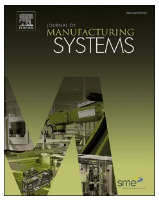
Contents lists available at ScienceDirect

Journal of Manufacturing Systems

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



Accurate prediction of machining cycle times and feedrates with deep neural networks using BiLSTM☆

Shih-Hsuan Chien [a], Burak Sencer [a,*], Robert Ward [b,c]

[a] School of Mechanical, Industrial and Manufacturing Engineering, Oregon State University, Corvallis, OR 97331, USA
[b] Advanced Manufacturing Research Centre, University of Sheffield, Rotherham S60 5TZ, United Kingdom
[c] Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield S1 3JD, United Kingdom

ARTICLE INFO

Keywords:
CNC
Cycle-time
Neural network

ABSTRACT

This paper presents a deep neural network-based approach to predict machining cycle (run) times along complex 3-axis machining part programs. CAD/CAM systems are utilized to generate part programs to machine parts with complex geometries and sculptured surfaces. The part program is processed by the numerical control (NC) unit of the machine tool to plan a feed profile. For such complex part programs, the actual (observed) feed profile and commanded (desired) one are significantly different. This paper uses bi-directional long short-term memory deep networks (bi-LSTM) to model the interpolation behavior of the NC systems to accurately predict how the feedrate profile is planned for any given part program. The G-line length, commanded feed, change in the feed direction through consecutive G-lines are used as a primary set of inputs to the network. The neural network is then trained by running series of toolpaths on the NC system and recording the resultant feedrate profile. Simulation and experimental studies are conducted on NC kernels to validate effectiveness of the proposed strategy. It is shown that machining cycle times on complex toolpaths can be predicted with $>$ 94 % accuracy.

1. Introduction

CAD/CAM systems are utilized to generate part programs, e.g. G-codes or CL files, for machining complex machining toolpaths. Typically, CAM systems are also utilized to simulate the tool motion along the part program and to predict the machining cycle (run) time. The machining cycle time predictions play a crucial role in small job shops producing die-and-moulds or complex aerospace components. In such enterprises, machining cycle time prediction is used for generating part manufacturing cost estimates i.e. the part quotes [1]. Machining cycle time predictions are also used for estimating the machine tool usage, tool-life, operator time, part delivery time commitments, etc. Therefore, it is up most critical for these small industries to come up with accurate prediction of machining cycle times. For mid and large size enterprises, prediction of accurate machining cycle times plays a key role in scheduling the machining processes on a large machine part and optimizing the flow of the components and processes amongst various types of machine tools [2]. It directly affects the efficiency of the production system. A multi-stage production system's efficiency can be improved by simply improving the cycle times of part programs [3]. Therefore, accurately predicting machining cycle times and improving them are vital issues for productivity and commercial success of modern machining industries.

CAM systems are primarily responsible for the prediction of cycle times. However, once the toolpath complexity starts to increase, CAM predictions begin to underestimate actual cycle times [4]. This is due to the fact that most CAM systems predict machining cycle times solely based on the commanded feedrate and the geometric length of the toolpath. For simple toolpaths consisting of long linear sections, the purely geometry-based cycle time prediction is acceptable. However, once the toolpath consists of short-segmented G-lines and sharp contours, the precision suffers greatly. This is due to the fact that CAM systems do not model the interpolation dynamics of modern NC systems [5].

Fig. 1 shows a commercial CNC machine [6], the red circle indicates the NC kernel where the interpolation algorithms are running. Modern NC systems employ complex trajectory generation (interpolation) schemes to generate smooth coordinated multi-axis motion [7–11]. These trajectory generation algorithms override the user commanded (desired) feedrate command and dynamically lower the feedrate based

☆ 51st SME North American Manufacturing Research Conference (NAMRC 51, 2023)
* Corresponding author.
E-mail address: Burak.Sencer@oregonstate.edu (B. Sencer).

https://doi.org/10.1016/j.jmsy.2023.05.020

Available online 31 May 2023

Fig. 1. Commercial CNC machine.

on the part geometry, kinematic (acceleration, jerk) limits of the drives and the dynamics of the machine tool. Such behavior can be easily observed at the junction points of successive G-lines. Machining feedrate is typically lowered at those "corners" as the feed motion changes its direction and transitions from one G-line to the next one [12]. It should be also noted that the time spent during this transition is mainly controlled by the acceleration and jerk limits of the machine and the geometric blending tolerance set by the end-user [13]. These key parameters dictate how much time is spent during these "cornering transients". A realistic example is provided in Fig. 2 where the feedrate profile for a complex toolpath consisting of long and short G-lines is depicted. As shown, the actual feedrate reaches the commanded one only long straight linear portions; whereas, feedrate is significantly lowered along the short segmented contour sections. Please note that the actual cycle time is significantly different than the one predicted by the CAM system as shown in Fig. 2b.

Manufacturing literature has been developing algorithms to accurately predict machining cycle times [14,15]. Most of the approaches have been focused on modeling the NC system behavior using heuristics

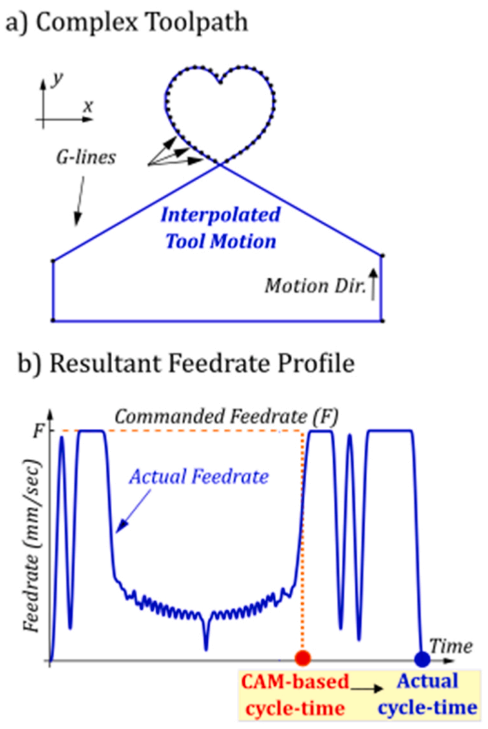


Fig. 2. Continuous interpolation of G-lines.

and based on the first principles. For instance, Chen et al. [16] correlated the time spend during a cornering transition to the angle between successive G-lines, i.e. the cornering angle. Others followed this approach [17,18] and improve over CAM system predictions. Altintas and Tulsyan [19] tackled the problem by modeling the trajectory generation algorithms and the kinematic profile. VCNC software has been developed to predict cycle time, which requires the user to input the acceleration and jerk limits of the kinematic profile and the interpolation tolerance to predict resultant feedrate profile [20]. Although these first principles-based approaches can provide accurate results on simple toolpaths, it fails once the toolpath complexity increases. This is mainly because the algorithms running in NC systems start to alter the kinematic limits on-the-fly and optimize feed profiles to generate faster motion with better surface finish.

Most of these industrial algorithms are proprietary and even kept as trade secrets. Ward et al. [21] and later Sencer [22] considered using artificial neural networks (ANN) to model behavior of the NC system. They have shown that machine learning based approaches are promising for predicting machining cycle times. However, these initial algorithms either required excessive computational time to predict the feedrate profile, or they failed when the complexity of the part program is high, which is usually the case in toolpaths found in aerospace applications.

This paper focuses on the accurate prediction of machining cycle times and in an effort to realize that, it presents a novel bi-directional LSTM (BiLSTM) deep learning network, which learns machine tool's feedrate scheduling and uses that to predict cycle times. The following sections present the network architecture, selection of input features, simulation and experimental results to demonstrate the effectiveness.

2. Cycle time prediction via BiLSTM networks

The first step in modeling NC system's behavior via a neural network (NN) system is to determine the inputs or so-called the "features" that needs to be learnt. In this work, average feedrate along successive G-lines are predicted by the neural network to estimate the cycle time.

2.1. NC system's interpolation behavior

Firstly, the NC system's interpolation behavior is analysed, and key parameters (features) are considered so that the developed neural network can be trained to accurately capture the interpolation behavior and predict average feedrates.

Today's NC systems interpolate toolpaths without fully stopping the tool motion at the end of each G-line (see Fig. 2) but rather decelerating to a lower speed, transitioning to the next G-line and then accelerate back to the user commanded feedrate [7]. Therefore, feed direction of the tool is altered gently without causing any vibrations [8]. Such motion is achieved by continuously and smoothly blending successive G-lines, which elongates machining cycle time based on the kinematic limits of the feed drives and the path geometry.

Depending on the toolpath geometry, NC systems may use one of the two major blending (interpolation) strategies; namely, G-lines are "locally" or "globally" blended to generate a continuous tool motion [23]. Based on the blending strategy, feedrate is reduced differently, and the resultant elongation of the cycle time is dictated. Fig. 3 illustrates local and global interpolation strategies and the corresponding feedrate profiles.

During a "local" blending interpolation behavior, the NC system plans a non-stop tool motion that transitions from one linear G-line to the next one while locally blending (smoothing) the junction point within a geometric blending tolerance ε set by the user (see Fig. 3). Tool's feed direction is changed by θ_k, and to realize a smooth tool motion within the machine tool's kinematic limits, the NC system lowers the machining feed to a blending speed of V_B (see Fig. 3b). Such "local" blends are observed at the junction points of long linear G-lines [24]. Endo and Sencer [22] have identified that key features controlling the

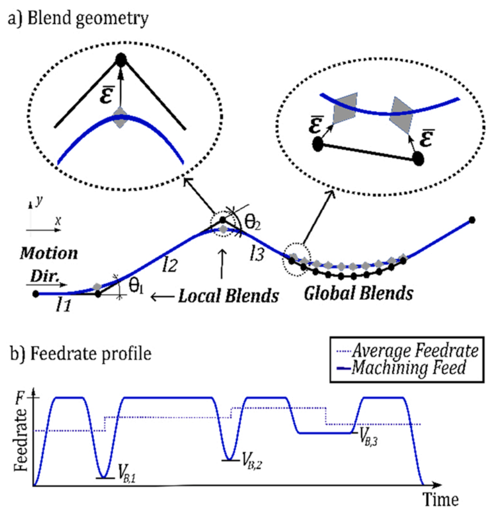


Fig. 3. Non-stop interpolation with local and global blending.

local blending behavior and the time spent during the blending transition are the angle between successive G-lines θ_k, the user commanded feedrate F, and the blending tolerance ε.

Secondly, complex sculptured surfaces consist of smooth curvatures, and CAM systems generate toolpaths by discretizing them into series of very short linear G-lines (see Fig. 3a). For such toolpaths consisting of such short linear commands, the length of a global blend may be longer than the short G-line itself. This is especially observed in aerospace and die-and-mould part programs. In such cases, the NC system may blend "sets of short linear" G-lines jointly using splines [7] or other techniques [25], which help them keep the feedrate high (see Fig. 3b). This strategy is classified as "global" blending interpolation. As a result, global blends occur when G-line lengths are shorter than the blending tolerance, $l_k < \varepsilon$ and successive G-lines have similar changes in the feed direction and linear lengths. Also, it should be noted that the feed drop is not excessive along global blends (see Fig. 3b).

2.2. Design of BiLSTM network and feature extraction

The objective is to develop a NN system to predict the average feedrate (speed) on a G-line considering the effect of local and global blends. If average feedrate value can be predicted accurately, cycle times can then be estimated in a computationally efficient and accurate manner.

When a part program is commanded (or generated) the only knowns (inputs) for the end-user are the toolpath's geometric information and simple interpolation related parameters such as the feedrate F and the interpolation tolerance ε. Therefore, the proposed NN uses these information and related parameters to predict the feedrate and the cycle

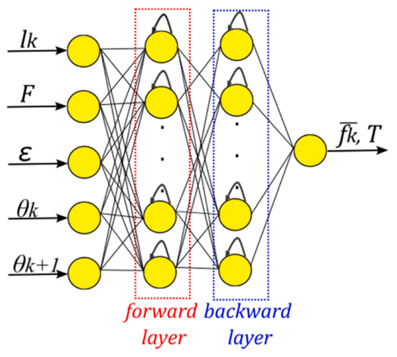


Fig. 4. NN structure for predicting cycle time.

time. Fig. 4 shows the proposed BiLSTM-based NN structure for predicting the average feedrate and the cycle time. Commanded feedrate F, nominal (commanded) length of each G-line l_k with k being the block counter, cornering angle at the start and end of each G-line θ_k and θ_{k+1} (see Fig. 3a), and the blending tolerance ε given by the end-user as the input features for the network. Based on this information, the proposed NN predicts the average feedrate $\overline{f_k}$ for each G-line and overall cycle time (motion time) T.

As indicated in Fig. 4, proposed NN predicts average feedrate (speed) for each G-line to compute how much time the tool would spend on that particular G-line (block). Notice that, due to the non-stop interpolation behavior of the NC system, the actual, i.e. interpolated, tool motion is different from the commanded nominal motion given in the part program (see blue line in Fig. 3a). This length difference in the commanded and actual (interpolated) tool motion must be considered. This difference can be seen in the zoomed view in Fig. 3a as well. Once successive G-lines are blended, length of the interpolated G-line becomes different (shorter) than the original length of the G-line due to the curvature of the blend geometry.

Consider the zoomed view in Fig. 3a for extraction of the average feedrate $\overline{f_k}$. Each commanded G-line point $N_k(x_n, y_n)$ is calculated to find its corresponding junction point $R_k(x_r, y_r)$ with the shortest distance $\overline{\varepsilon}$. The method of obtaining $\overline{\varepsilon}$ works as follows [26]:

Considering the distance vector between the data points in the interpolated tool motion $D_i(x_d, y_d)$ and the commanded G-line points $N_k(x_n, y_n)$,

$$[x_d - x_n]\vec{i} + [y_d - y_n]\vec{j} = r_x\vec{i} + r_y\vec{j} \tag{1}$$

The distance vector between all of the data points $D_i(x_d, y_d)$ and each commanded G-line point $N_k(x_n, y_n)$ are stored in a buffer. The shortest distance for each commanded G-line point $N_k(x_n, y_n)$ is then found within the buffer. The data points D_i that have the shortest distance corresponding to each commanded G-line point N_k are defined as the junction points $R_k(x_r, y_r)$. Once junction points are identified, each blended G-line length $\overline{l_k}$ is found by integrating the machining feed between each junction points from the feedrate profile (see Fig. 3b).

Each blended G-line length $\overline{l_k}$ and the corresponding time $\overline{t_k}$ spent traveling them are used to determine the average $\overline{f_k}$ of the block,

$$\overline{f_k} = \frac{\overline{l_k}}{\overline{t_k}} \tag{2}$$

The cycle time T for a part program is calculated as:

$$T = \sum_k \frac{\overline{l_k}}{\overline{f_k}} \tag{3}$$

Notice that successive corner angles and G-line lengths influence the length of a local blend and the average velocity.

To effectively optimize the feedrate for machining operations, it is crucial to understand how past and future data points relate to the current average velocity. As shown in Fig. 3, several G-lines before and after a global blend can affect the average speed. Conventional artificial neural networks fall short in this task since they only consider information at the current time step, making it challenging to analyze each G-line and how it affects the current average velocity [27]. Recurrent neural networks (RNN) can utilize information from prior inputs to influence the current input and output, but they only consider past information since they flow unidirectionally. To learn the relationship between future data information and current average velocity, a backward layer is necessary. This is where a bi-directional LSTM (BiLSTM) network comes in.

BiLSTM is an effective solution for mimicking the look-ahead strategy used in modern NC systems. These systems consider the previous and next blocks of a part program to optimize the feedrate for machining operations. BiLSTM can scan the part program from both directions, enabling it to process the program sequentially while considering the context of previous and next blocks [28]. This approach allows BiLSTM to learn the dependencies and relationships between different blocks in the part program, which is crucial for optimizing the feedrate. BiLSTM can capture long-term dependencies, consider the context of neighboring blocks, and process the part program sequentially in real-time applications. However, it's essential to avoid overfitting the training data by not having too many hidden neurons.

3. Training protocol

The training process for the BiLSTM network is shown in Fig. 5. After trajectory features are extracted from the training trajectories, the commanded feedrate F, nominal G-line length l_k, cornering angle of the start and end of each G-line θ_k, θ_{k+1}, and blending tolerance ε are used as input for the BiLSTM network. The average feedrate of each G-line is also recorded to be used for training.

The weights and biases of the BiLSTM network are updated for each iteration to decrease the loss function that is defined as the difference between the predicted average feedrate and the measured average feedrate of each G-line. Root Minimum square error (RMSE) is chosen as the loss function and backpropagation is used for weight calculation. After training, the BiLSTM network can then be tested on other trajectories for predicting average feedrate and total cycle time.

3.1. Design of training toolpaths

Training of the BiLSTM network is achieved by running training (test) toolpaths on the machine and recording the resultant feedrate profiles. Two types of training toolpaths are used to train the BiLSTM network.

Firstly, toolpaths that trigger "global" blending are utilized. Fig. 6a depicts an example of global blending training toolpath. Global blending training toolpaths contain features with randomized average cornering angle, $\theta_k \in (0,2pi)$ [rad] and relatively short linear lengths $l_k \in (0,1)$ [mm]. Global training paths are used because CAM systems usually generate smooth curvature toolpaths by discretizing them into series of short linear G-lines.

The second training toolpath are mixed with both globally blended and locally blended G-lines. Fig. 6b depicts an example of mixed training toolpath. Such mixed toolpaths are designed because complex trajectories used in manufacturing can contain both kinds of blended trajectories. The BiLSTM network could learn the behavior between "global" blending and "local" blending interpolated G-lines after training. Each toolpaths contain ~1000 G-lines with random feed direction changes.

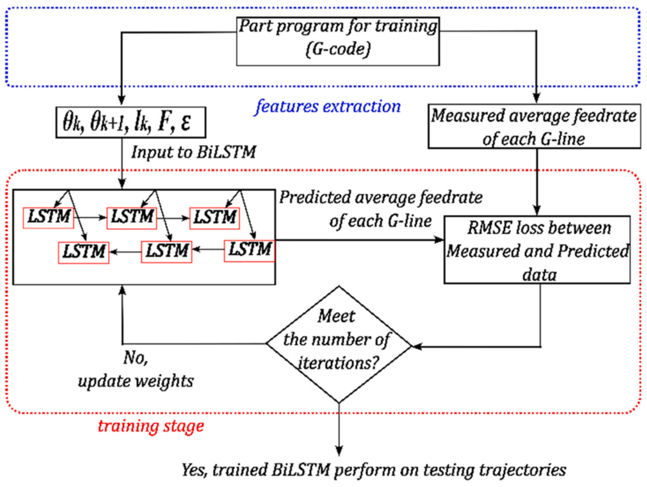

Part program for training
(G-code)
features extraction
$\theta_k, \theta_{k+1}, l_k, F, \varepsilon$
Input to BiLSTM
Measured average feedrate of each G-line
LSTM
Predicted average feedrate of each G-line
RMSE loss between Measured and Predicted data
Meet the number of iterations?
No, update weights
training stage
Yes, trained BiLSTM perform on testing trajectories

Fig. 5. Training process of BiLSTM model.

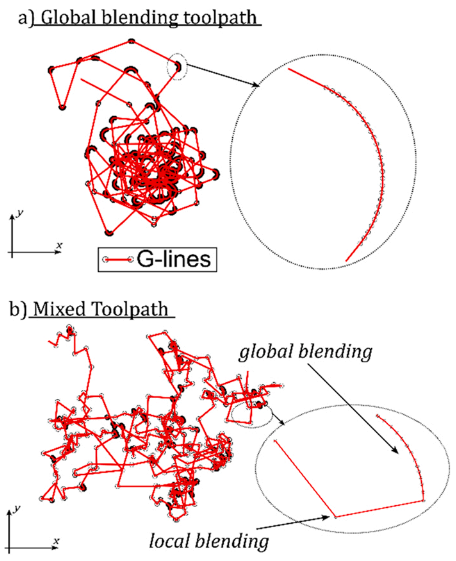


Fig. 6. Training toolpath design.

Local blended G-lines contain random feed direction changes $\theta_k \in (0,2pi)$ [rad] and various lines in the range of $l_k \in (0,1000)$ [mm]. The cornering angle and linear lengths of the global blending toolpaths are set in the same range as the first training toolpath. Local blend geometries accounts for 75 % of these mixed toolpaths, and the other 25 % are global blend geometries.

The nominal G-line length l_k, cornering angle θ_k, are the tuning parameters in designing training paths. All the training toolpaths are executed at various feedrate (F) and blending tolerance (ε) values, which require roughly ~80 simulation part programs each with ~1000 G-lines for training.

4. Results

4.1. Simulation study

This section presents validation of the proposed cycle time prediction approach. A NC kernel simulator that uses a FIR-filtering-based path-smoothing algorithm that blends toolpaths within user-defined error tolerances is utilized. Hayasaka et al. [29] developed a complex interpolation algorithm for high speed machining, and this algorithm is implemented in MATLAB to simulate the behavior of modern NC system. The algorithm is able to capture the complex interpolation behavior with local and global blending capability and look-ahead function.

The BiLSTM network was trained using MATLAB's Deep Learning Toolbox. The network is designed with a hidden layer consisting of ten neurons each in both the forward and backward layers, using the hyperbolic tangent activation function (tanh), followed by a fully connected output layer. The hyperbolic tangent function is selected as the activation function due to its capability to avoid getting trapped in local minima while training, allowing it to learn intricate relationships between inputs and outputs. The use of ten neurons is implemented to avoid complexity in the network and reduce the risk of overfitting during training.

Firstly, 80 sets of training toolpaths (part programs), each with approximately 1000 blend geometries at tolerances of $\varepsilon = 0.3$–0.5 mm are executed at F= 65–100 mm/s speed range for training, which took ~2 h to run. After the BiLSTM network has a satisfactory training result with overall cycle time prediction accuracy of > 98 %, the network is used for predicting cycle time for any testing toolpath that is not included in the training toolpaths.

Fig. 7 shows the tested toolpaths and corresponding predicted and measured average feedrate profiles for $F = 92$ mm/s and blending

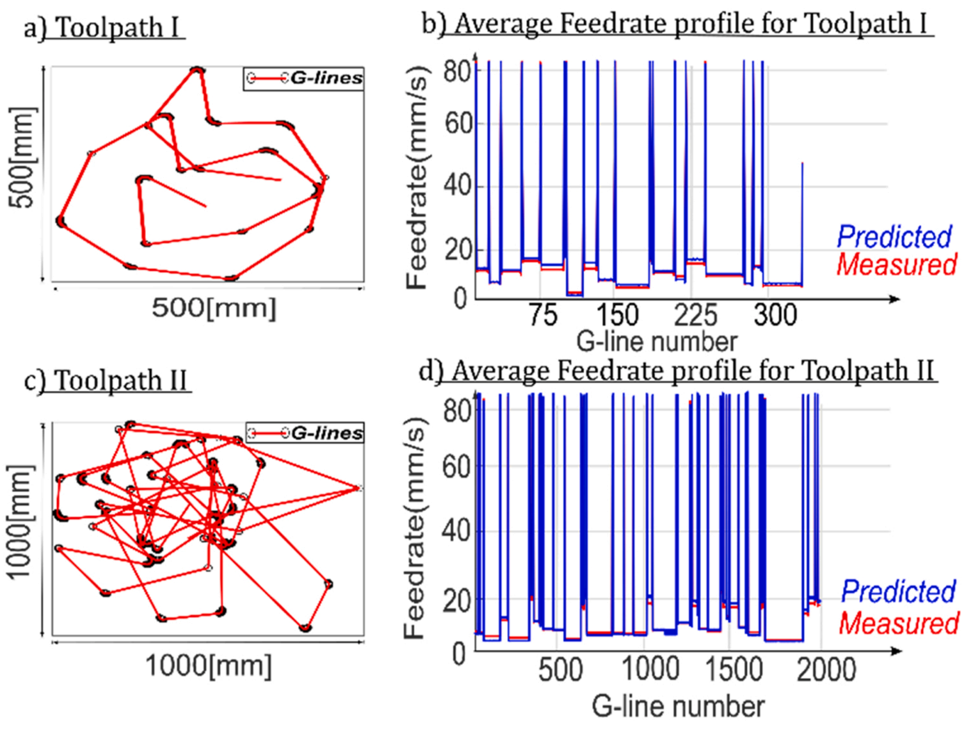


Fig. 7. Feed profile prediction performance along testing toolpaths.

tolerance of $\varepsilon = 0.32$ mm. As shown, the recorded feedrate along these training toolpaths fluctuate greatly. It barely reaches the commanded feedrate due to sharp contours. Therefore, it is considered challenging for CAM systems to predict the cycle time on such a toolpath. The proposed BiLSTM network; however, accurately predicts average feed profiles even when the feedrate drops heavily around local and global corners. Table 1 summarizes the cycle time prediction performances for various feedrate F and blending tolerances ε.

Proposed approach predicts cycle times with overall > 98 % accuracy. In comparison, CAM system underestimates cycle times with an error up to 34 % since it only computes the cycle time based on the user commanded feedrate F and the total geometric length of the part program.

Then, the proposed method is tested on a complex pocketing toolpath that is longer and more complicated as shown in Fig. 8 [22]. The toolpath consists of 5000 G-lines, including various straight lines, local and global blended G-lines. Fig. 9 shows the predicted and measured

Table 1
Cycle time prediction performance for toolpath I & II.

Cycle time prediction performance for toolpath I				
F [mm/s]	ε [mm]	Measured Cycle time [sec]	Prediction (Proposed) [sec]	CAM Prediction [sec]
75	0.3	45.4	44.9 (*err:1.2 %*)	31.3 (*err:29 %*)
	0.4	44.5	45.3 (*err:1.7 %*)	31.5 (*err:29 %*)
92	0.3	39.6	39.9 (*err:0.8 %*)	25.8 (*err:34 %*)
	0.4	39.2	39.7 (*err:1.4 %*)	25.8 (*err:34 %*)
Cycle time prediction performance for toolpath II				
F [mm/s]	ε [mm]	Measured Cycle time [sec]	Prediction (Proposed) [sec]	CAM Prediction [sec]
75	0.3	295	289 (*err:1.5 %*)	213 (*err: 28 %*)
	0.4	292	290 (*err: 1.0 %*)	213 (*err: 27 %*)
92	0.3	256	252(*err: 1.6 %*)	174 (*err: 32 %*)
	0.4	254	251(*err: 1.0 %*)	174 (*err: 31 %*)

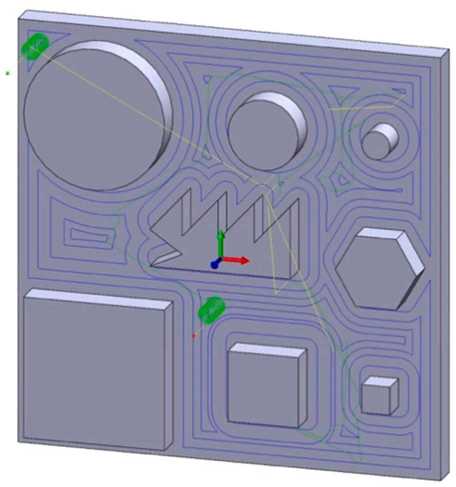


Fig. 8. Pocketing geometry and toolpath.

feedrate profile under commanded feedrate $F = 92$ mm/s and blending tolerance $\varepsilon = 0.32$ mm. It also zooms in the region including G-line number 1700–1900. This section is selected because it captures effect of both local and global blends and severe feed fluctuations. Global blended G-lines have the smallest average feedrate value of 10 mm/s, and the straight lines' have the largest average feedrate of 85 mm/s. Even with such big feedrate difference, the BiLSTM network can still have close estimate of the measured motion (feed) duration around those G-lines.

Table 2 summarizes the prediction results. The proposed approach can predict cycle time with > 97 % accuracy, and has consistent performance over various feedrate and tolerance values. The result is significantly more accurate than the CAM system, which shows a prediction error of > 40 % for this realistic toolpath.

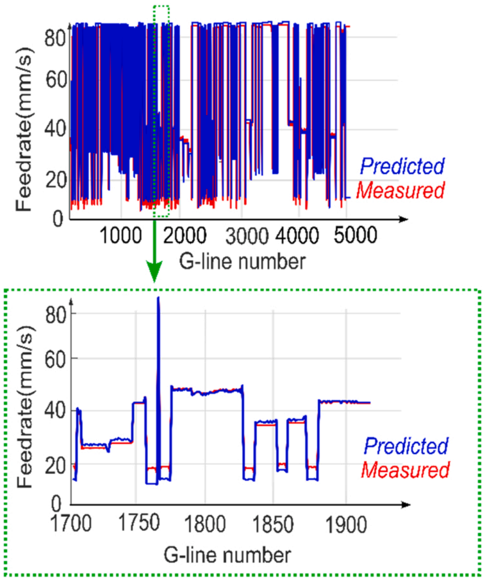


Fig. 9. Feed profile prediction performance along a pocketing toolpath.

Table 2
Cycle time prediction performance for the pocketing toolpath.

F [mm/s]	*ε* [mm]	Measured Cycle time [s]	Prediction (Proposed) [s]	CAM Prediction [s]
75	0.3	273	268 (*err: 1.8 %*)	174 (*err: 36 %*)
	0.4	266	261 (*err: 2.2 %*)	173 (*err: 35 %*)
92	0.3	223	220 (*err: 1.1 %*)	132 (*err: 41 %*)
	0.4	212	209 (*err: 1.5 %*)	125 (*err: 41 %*)

4.2. *Experimental validation*

Fig. 10 shows a commercial DMG-Mori Seiki eVo 40 5-axis CNC machine tool equipped with the Heidenhain TNC640 controller where experiments are carried on. Firstly, 30 training G-codes each with ~2000 blend geometries at $\varepsilon = 50–100$ μm tolerances are executed at F = 60–80 mm/sec speed range for training, which required ~3 h of air cutting time. Afterwards, BiLSTM training required ~4 h to complete. This is followed by testing on the pocketing toolpath shown in Fig. 8. Table 3 summarizes the overall prediction results. The proposed approach can predict cycle time with > 94 % accuracy in experiments, and has consistent performance over various feedrate and tolerance values.

5. Conclusion and future work

This paper presented a data-based approach for average feedrate and machining cycle time prediction for complex machining part programs. A machine learning-based approach using the BiLSTM network is defined, which can capture the NC system behavior and predict machining cycle times with > 94 % accuracy. It is shown that the BiLSTM network is suitable for predicting the average feedrate. The training procedure is practical and straightforward, consisting of executing various training part programs on the machine and using the resulting feedrate profile. The proposed method is expected to be implemented on real CAM systems for accurate feedrate and cycle time prediction. In the future, overall feedrate profile generation methods will also be explored.

Fig. 10. DMG-Mori Seiki eVo 40 5-axis CNC machine.

Table 3
Experimental validation of cycle time prediction for pocketing toolpath.

F [mm/s]	*ε* [mm]	Measured Cycle time [s]	Prediction (Proposed) [s]	CAM Prediction [s]
70	0.075	73.3	69.5 (*err: 5 %*)	63 (*err: 14 %*)
	0.1	71.2	69.1 (*err: 3 %*)	62 (*err: 13 %*)
80	0.075	64.1	60.2 (*err: 6 %*)	55 (*err: 13 %*)
	0.1	63.4	61.1 (*err: 4 %*)	56 (*err: 12 %*)

Declaration of Competing Interest

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

Acknowledgements

The authors gratefully thank Oregon Manufacturing Innovation Centre (OMIC) for funding this project.

References

[1] Silva FJG, Sousa VFC, Pinto AG, Ferreira LP, Pereira T. Build-up an economical tool for machining operations cost estimation. Metals 2022;12(7):1205.
[2] Saez M, Barton K, Maturana F, Tilbury DM. Modeling framework to support decision making and control of manufacturing systems considering the relationship between productivity, reliability, quality, and energy consumption. J Manuf Syst 2022;62:925–38.
[3] Li C, Chang Q. Hybrid feedback and reinforcement learning-based control of machine cycle time for a multi-stage production system. J Manuf Syst 2022;65:351–61.
[4] Siller H, Rodriguez CA, Ahuett H. Cycle time prediction in high-speed milling operations for sculptured surface finishing. J Mater Process Technol 2006;174(1–3):355–62.
[5] Yan X, Shirase K, Hirao M, Yasui T. NC program evaluator for higher machining productivity. Int J Mach Tools Manuf 1999;39(10):1563–73.
[6] Haas, Designed, Built & programmed by haas [Online]. Available: ⟨https://www.haascnc.com/productivity/control.html⟩ [Accessed: 12 November 2022].
[7] Altintas Y, Erkorkmaz K. Feedrate optimization for spline interpolation in high speed machine tools. CIRP Ann 2003;52(1):297–302.
[8] Sencer B, Ishizaki K, Shamoto E. High speed cornering strategy with confined contour error and vibration suppression for CNC machine tools. CIRP Ann 2015;64(1):369–72.
[9] Tajima S, Sencer B. Global tool-path smoothing for CNC machine tools with uninterrupted acceleration. Int J Mach Tools Manuf 2017;121:81–95.
[10] Zhang Q, Li S, Guo J. Smooth time-optimal tool trajectory generation for CNC manufacturing systems. J Manuf Syst 2012;31(3):280–7.
[11] DiMarco C, Ziegert JC, Vermillion C. Exponential and sigmoid-interpolated machining trajectories. J Manuf Syst 2015;37:535–41.
[12] Ward R, Sencer B, Jones B, Ozturk E. Accurate prediction of machining feedrate and cycle times considering interpolator dynamics. Int J Adv Manuf Technol 2021;116(1–2):417–38.
[13] Erkorkmaz K, Heng M. A heuristic feedrate optimization strategy for NURBS toolpaths. CIRP Ann 2008;57(1):407–10.

[14] Yamamoto Y, Aoyama H, Sano N. Development of accurate estimation method of machining time in consideration of characteristics of machine tool. J Adv Mech Des, Syst Manuf, Jpn Soc Mech Eng 2017.
[15] Takizawa H, Aoyama H, Won SC, Rapid estimation of die and mold machining time without NC data by AI based on shape data. In: Proceedings of the 2020 International Symposium on Flexible Automation, American Society of Mechanical Engineers.
[16] Heo EY, Kim DW, Kim BH, Frank Chen F. Estimation of NC machining time using NC block distribution for sculptured surface machining. Robot Comput Integr Manuf 2006;22(5–6):437–46.
[17] So BS, Jung YH, Park JW, Lee DW. Five-axis machining time estimation algorithm based on machine characteristics. J Mater Process Technol 2007;187–188:37–40.
[18] Liu C, Li Y, Wang W, Shen W. A feature-based method for NC machining time estimation. Robot Comput Integr Manuf 2013;29(4):8–14.
[19] Altintas Y, Tulsyan S. Prediction of part machining cycle times via virtual CNC. CIRP Ann 2015;64(1):361–4.
[20] Altintas Y, Kersting P, Biermann D, Budak E, Denkena B, Lazoglu I. Virtual process systems for part machining operations. CIRP Ann 2014;63(2):585–605.
[21] Sun C, Dominguez-Caballero J, Ward R, Ayvar-Soberanis S, Curtis D. Machining cycle time prediction: data-driven modelling of machine tool feedrate behavior with neural networks. Robot Comput Integr Manuf 2022;75(June 2021):102293.
[22] Endo M, Sencer B. Accurate prediction of machining cycle times by data-driven modelling of NC system's interpolation dynamics. CIRP Ann 2022;71(1):405–8.
[23] Tajima S, Sencer B, Shamoto E. Accurate interpolation of machining tool-paths based on FIR filtering. Precis Eng 2018;52(January):332–44.
[24] Tajima S, Sencer B. Accurate real-time interpolation of 5-axis tool-paths with local corner smoothing. Int J Mach Tools Manuf 2019;142:1–15.
[25] Tajima S, Sencer B. Global tool-path smoothing for cnc machine tools with uninterrupted acceleration. Int J Mach Tools Manuf 2017;121:81–95.
[26] Erkorkmaz Kaan, Altintas Yusuf. High speed contouring control algorithm for CNC machine tools. ASME 1998 Int Mech Eng Congr Expo 1998:463–9.
[27] Siami-Namini S, Tavakoli N, Namin AS. The performance of LSTM and BiLSTM in forecasting time series. 2019 IEEE Int Conf Big Data (Big Data) 2019:3285–92.
[28] Pirani, M, Thakkar, P, Jivrani, P, Bohara, MH, and Garg, D, 2022, A comparative analysis of ARIMA, GRU, LSTM and BiLSTM on financial time series forecasting, In: Proceedings of the 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), pp. 1–6.
[29] Hayasaka T, Minoura K, Ishizaki K, Shamoto E, Burak S. A lightweight interpolation algorithm for short-segmented machining tool paths to realize vibration avoidance, high accuracy, and short machining time. Precis Eng 2019;59:1–17.