J(p)$ as a linear function around the current guess p_0

$$J(p) = \sum(d_i(p_0) + \nabla d_i(p_0) \cdot (p - p_0))^2 \quad (3)$$

where $d_i(p_0)$ is the residual at p_0, and $\nabla d_i(p_0)$ is its gradient. The algorithm then searches for an updated parameter vector p_{new} within a trust region to ensure that $J(p_{new}) < J(p_0)$. If this condition is met, p_{new} becomes the new estimate, and the process repeats until the objective function converges or another stopping criterion is met.

2.3 Curve fitting using a neural network model with Gaussian model fitting

In a curve fitting problem, it would be typically necessary for a parametric function $f(x; a_1, a_2 \dots a_k)$ with coefficients $a_1, a_2 \dots a_k$ to be created that would best fit a set of *N* data points $\{x_i, y_i\}_{i=1}^{N}$ based on some error limiting condition. If this is based on minimizing the sum of squared residuals, then for random errors ε_i in the data points, we have

$$y_i = f(x; a_1, a_2 \dots a_k) + \varepsilon_i \tag{4}$$

$$arg \sum_{i=1}^{N} \{y_i - f(x; a_1, a_2 \dots a_k)\}^2 \tag{5}$$

This is essentially an approximation problem in which a linear combination made out of a series simple parameterized basis functions $h(x_i, a_i)$ are superimposed such that for β being the coefficients of *m* number of basis functions being used, such that

$$f(x) = \sum_{i-1}^{m} \beta_i h(x; a_i) \tag{6}$$

The target function can be approximated by a single-layer feedforward network provided that the number of neurons is sufficiently large. This allows for a set of free parameters w to be determined, such that a function using these parameters is generated that would serve as an approximation for the target function. Logically, two layers of the feedforward network should be enough to provide an approximation of any continuous function with the set of data that is available. Under this mode, the function for fitting can be approximately represented by a linear combination of a series logistic function if the activation functions at the hidden layer and output layer are chosen such that they are logical and linear. The values of the network parameters and weights can be determined by using a backpropagation (BP) training algorithm.

The model architecture adopted employs a dual-path approach for feature extraction from the input signal (see Figure 4). One path uses convolutional layers to register any localized patterns within the temporal data, while another path leverages a Fast Fourier Transform (FFT) to extract frequency domain features. The temporal path consists of two 1D convolutions with Rectified Linear Unit (ReLU) activations and maximum pooling, that ultimately flattens into a 64x32 feature vector. The FFT path processes the frequency components of the input signal through a similar convolutional structure before flattening to a 32x32 output. These two feature sets are then concatenated to create a data representation that combines both spatial and frequency domain information. The combined feature vector is then fed into a series of fully connected layers (first with 256 and then with 128 layers using ReLU) which culminates in a final layer that comprises 3 neurons. The raw output of these three neurons correspond to the parameters of a standard form Gaussian function that consists of the amplitude, mean and standard deviation. To ensure that these parameters are in the appropriate ranges, specific activation functions were chosen. The Sigmoid function is used to constrain the amplitude and mean values to between 0 and 1. The Softplus function is used to ensure that the standard deviation values are positive.

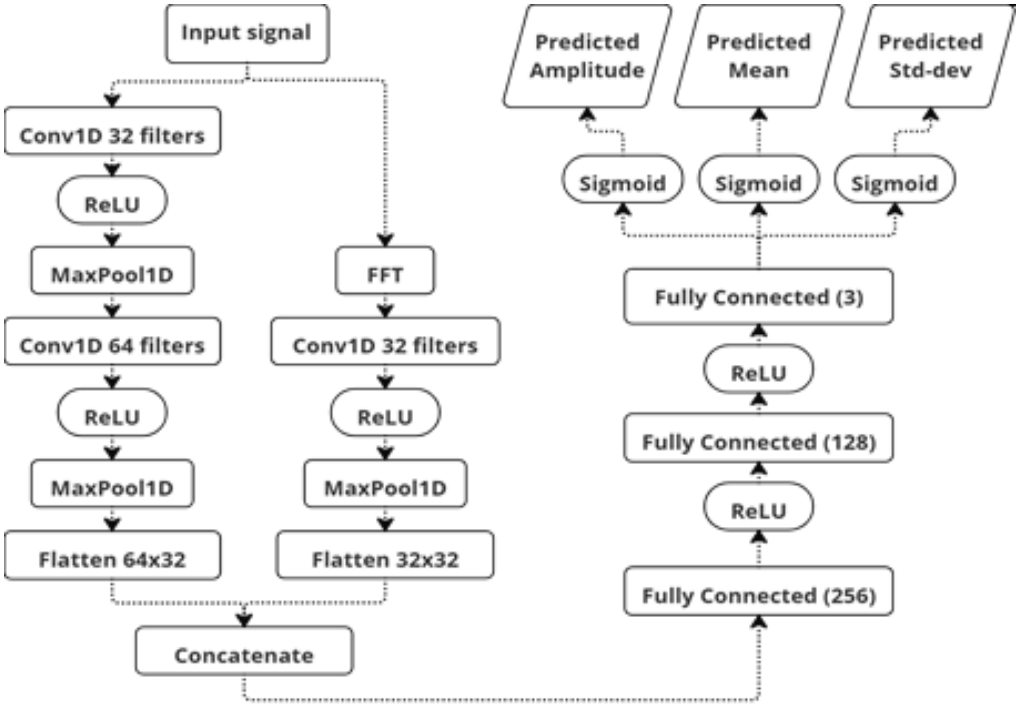

Figure 4. Flowchart depiction of the model architecture for the training adopted for the neural network model developed.

During training, the model weights were optimized to best fit Gaussian distributions to the input signals. This involves adopting a custom loss function [17, 18] as well as an adaptive learning rate [19] mechanism with an early stopping strategy. The loss function was defined as a composite of several components, with the intention of leveraging the known attributes of a Gaussian curve to improve the model's performance in signal reconstruction and parameter estimation. Individual loss components were used for each signal in the batch. A Gaussian reconstruction was done to ensure that the reconstructed Gaussian signal $\hat{y}$ would closely align with the known target signal y using the Mean Squared Error (MSE) metric

$$L_{reconstruction} = \frac{1}{N} \sum_{i=1}^{N} (\hat{y} - y)^2 \tag{7}$$

Here, N is the number of samples in the signal. This process encourages overall shape similarity between the predicted and target signals. Each Gaussian parameter was used to contribute to one individual loss term in order to improve the accuracy of each estimated parameter. The loss in accounting for the amplitude A, mean μ and standard deviation σ was calculated using

$$L_{params} = |\hat{A} - A| + |\hat{\mu} - \mu| + |\hat{\sigma} - \sigma| \tag{8}$$

The variables with a hat symbol in the equation denote the reconstructed values of the corresponding variable. To aid in peak detection, the signal-to-noise ratio (SNR), threshold, sigmoid, and smoothness functions were used to attain a peak score where

$$S_{peak} = sigmoid(SNR_{target} - threshold) \times (1 - smoothness) \tag{9}$$

The presence of any peak was established using

$$has\_peak = sigmoid((S_{peak} - peak\_threshold) \times scale\_factor) \quad (10)$$

To help minimize incorrect peak predictions, a penalty term that is scaled based on the amplitude was also introduced where

$$L_{incorrect-peak} = (1 - has\_peak) \times A \quad (11)$$

While it is possible to derive experimental data for model training using the readings from an accelerometer mounted on the 3D printer located inside a vehicle as it negotiates through numerous speed bumps [19], this would be cumbersome and potentially result in significant wear in the vehicle's suspension system as well as damage to the 3D printer. Instead, we used a synthetic dataset consisting of 10,000 Gaussian generated signals for training. The use of synthetic datasets is well established in NN training [20] and can be effective when underpinned by physics informed neural networks (PINN) strategies in which one or more governing equations can serve as the loss function [21]. The formulation of the synthetic signals, incorporating a dominant peak, secondary artifacts and environmental noise, is described as

$$f(x) = A_1 e^{-\frac{(x-\mu_1)^2}{2\sigma_1{}^2}} + \sum_{i=2}^{n}(\alpha_i \times A_1) e^{-\frac{(x-\mu_i)^2}{2\sigma_i{}^2}} + N(0, \sigma_{noise}^2) \quad (12)$$

The first term represents the dominant Gaussian peak, where A_1 is the amplitude, μ_1 is the mean (or center of the peak), and σ_1 is the standard deviation, which defines the width of the peak. The second term describes secondary artifacts that simulate additional vibrational effects (e.g., damping noise). Here the amplitude is scaled by a scale factor, α_i, which reduces the amplitude relative to the dominant peak A_1. Each secondary peak is defined with its own mean, μ_i, and standard deviation, σ_i, allowing for variation in their position and width. The third term represents Gaussian white noise, denoted by $N(0, \sigma^2_{noise})$, which simulates environmental vibrational noise encountered during mobile 3D printing. This noise term follows a normal distribution with zero mean and a variance of σ^2 noise, ensuring the synthetic data realistically mimics the stochastic nature of environmental disturbances. By introducing variability to the amplitude, mean and standard deviation of the synthetic Gaussian signals, the dataset is made to reflect a wide range of potential vibrational events. The model training utilized the Adam optimizer, which is known for its efficiency in handling sparse gradients and noisy data. The learning rate was initially set to 0.001 and a weight decay of 1e-5 was applied to avoid overfitting. To adapt to the training dynamics, a scheduler was employed so that the learning rate was reduced by half should there be no improvement obtained in the loss for 5 consecutive epochs.

3. Results and Discussion

The simulations and neural network processing were conducted on a PC using a 13th Gen Intel Core i5-13600K CPU. The neural network processing also involved use of a NVIDIA GeForce RTX 3070 GPU installed in the PC. The performance of the NN model here is compared against a widely-used implementation of the Levenberg–Marquardt algorithm, known as the

damped least squares (DLS) method, to establish a baseline comparison with standard fitting techniques. To appreciate the context of this comparison, the underlying workings of each approach can be outlined. The NN model uses extensive pre-training whereby the underlying weights are adjusted to create a model that can be generalized to find the best Gaussian fit based on the training data supplied. In doing so, the model has to deal with any inherent noise or disturbances. The DLS method, alternatively, makes iterative attempts to construct a Gaussian curve that best fits each signal independently at runtime. While this approach tends to be immune to any possible overfitting, it can nevertheless lead to the approach being less robust in dealing with more extreme noise or secondary artifacts in the signal.

Figure 5 presents the density distributions of the losses calculated from 10,000 samples of synthetically generated signals. It revealed the NN model to have a higher density of loss values near the origin. This indicates that the loss distribution of the NN model tended to be narrow and skewed towards lower values. In contrast, the DLS method has a larger spread and gravitated towards higher loss values. This broadly suggests that the DLS method tended to produce less consistent fits compared with the NN model.

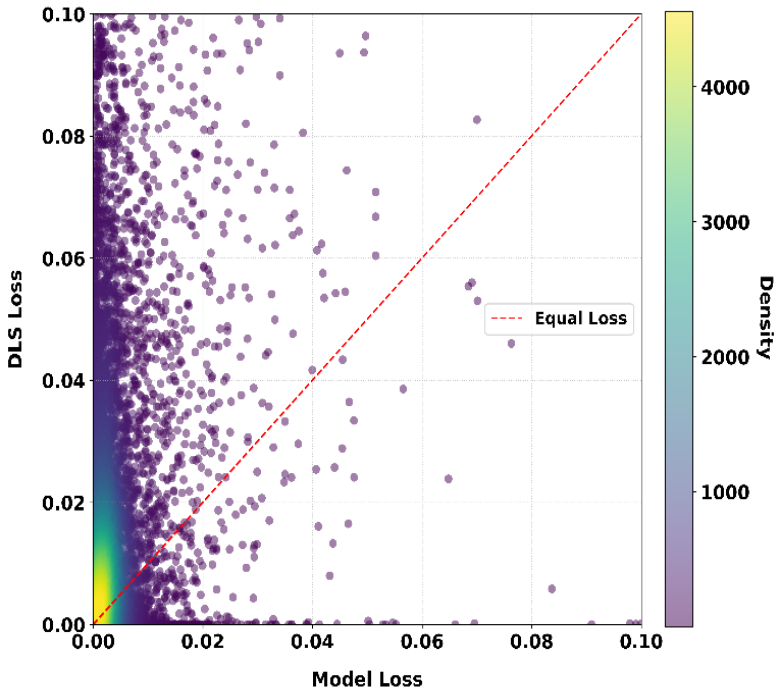

Figure 5. The loss density distribution using the neural network (NN) model compared with the damped least square (DLS) model based on the 10,000 synthetic simulation signals generated. The NN model outcomes tended to be narrower as well as skewed towards lower values.

Further analysis also found that the NN model's cumulative distribution function exhibited a steeper rise, indicating a probability distribution that is primarily composed of a low level of losses. The DLS cumulative distribution function, alternatively, is comparatively less steep, with a distribution coverage that extends for a larger range of loss values. This result imputes the DLS being less consistent and providing lower accuracy results.

In the context of mobile AM, the ability to process the accelerometer signals with minimal computational delay is critical. For both the NN and DLS methods, the time efficiency of fitting is crucial to the system's overall efficacy, as it dictates whether the data can be processed fast enough to prevent information loss or delays in responding to critical vibrational events. Failure to process the data in a sufficiently timely manner could result in buffer overflow or unacceptably delayed systems responses, which can prevent timely interventions to be made to determine if the vibrational disturbance is of sufficient merit to result in the print process being abandoned.

The comparative performance of the NN Fitting Model was assessed against the DLS method by measuring the time required to fit the 10,000 synthetic signals, processed sequentially, without batching. The cumulative processing time is presented in Figure 6, where the NN Fitting Model demonstrates a superior processing speed relative to the DLS method. This result aligns with expectations, as the DLS method relies on iterative optimization for each individual signal, whereas the NN model, having undergone pre-training, operates primarily through inference from matrix multiplication. This inherent advantage allows the NN model to classify signals with higher computational efficiency, positioning it as the more suitable choice for applications where time efficiency is important.

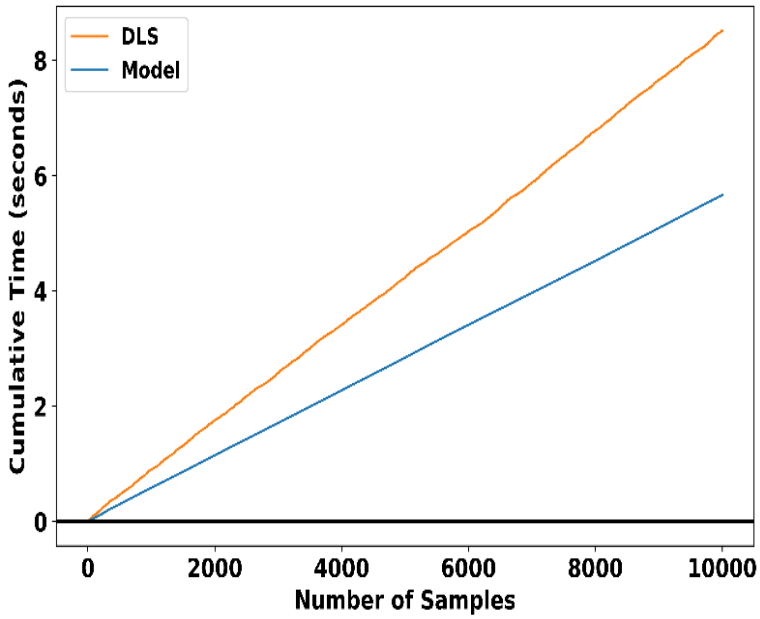

Figure 6. The cumulative time taken by the NN and DLS models to process the 10,000 synthetic simulation signals generated.

Figures 7(A) to (F) provide typical z-axis experimental acceleration versus time traces recorded with the vehicle travelling over speed bumps. While there was an expected variation in these values, accelerations as high as 8g were recorded. This underlined the susceptibility for print failures to occur when the vehicle encountered speed bumps. In virtually all the cases, the acceleration associated with an inversion of travel direction from rising to falling after the wheel had traversed the top of the bump (point B in Figure 2) tended to be dominant. By doing a simple sign inversion, it was possible for the NN model to generate good Gaussian

fittings of this phase alone while ignoring the phases attributed to the initial impact that tended to lift the wheel upwards, as well as the second upward acting reaction impact when the wheel contacted the speed bump again. Such a customized fitting process would be more difficult to achieve using the DLS method.

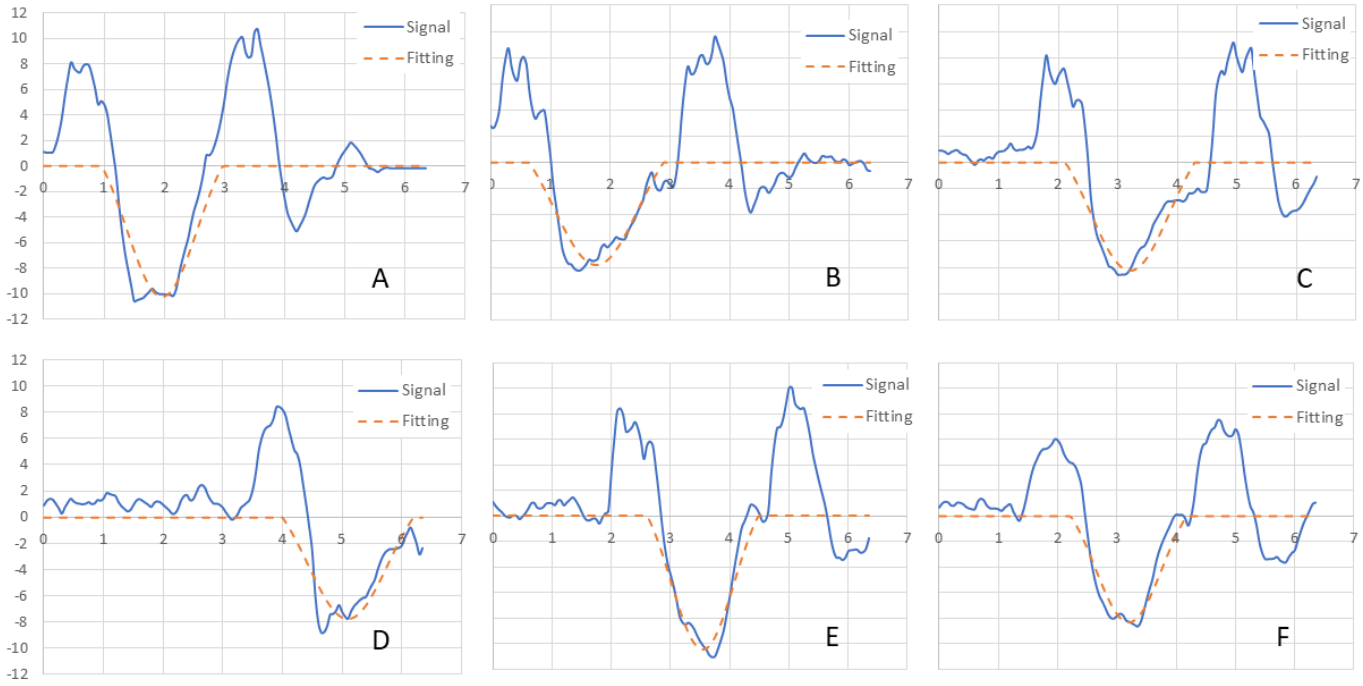

Figure 7. Experimental accelerometer (z-axis) signals (A to F) recorded using the setup depicted in Figure 3 along with their respective fitted distributions derived using the NN model developed.

When processing the experimental data, an average processing latency of a mere 4.82ms per window (128 sample points) was found when utilizing the RTX3070, in which processing times ranged from 4.37ms to 28.3ms. The NN model thus was able to achieve a throughput of 10,133 samples per second which translated to a capacity of up to 158 windows per second of processing speed. These metrics indicate that the NN model is well-optimized for the hardware used and can maintain a consistent processing performance when continuous data streaming is implemented. These results also impute the model's suitability for applications that apply sampling rates of within 10kHz, with a sufficient margin allocated for processing overhead. At the maximum latency of 28.3ms, the system would still be able to accommodate data streams that are limited to 35 windows per second.

4. Conclusion

This study examined how accelerometer data from a 3D printer used in mobile additive manufacturing (AM) can reveal impact vibration characteristics when the vehicle experiences strong gravitational effects, such as while driving over speed bumps. Using 10,000 synthetically generated signal samples, the implemented neural network (NN) model was found to produce a higher concentration of loss values near zero, with distributions that were narrower and skewed toward lower values compared

to the DLS method. The NN approach also demonstrated faster processing speeds. When applied to continuous experimental data, the NN method achieved throughputs of up to 10,133 samples per second. It was also able to yield the acceleration versus time trends associated only with an inversion of travel direction from rising to falling after the wheel of the vehicle after it traversed the top of a speed bump. These results indicate that the NN approach could effectively support a strategy in which impact vibration characteristics are rapidly assessed to determine whether printing operations in mobile AM should be halted. This capability will naturally need to be supported by additional studies that correlate print quality with impact vibration behavior. The model's lightweight construction and suitability for real-time execution also make it a strong candidate for deployment at the edge on low-power microcontrollers, enabling low-latency responses [22]. Additionally, given the known link between vibration and nozzle clogging in 3D printing [23, 24], this approach may also be extended to address that issue as well.

Acknowledgements

Specific materials and instruments made available by Dextech Technologies Pty Ltd are appreciated.

References

1. Kunovjanek, M., Knofius, N., Reiner, G. (2023) Additive manufacturing and supply chains - a systematic review. Production Planning and Control, 33, 1231–1251.
2. Daareyni, A., Pagone, E., Thayapararajah, S., Mokhtarian, H., Tosello, G., Ituarte, I. F. (2025) "Intelligent manufacturing paradigms: linking design optimization and sustainability in large-area additive manufacturing," The International Journal of Advanced Manufacturing Technology, DOI:10.1007/s00170-025-15832-0.
3. Sajadieh, S. M. M., Noh, S. D. (2024) "Towards sustainable manufacturing: A maturity assessment for urban smart factory," International Journal of Precision Engineering and Manufacturing, 11, 909–937.
4. Ryan, M. J., Eyers, D. R., Potter, A. T., Purvis, L., Gosling, J. (2017) "3D printing the future: scenarios for supply chains reviewed," International Journal of Physical Distribution and Logistics Management, 47, 992-1014.
5. Abdolazimi, O., Ma, J. (2024) "An integrative production-delivery system with mobile additive manufacturing and truck-drone delivery considering optimal printing sequence and preferred delivery time window," International Journal of Production Research, DOI: 10.1080/00207543.2024.2354827
6. Kenger Z.D., Koç, C., Özceylan, E. (2021) "Integrated disassembly line balancing and routing problem with mobile additive manufacturing," International Journal of Production Economics, 235, 108088.
7. Khan, M. F., Alam, A., Siddiqui, M. A., Alam, M. S., Rafat, Y., Salik, N., Al-Saidan, I. (2021) "Real-time defect detection in 3D printing using machine learning," Materials Today Proceedings, 42, 521-528.

8. Jensen N. J., Parker, G. G., Blough, J. R. (2023) “Base vibration effects on additive manufactured part quality,” Experimental Techniques, 48, 159–170.
9. Srivastava, A., Gautam, C., Bhan, N., Dayal, R. (2019) “Vibration analysis of cantilever shaped 3D printers,” IOP Conference Series: Materials Science and Engineering 594, 012020.
10. Kırbas U., Karasahin, M. (2018) “Comparison of speed control bumps and humps according to whole-body vibration exposure,” Journal of Transportation Engineering, Part A: Systems, 144, 04018054.
11. Dong, S., Liu, Y., Yu, H., Wang, Y., Wu, J., (2024) “An iterative curve-fitting baseline correction method for Raman spectra driven by neural network,” Applied Spectroscopy, 78, 111-119.
12. Bishop, C. M., Roach, C. M. (1992) “Fast curve fitting using neural networks,” Review of Scientific Instruments, 63, 4450–4456.
13. McMorran, D., Samarasinghe, S. K., Muradoglu, M., Chung, D. C. K., Williams, B., Liew, O. W., Ng, T.W. (2018) “Sensor and actuator simulation training system for en-route intravenous procedure,” Sensors and Actuators A 279, 680-687.
14. Levenberg, K. (1944) “A method for the solution of certain non-linear problems in least squares,” Quarterly of Applied Mathematics 2, 164-168.
15. Marquardt, D. W. (1963) “An algorithm for least-squares estimation of nonlinear parameters,” Journal of the Society for Industrial and Applied Mathematics 11, 431-441.
16. Shakarji, C. M. (1998) “Least-squares fitting algorithms of the NIST algorithm testing system,” Journal of Research of the National Institute of Standards and Technology, 103, 633-641.
17. Dessain, J. (2023) “Improving the prediction of asset returns with machine learning by using a custom loss function,” Advances in Artificial Intelligence and Machine Learning, 3, 1640-1653.
18. Chen, J., Pierce, J., Williams, G., Simpson, T. W., Meisel, N., Narra, S. P., McComb, C. (2024) “Accelerating thermal simulations in additive manufacturing by training physics-informed neural networks with randomly synthesized data,” Journal of Computing and Information Science in Engineering, 24, 011004.
19. Takase, T., Oyama, S., Kurihara, M. (2018) “Effective neural network training with adaptive learning rate based on training loss,” Neural Networks 101, 68-78.
20. Manettas, C., Nikolakis, N., Alexopoulos, K. (2021) “Synthetic datasets for deep learning in computer-vision assisted tasks in manufacturing,” Procedia CIRP 103, 237–242.

21. Yoo, S., Kang, M., Yoon, H., Kim, T. (2025) “A physics-informed neural network approach for solving structural eigenvalue problem,” International Journal of Precision Engineering and Manufacturing https://doi.org/10.1007/s12541-025-01280-z.
22. Huang, H., Yang, L., Wang, Y., Xu, X., Lu, Y. (2021) “Digital twin-driven online anomaly detection for an automation system based on edge intelligence,” Journal of Manufacturing Systems 59, 138–150.
23. Tlegenov, Y., Hong, G. S., Lu, W. F. (2018) “Nozzle condition monitoring in 3D printing,” Robotics and Computer-Integrated Manufacturing 54, 45-55.
24. Ansoliya, A., Gupta, V. K. (2025) “Nozzle clogging detection in FFF 3D printing using machine learning approach: A comparative investigation,” International Journal of Precision Engineering and Manufacturing DOI: 10.1007/s12541-025-01284-9.