Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 104 (2021) 1753–1758

www.elsevier.com/locate/procedia

54th CIRP Conference on Manufacturing Systems

A 3D Deep Learning Model for Rapid Prediction of Structural Dynamics of Workpieces During Machining

Ali Maghami[a], Meshkat Salehi[a], Matt Khoshdarregi[a,*]

[a]*Intelligent Digital Manufacturing Laboratory, Department of Mechanical Engineering, University of Manitoba, Winnipeg, MB R3T 5V6, Canada*

* Corresponding author. Tel.: +1-204-474-6153; fax: +1-204-275-7507. *E-mail address:* m.khoshdarregi@umanitoba.ca

Abstract

Chatter stability in machining flexible parts depends directly on the structural dynamics of the workpiece. This paper proposes a novel two-stage framework that combines finite element (FE) and data-driven deep learning techniques to rapidly predict the varying dynamics of workpieces during machining. An automated framework is developed to create a large training dataset of CAD models with gradually-changing geometries. A deep 3D Convolutional Neural Network (3D-CNN) is developed to "learn" the variations in dynamic parameters as a function of geometry. The current model has been successfully implemented for prediction of natural frequencies of workpieces during turning operations. The proposed framework can be used as a computationally efficient tool in online process monitoring and automated correction applications.

Keywords: Machining dynamics; FRF updating; structural modeling; finite element; deep learning; convolutional neural networks

1. Introduction

Chatter vibrations in machining operations are detrimental to the tool life and surface quality of the workpiece. In machining processes such as turning [1,2] and thin-wall milling [3], chatter typically happens due to the structural flexibility of the workpiece. Predicting and controlling chatter in these operations can be challenging due to the varying structural dynamics of the workpiece as material is removed [4].

In order to prevent chatter, the common practice is to use stability lobes ahead of time to select proper machining parameters such as depth of cut and spindle speed [5]. In addition, it is possible to design active and passive damping systems to suppress vibrations during operation [6,7]. These techniques typically require knowledge of the varying structural dynamics of the workpiece during the entire operation in order to tune the parameters of the damper.

Finite element (FE) has been used extensively as an offline tool for predicting the structural dynamics of workpieces and planning the process ahead of time [8,9,10]. However, with the new paradigms of flexible and intelligent manufacturing (Industry 4.0), machines will need to be able to make certain process planning and corrective decisions in real time without any human intervention [11]. For example, a machine may need to adjust the feedrate, depth of cut, and even tool path based on feedback from its real time monitoring system. In such cases, the machine must be able to quickly update the structural model of the workpiece in the digital twin and recalculate the optimum machining parameters such as spindle speed on the fly.

However, FE models are computationally expensive and can take anywhere from a few seconds to hours to run. Therefore, FE methods cannot be used for real time process planning and monitoring applications. To tackle this challenge, the present paper proposes a novel framework based on deep learning techniques that can rapidly predict the dynamic parameters of machined parts at any point within milliseconds.

In recent years, deep neural networks have been increasingly used to develop self-learning and fully automated data-driven models for engineering applications such as stress analysis [12,13], condition monitoring [14, 15], inspection [16], and part

2212-8271
10.1016/j.procir.2021.11.295

feature recognition [17]. In the present paper, a novel 3D convolutional neural network (3D-CNN) has been developed to learn the variation of structural dynamics of a part as a function of geometry. It should be mentioned that the proposed 3D-CNN does not replace FE models but rather learns from them so it can provide rapid predictions for real time applications.

Figure 1 shows an overview of the proposed framework. First, a large database of workpiece geometries is generated automatically using PythonOCC. The geometries are voxelized and represented digitally as 3D matrices and provided as inputs to the 3D-CNN. A finite element code has also been developed in MATLAB to perform modal analysis for each of the generated geometries. The corresponding dynamic parameters obtained from FE are used as output data for training the neural network. Once trained, the 3D-CNN can predict the structural parameters of a workpiece at any point during machining within a few milliseconds. The developed framework has currently been implemented for predicting the first five eigenfrequencies of the workpiece during turning operations. Extraction of the complete frequency response function (FRF) and extension to other machining operations such as thin-wall milling is currently in progress and will be published in future works.

The rest of the paper is organized as follows. Section 2 presents the automated CAD generation algorithm, and Section 3 details the developed FE module for modal analysis. The architecture of 3D-CNN and the training process is presented in Section 4, followed by training test results in Section 5. The paper is concluded in Section 6.

2. Modeling geometrical variations

In order to develop a deep learning model for prediction of varying dynamics of workpieces, the first step is to generate training data, i.e. a large dataset of geometries (input) and their corresponding eigenfrequencies (output). In this research, an automated CAD generator algorithm has been developed using the PythonOCC library. This algorithm generates a large number of CAD files to model the transformation of the workpiece during axial/radial turning, facing, and grooving operations subject to different material removal steps.

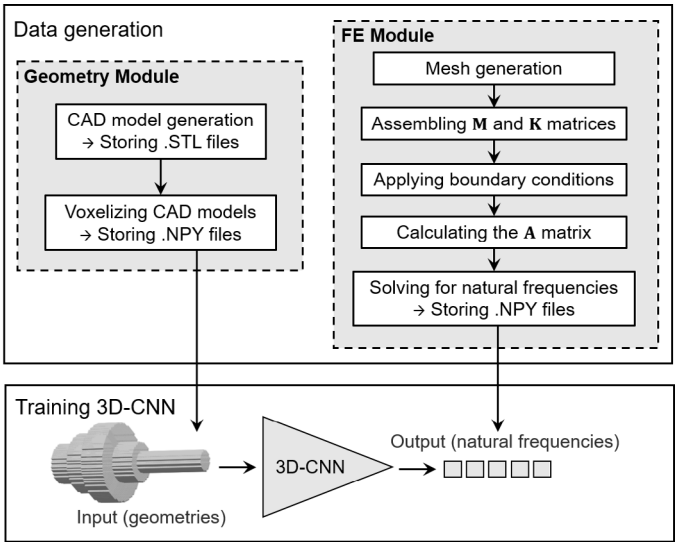


Fig. 1. Overall procedure of generating the data and training 3D-CNN.

As shown in Figure 2.a, consider the initial workpiece to be a cylinder with the inner radius of r_{in}, outer radius of r_{out}, and length of l. The geometry is defined with reference to a polar coordinate system (r, θ, z) for applying the geometrical transformations, and a Cartesian coordinate system (x, y, z) for FE mesh generation. The cylinder is considered clamped at $z = 0$ to represent the boundary condition at the spindle chuck.

To simulate the geometrical variations of the workpiece, the cylinder is divided into n_r radial and n_z axial segments as illustrated in Figure 2.b. Different machined geometries are modeled by removing a series of segments in different orders while ensuring the following two connectivity rules are satisfied:

i) At each axial segment, the number of removed radial segments can vary between 1 to $n_r - 1$. The most inner radial segments cannot be removed unless they are at the tip of the workpiece.

ii) At each axial segment, radial segments can only be removed from outer layers. Inner segments of the material cannot be removed.

Figure 3 shows sample geometries obtained using the developed CAD generator algorithm.

It should be noted that the appropriate number of axial and radial segments depends on the application and the required resolution in dynamic parameters. Based on the defined connectivity rules, it can be shown that the total number of possible geometries for a cylinder with n_z axial segments and n_r radial segments is

$$n_r^{n_z} + n_r^{n_z-1} + \cdots + n_r = \sum_{i=1}^{n_z} n_r^i \,. \tag{1}$$

However, not all possible geometries are practical from a machining point of view. Therefore, the total number of geometries can be reduced significantly by imposing limits on the location and maximum number of removed segments.

3. Eigenfrequency analysis of generated geometries

As the next step for generating training data, the eigenfrequencies corresponding to each of the created

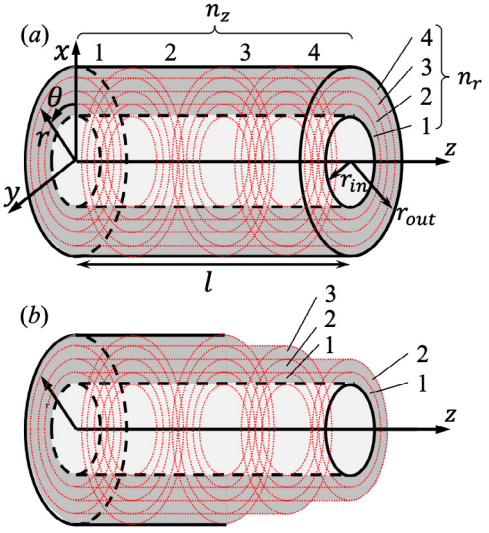


Fig. 2. (a) Definition of coordinate systems and geometry segmentation; (b) sample geometry obtained by removing segments.

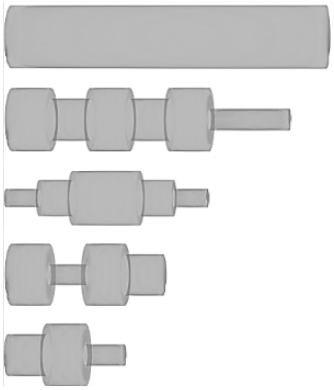


Fig. 3. Sample geometries created by the CAD generator algorithm.

geometries must be obtained. To achieve this, an FE code has been developed in MATLAB to automatically generate mesh for each geometry and solve the corresponding eigenvalue problem. As shown in Figure 4, each geometrical segment is first divided into smaller hexahedral elements by meshing the segment in the radial and axial directions. All of the elements of the workpiece are then assembled in the global stiffness ($\mathbf{K}$) and mass ($\mathbf{M}$) matrices to form a unified solid body. The generalized eigenvalue problem for free vibrations can then be written as

$$(\mathbf{K} - \omega^2\mathbf{M})\mathbf{\Phi} = 0, \tag{2}$$

where $\mathbf{\Phi}$ and ω are the matrices of mode shapes and natural frequencies, respectively. Natural frequencies can be found by solving

$$\det|\mathbf{K} - \omega^2\mathbf{M}| = 0. \tag{3}$$

In this work, the first 5 unique natural frequencies of the workpiece are obtained using the *eigs* function in MATLAB. This function interfaces with ARPACK [18], a highly efficient package which allows for fast computation of a limited number of eigenpairs in a standard eigenvalue problem. To solve the free vibration problem using ARPACK, Eq. (3) needs to be transformed into the standard eigenvalue form. To do so, the mass matrix $\mathbf{M}$ is diagonalized using the row sum method [19], and then decomposed as

$$\mathbf{M} = \mathbf{D}\mathbf{D}^{\mathbf{T}}, \tag{4}$$

where $\mathbf{D}$ is the diagonal matrix whose elements, d_{ii}, are

$$d_{ii} = \sqrt{m_{ii}}, \tag{5}$$

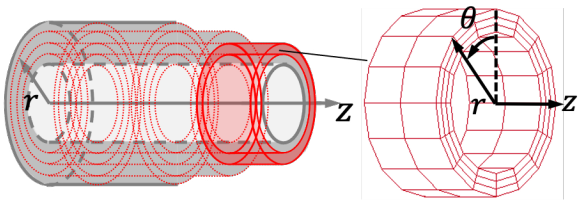


Fig. 4. Mesh generation for each geometrical segment for FE analysis.

and m_{ii} is the i^{th} element of the diagonalized mass matrix $\mathbf{M}$. Substituting Eq. (4) into Eq. (2) and introducing $\mathbf{\Psi} = \mathbf{D}\mathbf{\Phi}$ and $\lambda = \omega^2$, the standard eigenvalue problem can be found as

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{\Psi} = 0, \tag{6}$$

where $\mathbf{A} = \mathbf{D^{-1}}\mathbf{K}\mathbf{D^{-1}}$ and $\mathbf{I}$ is the identity matrix.

4. Training process

In order to provide the generated geometries as inputs to a deep learning model, each geometry must be represented as a matrix of input data. Voxelization of 3D objects is a common way for digital representation of 3D geometries [20]. As illustrated in Figure 5.a, the part is enclosed in a 3D grid. The grid network is designed based on the size of the largest part, i.e. the initial workpiece before machining. This 3D grid is kept the same for all generated geometries. Each voxel (small cube) in the 3D grid represents a binary digit indicating whether or not the voxel is located inside (belongs to) the part geometry. Using this method, the generated CAD models can be represented as a matrix of binary digits. The size of voxels must be selected in such a way that the voxelized model can properly capture the removal of geometrical segments in different steps. As illustrated in Figure 5.b, the cross section of the part is rendered through the axial direction. The value of voxels is set to 1 if the center of the voxel is inside the 3D volume occupied by the CAD model. A sample voxelized cylinder with a $20 \times 20 \times 100$ voxel grid is shown in Figure 5.c. The voxelized geometries are stored in NPY file format, which is an efficient format for loading the training dataset during the learning process.

It should be mentioned that in the training data, the natural frequencies (output) obtained from the FE model are based on the exact geometries, whereas the inputs are based on the voxelized models. Since in the training data the voxelized models are paired with the natural frequency of the exact

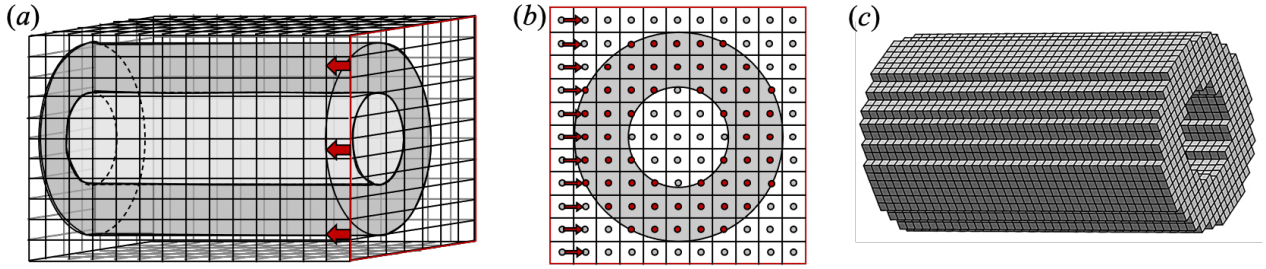


Fig. 5. Voxelization process; (*a*) A schematic of the 3D voxel grid containing the cylindrical part; (*b*) The circular cross section of the CAD model is rendered at each section along the axial direction of the cylinder; (*c*) A sample voxelized model.

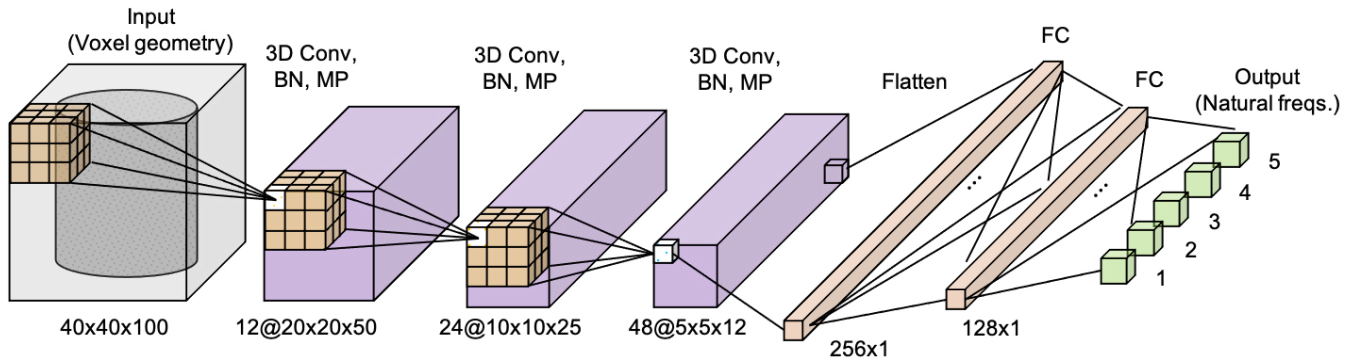


Fig. 6. Architecture of the 3D Convolutional Neural Network (3D-CNN).

geometries, the neural network learns to do the same. Basically, at least for the geometries that are close to the training data, the neural network can provide the same natural frequency as FE even if the input is the simplified (voxelized) model. In more general cases, voxelization will introduce some digitization errors, but these errors can be ignored compared to the prediction errors injected by the neural network.

Having generated the training data, i.e. the voxelized input geometries and their corresponding first 5 natural frequencies, the next step is to design and implement a deep learning model. In this research, a 3D convolutional neural network (3D-CNN) has been developed as shown in Figure 6. The first stage includes three blocks each comprising a 3D convolution layer followed by a batch normalization (BN) and max-pooling (MP) layer. During the convolution operations, each filter moves with a fixed stride along the width, height, and depth of the input geometry and generates a scalar number as the filter output. The rectified linear unit (ReLU) activation function is then employed to perform a nonlinear transformation on filter outputs and generate activation values. The 3D convolution layer receives a 5D tensor with the shape (batch size, input width, input height, input depth, 1). After convolving the 3D filters of shape $(f \times f \times f)$ with stride $(1 \times 1 \times 1)$, the convolution layer outputs a 5D tensor of shape (batch size, input width, input height, input depth, number of filters). In this research, the kernel size of $f = 3$ is used in all convolutions while 12, 24 and 48 filters are used for convolutions 1 to 3, respectively. After each convolution layer, batch normalization layer is employed. Batch normalization increases the stability and performance of deep networks by ensuring that the inputs of layers have a similar range [21]. After each batch normalization, a 3D max pooling layer with $(2 \times 2 \times 2)$ kernel is used to perform down sampling and reduce the spatial size. This layer replaces each $2 \times 2 \times 2$ subregion with its maximum value.

The last (third) convolution block is flattened and then followed by three fully connected (FC) layers. The FC layers reduce the dimensions of the 3D feature map and capture the nonlinear relationship between the high-level features and the output predictions. The ReLU function is also used as the activation function for FC layers. The final layer is an FC layer with 5 neurons that outputs the first five natural frequencies of the input geometry.

In the learning phase, the model parameters including weights and biases are trained so that the network can predict the ground truth (FE results) as accurately as possible. Mathematically, the problem is to find a 3D-CNN regression model $M(x)$ that minimizes the loss function $L(M(x), y)$, where x and y are the matrices of input geometry and natural frequencies, respectively. In this work, the mean squared logarithmic error (MSLE) has been employed as the loss function and defined as

$$\text{MSLE} = \frac{1}{n}\sum_{i=1}^{n}(\log(y_i + 1) - \log(\hat{y}_i + 1))^2, \tag{7}$$

where n is the number of output neurons, which here is 5, and y_i and $\hat{y}_i$ are the ground truth and predicted values of the output (natural frequencies), respectively.

5. Test results

The proposed 3D neural network has been implemented using the Keras library in Python with TensorFlow backend. The 3D-CNN model parameters are optimized through back-propagation and using Adam optimizer. A grid size of $40 \times 40 \times 100$ is used for the voxel representation of input geometries. A solid cylinder with a length of 0.5 m and a diameter of 10 cm has been geometrically segmented with 8 and 4 axial and radial segments, respectively. Material properties of steel with an elastic modulus of 200 GPa and density of 7,800 kg/m^3 have been used in FE solutions. In total, 87,380 sets of input geometries and their corresponding natural frequencies have been generated for training. Out of these, 15,000 samples have been used for the validation process during training, and 15,000 samples for the test phase. The remaining 57,380 samples are used as training data. The training process is performed with the batch size of 60 using an Nvidia RTX 2060 GPU with 6GB of GPU RAM. The mean absolute percentage error (MAPE) is used as the metric for evaluation of the prediction accuracy of the trained model, and is defined as

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|. \tag{8}$$

Figure 7 shows the MAPE metric as a function of training iteration (epoch) for training and validation data. It can be seen that after 300 epochs, overfitting starts without significant

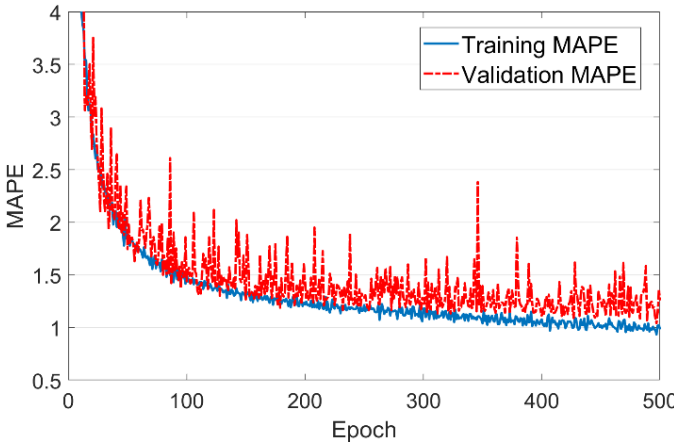


Fig. 7. Training and validation MAPE vs. training epochs.

improvement in validation accuracy. Therefore, the model at 300th epoch is used as the trained neural network for the test phase. The test data that are not seen by the model during the training test are used to evaluate the trained model. The MAPE value of 1.12% is found for the test data, which is close to the validation MAPE, 1.11%. This indicates the high performance of the trained network in predicting natural frequencies for different intermediate geometries. Table 1 lists some examples of the test data obtained from the trained 3D-CNN and FE. The natural frequencies obtained from FE are based on the exact geometries and not the voxelized models.

The 3D-CNN regression model learns to predict the corresponding natural frequencies based on the part geometry, i.e. mass distribution. However, as discussed in Section 2, all of the generated training geometries have been obtained by removing step-wise radial/axial segments (Figure 2.b). An investigation has been performed to assess the ability of the trained model to apply the learned relationship between mass distribution and natural frequencies to more general geometries. Table 2, shows some sample geometries with continuous surface profiles and their corresponding frequencies obtained from FE and 3D-CNN. None of these geometries were included in the training phase. It can be seen that the prediction accuracy is slightly lower with errors up to 10%. Higher accuracy can readily be achieved by using finer geometrical segmentations (Figure 2.b) in training dataset, but at the cost of increased computational load.

To provide a comparison in terms of computational time, both 3D-CNN and FE methods have been used to generate the solution for a sample part. A Core i5-8500B CPU has been used to perform both computations. The FE simulation with 2,304 elements was completed in 17.11 s while the 3D-CNN's prediction was generated in 0.05 s, i.e. over 100 times faster. If the 3D-CNN is run on a GPU, the computational time can be reduced to below 5 ms, which is suitable for online monitoring and control applications.

Table 1. Comparison between natural frequencies obtained from FE and 3D-CNN from (unseen) test data with step-wise geometry.

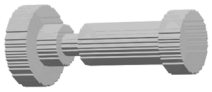
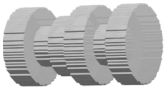
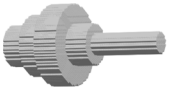
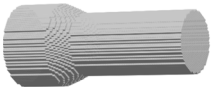
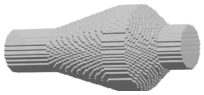
Input geometry to 3D-CNN	Method	ω_1 (Hz)	ω_2 (Hz)	ω_3 (Hz)	ω_4 (Hz)	ω_5 (Hz)
	FE	70	567	919	1838	2475
	3D-CNN	69	566	916	1843	2496
	Prediction error	1.4%	0.2%	0.3%	0.3%	0.8%
	FE	227	988	1489	2584	2621
	3D-CNN	229	999	1489	2561	2680
	Prediction error	0.9%	1.1%	0.0%	0.9%	2.2%
	FE	326	699	1489	2725	3497
	3D-CNN	336	694	1493	2770	3466
	Prediction error	3.1%	0.7%	0.3%	1.6%	0.9%

Table 2. Comparison between natural frequencies obtained from FE and 3D-CNN for new (unseen) geometries with continuous surface profiles.

Input geometry to 3D-CNN	Method	ω_1 (Hz)	ω_2 (Hz)	ω_3 (Hz)	ω_4 (Hz)	ω_5 (Hz)
	FE	308	1421	1954	2898	3334
	3D-CNN	292	1413	1865	2775	3435
	Prediction error	5.2%	0.6%	4.5%	4.2%	3.0%
	FE	242	1326	1365	2432	3426
	3D-CNN	234	1260	1406	2347	3497
	Prediction error	3.3%	5.0%	3.0%	3.5%	2.1%
	FE	104	717	1105	1886	3453
	3D-CNN	110	742	1029	1784	3357
	Prediction error	5.8%	3.5%	6.9%	5.4%	2.8%

6. Conclusions

In the new generation of manufacturing systems, machine tools must be able to monitor operations and automatically adjust process parameters during operation. This requires rapid updating of models to determine the best set of process parameters such as cutting speed, depth of cut, and feedrate. This paper proposes a deep learning-based technique for rapid prediction of varying structural parameters of workpieces during turning operations.

In order to generate training data, a large database of machined cylindrical parts and their intermediate steps has been created using an automated CAD generator in Python. Each geometry has been voxelized and represented in the form of a 3D digital matrix. An FE code has been developed in MATLAB to perform eigenvalue analysis and find the first 5 natural frequencies for each geometry. A 3D-CNN architecture has been developed and trained using voxelized CAD data as input and natural frequencies as output. The trained model has been examined against test data with the same geometrical characteristics as training data. A mean absolute percentage error (MAPE) of 1.12% has been obtained over 15,000 test data, indicating the high performance of the trained model in predicting natural frequencies of intermediate machined geometries.

Once the neural network is trained, it can generate rapid predictions for a variety of geometries that fit in the characteristics of the training data. The proposed method also shows the potential to provide a fast FRF prediction tool for real time process monitoring and control of machining operations.

Acknowledgements

This research has been supported financially by the Natural Sciences and Engineering Research Council (NSERC) of Canada and the Canadian Network for Research and Innovation in Machining Technology (CANRIMT II).

References

[1] Budak E, Ozlu E. Analytical modeling of chatter stability in turning and boring operations: a multi-dimensional approach. CIRP Ann Manuf Technol 2007;56:401-404.
[2] Khoshdarregi MR, Altintas Y. Dynamics of multipoint thread turning—part I: general formulation. J Manuf Sci Eng 2018;140:061003.
[3] Tuysuz O, Altintas Y. Frequency domain updating of thin-walled workpiece dynamics using reduced order substructuring method in machining. J Manuf Sci Eng 2017;139:071013.
[4] Thevenot V, Arnaud L, Dessein G, Cazenave–Larroche G. Influence of material removal on the dynamic behavior of thin-walled structures in peripheral milling. Mach Sci Technol 2006;10:275-287.
[5] Altintas Y, Stepan G, Budak E, Schmitz T, Kilic ZM. Chatter stability of machining operations. J Manuf Sci Eng 2020;142:110801.
[6] Zaeh MF, Kleinwort R, Fagerer P, Altintas Y. Automatic tuning of active vibration control systems using inertial actuators. CIRP Ann Manuf Technol 2017;66:365-368.
[7] Munoa J, Beudaert X, Erkorkmaz K, Iglesias A, Barrios A, Zatarain M. Active suppression of structural chatter vibrations using machine drives and accelerometers. CIRP Ann Manuf Technol 2015;64:385-388.
[8] Mañé I, Gagnol V, Bouzgarrou BC, Ray P. Stability-based spindle speed control during flexible workpiece high-speed milling. Int J Mach Tools Manuf 2008;48:184-194.
[9] Song Q, Ai X, Tang W. Prediction of simultaneous dynamic stability limit of time–variable parameters system in thin-walled work- piece high-speed milling processes. Int J Adv Manuf Technol 2011;55:883-889.
[10] Yang Y, Zhang WH, Ma YC, Wan M, Dang XB. An efficient decomposition-condensation method for chatter prediction in milling large-scale thin-walled structures. Mech Syst Signal Process 2019;121:58-76.
[11] Lu Y, Xu X, Wang L. Smart manufacturing process and system automation – a critical review of the standards and envisioned scenarios. J Manuf Syst 2020;56:312-325.
[12] Liang L, Liu M, Martin C, Sun W. A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. J R Soc Interface 2018;15:20170844.
[13] Khadilkar A, Wang J, Rai R. Deep learning–based stress prediction for bottom-up sla 3d printing process. Int J Adv Manuf Tech 2019;102:2555-2569.
[14] Hassan M, Sadek A, Attia MH. A generalized multisensor real-time tool condition–monitoring approach using deep recurrent neural network. Smart Sustainable Manuf Syst 2019;3:20190020.
[15] Franciosa P, Sokolov M, Sinha S, Sun T, Ceglarek D. Deep learning enhanced digital twin for Closed-Loop In-Process quality improvement. CIRP Ann Manuf Technol 2020;69:369-372.
[16] Lin X, Wang X, Li L. Intelligent detection of edge inconsistency for mechanical workpiece by machine vision with deep learning and variable geometry model. Appl Intell 2020;50:2105-2119.
[17] Zhang Z, Jaiswal P, Rai R. FeatureNet: machining feature recognition based on 3d convolution neural network. Comput Aided Des 2018;101:12-22.
[18] Lehoucq RB, Sorensen DC, Yang C. ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods. SIAM; 1998.
[19] Zienkiewicz OC, Taylor RL, Zhu JZ. 2005, The finite element method: its basis and fundamentals. 7th ed. Butterworth-Heinemann: Oxford; 2013.
[20] Ghadai S, Balu A, Sarkar S, Krishnamurthy A. Learning localized features in 3d cad models for manufacturability analysis of drilled holes. Comput Aided Geom Des 2018;62:263-275.
[21] Kohler J, Daneshmand H, Lucchi A, Zhou M, Neymeyr K, Hofmann T. Towards a theoretical understanding of batch normalization. arXiv:1805.10694; 2018.