AN ABSTRACT OF THE THESIS OF

Masafumi Endo for the degree of Master of Science in Robotics presented on August 9, 2021.

Title: Machining Cycle-time Prediction in Combination with Analytical and Data-driven CNC Interpolator Modeling

Abstract approved: ____________________________

Burak Sencer

This work presents a new approach to enable accurate machining cycle-time prediction. While conventional computer-aided manufacturing (CAM) software estimates cycle times for given toolpaths, huge errors occur due to neglecting interpolator dynamics of numerical control (NC) systems. Typically, the machine changes its feedrate such as slowing down and speeding up depending on the existence of junction points between consecutive waypoints to realize smooth motion without violating unacceptable trajectory error. In order to capture and utilize such machine-specific dynamic behavior, a cycle-time prediction framework is proposed in combination with an analytical jerk-limited feedrate planner (JFLP) and data-driven artificial neural networks (ANNs). The high-level interpolation models are first defined as local corner interpolator and global corner interpolator. Then, the cycle-time prediction framework is presented that analyzes geometric characteristics of given toolpaths, classifies types of interpolat-
tion models for every corner, and predicts parametric information by ANNs necessary for JLFP to plan kinematic profiles. In addition, the process to extract ground truth information, which is utilized for training ANNs, based on toolpath geometry and pre-recorded kinematic profiles is explained in detail. In order to validate its performance, several experiments including data collection for training are conducted using the actual CNC machine. Experimental results against realistic toolpaths validate its effectiveness in predicting cycle times accurately.
Machining Cycle-time Prediction in Combination with Analytical and Data-driven CNC Interpolator Modeling

by

Masafumi Endo

A THESIS

submitted to

Oregon State University

in partial fulfillment of the requirements for the degree of

Master of Science

Presented August 9, 2021
Commencement June 2022
Master of Science thesis of Masafumi Endo presented on August 9, 2021.

APPROVED:

________________________________________
Major Professor, representing Robotics

________________________________________
Associate Dean for Graduate Programs, College of Engineering

________________________________________
Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

________________________________________
Masafumi Endo, Author
ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Burak Sencer, for providing guidance and support throughout my studies. His hardworking and enthusiastic approach to his work is inspiring, and it has been a pleasure working with him.

I wish to thank my colleagues at Manufacturing Process Control Laboratory, as well as the faculty and staff of School of Mechanical, Industrial, and Manufacturing Engineering.

Finally, I would like to gratefully acknowledge the support of my family and friends in my graduate studies.
# TABLE OF CONTENTS

1  Introduction .......................................................... 1

2  Literature Review .................................................... 4
   2.1  Introduction ...................................................... 4
   2.2  Model-Based Approach .......................................... 4
   2.3  Data-Driven Approach .......................................... 6

3  Jerk-Limited Feedrate Planning .................................. 8
   3.1  Introduction ...................................................... 8
   3.2  Mathematical Explanation ..................................... 8
   3.3  Implementation .................................................. 13

4  Cycle-Time Prediction using JLFP and ANN Techniques ...... 15
   4.1  Introduction ...................................................... 15
   4.2  Interpolation Models of NC Systems ........................ 16
       4.2.1  Point-to-Point Interpolation ........................... 16
       4.2.2  Local Corner Interpolation ............................. 18
       4.2.3  Global Corner Interpolation ............................ 21
   4.3  Cycle-Time Prediction Framework ............................ 23
       4.3.1  Algorithm Structure ..................................... 23
       4.3.2  Corner Parameter Identification ....................... 37

5  Experiments ........................................................... 44
   5.1  Introduction ...................................................... 44
   5.2  Training Toolpath Design ..................................... 44
   5.3  Experiments using Actual Machine .......................... 47
       5.3.1  Experimental Setup ..................................... 48
       5.3.2  Cycle-Time Prediction against Contour and Trochoidal
               Toolpaths .................................................. 50
       5.3.3  Cycle-Time Prediction against Pocketing Toolpath ... 56
   5.4  Discussion ........................................................ 64

6  Conclusion & Future Work .......................................... 66
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Illustrative example of jerk-limited kinematic profiles.</td>
<td>14</td>
</tr>
<tr>
<td>4.1</td>
<td>Generated trajectory with corresponding feedrate profile by P2P interpolation.</td>
<td>17</td>
</tr>
<tr>
<td>4.2</td>
<td>Generated trajectory with corresponding feedrate profile by local corner interpolation.</td>
<td>18</td>
</tr>
<tr>
<td>4.3</td>
<td>Jerk-limited kinematic profile by JLFP.</td>
<td>19</td>
</tr>
<tr>
<td>4.4</td>
<td>Generated trajectory with corresponding feedrate profile by global corner interpolation.</td>
<td>22</td>
</tr>
<tr>
<td>4.5</td>
<td>Algorithm structure of cycle-time prediction framework.</td>
<td>24</td>
</tr>
<tr>
<td>4.6</td>
<td>Illustrative example of individual corner $C_i$ with its geometric information.</td>
<td>25</td>
</tr>
<tr>
<td>4.7</td>
<td>Illustrative example of circle fitting technique against global corner.</td>
<td>29</td>
</tr>
<tr>
<td>4.8</td>
<td>Prediction sequence of corner merging networks (CM-Nets).</td>
<td>30</td>
</tr>
<tr>
<td>4.9</td>
<td>Prediction sequence of corner feedrate networks (CF-Nets) and kinematic limits networks (KL-Nets).</td>
<td>32</td>
</tr>
<tr>
<td>4.10</td>
<td>Illustrative example of identified corner information based on its geometry and kinematic profile.</td>
<td>39</td>
</tr>
<tr>
<td>5.1</td>
<td>Illustrative example of randomly generated local corner toolpath.</td>
<td>45</td>
</tr>
<tr>
<td>5.2</td>
<td>Illustrative example of randomly generated global corner toolpath.</td>
<td>46</td>
</tr>
<tr>
<td>5.3</td>
<td>DMU 40 eVo linear ©DMG Mori.</td>
<td>47</td>
</tr>
<tr>
<td>5.4</td>
<td>Three toolpaths utilized for evaluating cycle-time prediction performance.</td>
<td>49</td>
</tr>
<tr>
<td>5.5</td>
<td>Classification result of CM-Nets against contour toolpath.</td>
<td>51</td>
</tr>
<tr>
<td>5.6</td>
<td>Measured (black) and predicted (red) kinematic profiles with $F = 3500\text{[mm/min]}$ and $\varepsilon = 10\text{[micron]}$ of contour toolpath.</td>
<td>52</td>
</tr>
<tr>
<td>5.7</td>
<td>Classification result of CM-Nets against trochoidal toolpath.</td>
<td>53</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES (Continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.8</td>
<td>Measured (black) and predicted (red) kinematic profiles with $F = 3500 \text{ [mm/min]}$ and $\varepsilon = 10 \text{ [micron]}$ of trochoidal toolpath.</td>
</tr>
<tr>
<td>5.9</td>
<td>Classification result of CM-Nets against pocketing toolpath.</td>
</tr>
<tr>
<td>5.10</td>
<td>Measured (black) and predicted (red) kinematic profiles with $F = 3500 \text{ [mm/min]}$ and $\varepsilon = 10 \text{ [micron]}$ of pocketing toolpath.</td>
</tr>
<tr>
<td>5.11</td>
<td>Zoomed pocketing toolpath and corresponding feedrate profiles in displacement domain where incorrect prediction occurs (case 1 and case 2).</td>
</tr>
<tr>
<td>5.12</td>
<td>Zoomed pocketing toolpath and corresponding feedrate profiles in displacement domain where incorrect prediction occurs (case 3 and case 4).</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Provided parameters into JLFP</td>
<td>14</td>
</tr>
<tr>
<td>5.1</td>
<td>Specification of toolpaths for network training.</td>
<td>48</td>
</tr>
<tr>
<td>5.2</td>
<td>Number of hidden neurons of ANN sequences.</td>
<td>48</td>
</tr>
<tr>
<td>5.3</td>
<td>Cycle time prediction result against contour toolpath.</td>
<td>50</td>
</tr>
<tr>
<td>5.4</td>
<td>Cycle time prediction result against trochoidal toolpath.</td>
<td>55</td>
</tr>
<tr>
<td>5.5</td>
<td>Cycle time prediction result against pocketing toolpath.</td>
<td>56</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

Advanced manufacturing technology to create industrial products is a foundation of today’s society. Optimization of the industrial processes is one of the big challenges to respond growing need for creating these products. Especially, accurate prediction of part manufacturing times can accelerate to support more robust process planning, estimate part costs accurately, and optimize part programs for higher throughput. However, the current practical methodologies such as utilizing computer-aided manufacturing (CAM) software typically provide inaccurate cycle-time estimations due to the lack of considering computer numerical control (CNC) machine behavior; as a result, industrial companies have been forced to inefficient manufacturing processes.

It is important to realize that numerical control (NC) system plays a role to decide the dynamic motion behavior of CNC machines. The system generates a trajectory for given part-programs, or toolpaths while satisfying user-specified requirements such as desirable feedrate and allowable error tolerance. Although the machine generates trajectory along the toolpath while keeping the commanded feedrate as possible, it is inevitable to slow down its value depending on the geometric characteristics of the toolpath due to the necessity of changing the direction to move without violating the allowable amount of error against reference trajectory. Such frequent modification of feedrate accumulates cycle-time errors between nominal and actually executed time so that revealing the dynamics of NC systems is critical to enhancing accurate cycle-time
In order to accomplish accurate cycle-time prediction by inserting the machine-
dependent dynamic behaviors, several research works have been conducted so far. 
Past works are distinguished as two main approaches: analytical modeling and data-
driven prediction. The former approach tries the prediction based on the analysis of 
the target machine and reasonable assumptions to model its interpolation dynamics. 
One work predicts cycle times by developing the virtual CNC machine that takes 
toolpaths information and outputs corresponding feedrate profiles [1]. Another recent 
work models the corner blending behavior of the target NC systems and kinematic 
profiles are predicted by using finite impulse response (FIR)-based low-pass filters [2]. 
Although these approaches work well against specific machines that are used for these 
researchers, they could not solve the cycle times of other machines, or require machine-
specific turning processes. Instead of the explicit modeling by human experts, the 
latter data-driven approach has the potential to improve the generalization ability 
of cycle-time prediction. One recent work utilizes artificial neural networks (ANNs) 
to predict feedrate value close to every junction point in the provided toolpath and 
estimate cycle times [3]. This research shows the potential of machine-learning-based 
cycle-time prediction, However, the prediction performance of this approach would 
drops depending on the geometric characteristics of given toolpaths, because of the 
high similarity of training and testing toolpaths.

This thesis presents a new cycle-time prediction approach that has generalization 
ability against dynamic behavior of CNC machines and toolpath geometry. The idea of 
the proposed method is to utilize the advantages of both analytical modeling and data-
driven prediction while eliminating their shortcomings. In order to implement this idea, the cycle-time prediction framework is proposed with the combination of a jerk-limited feedrate planner (JLFP) as the analytical model of general machine behavior and ANN sequence as the data-driven prediction for controller-dependent interpolator dynamics. The framework ultimately outputs kinematic profiles with predicted cycle time based on JLFP while utilizing predicted information by ANN sequence as its parameters, which express various interpolation behavior of any machine against given toolpaths. The performance evaluation is conducted against several realistic toolpaths using pre-recorded information obtained from an actual machine, and compared to the conventional CAM-estimated cycle times.

The remainder of this thesis is organized as follows. Chapter 2 reviews literature of cycle-time prediction approaches and related technical methodologies including ANN-based manufacturing research works. Chapter 3 introduces a mathematical explanation of JLFP and shows an illustrative example of the implementation result. Then, in Chapter 4, the cycle-time prediction framework is described in detail, from the definition of analytical models, algorithm structure with the combination of JLFP and ANNs, and identification procedure of ground truth information for training based on pre-recorded kinematic profiles. Experimental results for evaluation the cycle-time prediction performance are shown in Chapter 5. Finally, Chapter 6 concludes this thesis.
Chapter 2: Literature Review

2.1 Introduction

Understanding machine-specific dynamic behavior during trajectory generation is critical to achieving accurate cycle-time prediction. Past research works tackled this challenge by two types of approach in a broad sense: model-based and data-driven cycle-time approaches. This chapter summarizes the literature of modeling interpolator dynamics analytically as well as applying data-driven techniques based on machine learning methods to accelerate sophisticated manufacturing processes.

2.2 Model-Based Approach

Interpolation models of CNC machines have been studied for accomplishing accurate trajectory generation along given toolpaths. The early work proposes jerk-limited trajectory generation with quintic spline interpolation for high-speed CNC systems [4]. Later research works focus on improving the optimality of the machine motion against time [5], smoothing quality of local corner where machine individually interpolates junction points in given toolpaths [6][7], or global corner where several corners are blended while avoiding violation of unacceptable error against reference trajectory [8]. Such trajectory generation algorithms are implemented in CNC machines and they produce manufacturing products based on given toolpaths. Although it is possible
to access these interpolation algorithms in terms of understanding them theoretically, typical CNC machines do not provide specific information of what types of controllers are used, and each NC system has a different turning weight set by industrial companies. Therefore, it is not possible for end-users to clearly understand interpolation models inside machines, and they have to rely on CAM-estimated cycle times, or empirically estimate cycle times. The main issue of CAM-based estimation is that does not consider dynamic behavior that comes from toolpath geometry, but simply calculates cycle times by dividing the total amount of travel by commanded feedrate; thus, accurate estimation of cycle times is not possible especially when given toolpaths are complex.

In order to enhance more accurate cycle-time prediction, several studies try to reveal the dynamic behavior of CNC machines based on analytical modeling and reflect the identified models for the prediction procedure. Altintas et al. present an accurate cycle-time prediction method by explicitly modeling the trajectory generation process including a smoothing strategy and predicting cycle times based on kinematic profiles generated virtually reproduced CNC machines [1]. So et al. establish a cycle-time calculation algorithm by analyzing the operational characteristics of five-axis machines and formulating equations that output estimated cycle times based on geometric information of given toolpaths [9]. Ward et al. recently present feedrate and cycle-time prediction methods by modeling trajectory generation behavior and predicting kinematic profiles based on FIR-based low-pass filters [2]. Each research work shows good prediction performance for the target machines, however, they lack generalization ability against other types of interpolation models without special treatment so that
predicted cycle times can not be reliable to use. Therefore, it is critical to consider the way to improve generalization ability instead of machine-dependent modeling strategy.

2.3 Data-Driven Approach

Data-driven approaches aim to accomplish their objectives such as generating trajectories and predicting cycle times using statistical or machine learning techniques with minimizing human intervention. Heo et al. propose a machining time estimation method by analyzing the distribution of NC blocks [10]. Recent works more focus on applying machine learning methodologies due to the growth of machine learning methods such as deep learning and the ability to access more advanced computational resources [11]. The general strength of machine learning is the ability of modeling complex relationships between information being predicted (output) and being provided into the algorithm (input) without human intervention. Such black-box modeling has impacted many research fields including computer vision, robotics, and manufacturing technologies. Several research works focus on accomplishing accurate trajectory generation for CNC machines using recurrent neural networks (RNNs), which are types of artificial neural networks [12][13][14]. Bhatt et al. proposed a new scheme to reduce trajectory execution errors for industrial manipulators using the sequence of ANNs [15]. Recent research proposed by Li et al. applied a reinforcement learning technique to realize smooth trajectory generation [16]. In addition to the research works so far that aim to trajectory generation, several studies focus on predicting cycle times using machine learning techniques. Yamamoto et al. propose an approach
to directly predict cycle times by ANNs and validate its performance against simple toolpaths [17]. Takizawa et al. recently establish a framework that transfers toolpath shape as image information, then directly predicts cycle times using convolutional neural networks (CNNs) [18]. These methods, however, directly predict cycle times without utilizing knowledge of the dynamic behavior of CNC machines such as kinematic profiles. The most recent research proposes a strategy to predict cycle time by learning the dynamic behavior of the target CNC machine from gathered feedrate profiles [3]. The proposed method first identifies corner feedrate at every junction point, then trains ANNs to predict corner feedrate based on given toolpath geometry. The predicted feedrate is used for calculating cycle times. Although the experimental validation shows accurate cycle-time prediction results against similar toolpaths to training data, only utilizing corner feedrate would not be enough to describe interpolator dynamics applied for generating trajectory along various types of toolpaths with different geometric characteristics.

This research utilizes a data-driven approach to capture the dynamic machine behavior but also applies analytical modeling to provide high-level nature of kinematic profiles. The combination of these two approaches aims to eliminate the disadvantages of each methodology to accomplish accurate cycle-time prediction with generalization ability against any machine and any toolpath composed of different geometry.
3.1 Introduction

Achievement of smooth motion is critical for CNC machines to generate trajectories with time efficiency. Jerk-Limited Feedrate Planning (JLFP) is a fundamental methodology to plan a smooth feedrate motion along the given toolpaths. Its profile starts from the initial feedrate, accelerates to the target feedrate, keeps its value if possible, and slows down to the final feedrate. JLFP repeatedly generates kinematic profiles for every line segment in given toolpaths; hence, it is important to understand its nature for cycle-time prediction. This chapter describes mathematical forms of JLFP and its implementation results.

3.2 Mathematical Explanation

JLFP plans kinematic profiles consisting of seven-time phases $T_i \ (i = 1, \ldots, 7)$. It first increases feedrate during $T_1$ to $T_3$, maintains its value at $T_4$, and decreases feedrate during $T_5$ to $T_7$. These time phases are named acceleration, constant feedrate, and deceleration duration, respectively. Suppose that initial, target, and final feedrate $f^a$, $F$, and $f^e$ are known, and jerk and acceleration limits for acceleration and deceleration duration $J^a$, $A$, $J^d$, and $D$ are provided. The acceleration $a(t)$, feedrate $f(t)$, and displacement $s(t)$ are derived by integrating the jerk $j(t)$ as follows,
The jerk profile for each time phase is written as follows,

$$j(\tau) = \begin{cases} 
J^a, & 0 \leq t < t_1 \\
0, & t_1 \leq t < t_2 \\
-J^a, & t_2 \leq t < t_3 \\
0, & t_3 \leq t < t_4 \\
-J^d, & t_4 \leq t < t_5 \\
0, & t_5 \leq t < t_6 \\
J^d, & t_6 \leq t_7 
\end{cases} \quad (3.2)$$

where $t$ expresses absolute time, $t_1$ to $t_7$ express the time boundaries of each phase. Note that the magnitudes of jerk in each duration are set as same value through this research. The acceleration profile for each time phase is then written by integrating Eq. 3.2 with respect to time as follows,
\[ a(\tau) = \begin{cases} 
J^a\tau_1, & 0 \leq t < t_1 \\
A, & t_1 \leq t < t_2 \\
A - J^a\tau_3, & t_2 \leq t < t_3 \\
0, & t_3 \leq t < t_4 \\
-J^d\tau_5, & t_4 \leq t < t_5 \\
-D, & t_5 \leq t < t_6 \\
-D + J^d\tau_7, & t_6 \leq t \leq t_7 
\end{cases} \]  

(3.3)

where \( \tau \) is the relative time variable that starts at the beginning of the each condition.

The feedrate profile for each time phase is also written by integrating Eq. 3.3 with respect to time as follows,

\[ f(\tau) = \begin{cases} 
\frac{f^s + \frac{1}{2} J^a\tau_1^2}{1}, & 0 \leq t < t_1 \\
f_1 + A\tau_2, & t_1 \leq t < t_2, f_1 = f^s + \frac{1}{2} J^aT_1^2 \\
f_2 + A\tau_3 - \frac{1}{2} J^a\tau_3^2, & t_2 \leq t < t_3, f_2 = f_1 + AT_2 \\
f_3, & t_3 \leq t < t_4, f_3 = f_2 + AT_3 - \frac{1}{2} J^aT_3^2 = F \\
f_4 - \frac{1}{2} J^d\tau_5^2, & t_4 \leq t < t_5, f_4 = f_3 \\
f_5 - D\tau_6, & t_5 \leq t < t_6, f_5 = f_4 - \frac{1}{2} J^dT_5^2 \\
f_6 - D\tau_7 + \frac{1}{2} J^d\tau_7^2, & t_6 \leq t \leq t_7, f_6 = f_5 - DT_6 
\end{cases} \]  

(3.4)
where \( f^s \) and \( F \) denote the initial, and target feedrate that should be satisfied at the end of acceleration duration. In addition, \( f_i \) expresses the feedrate reached at the end of the corresponding time phase. The displacement profile for each time phase is finally written by integrating 3.4 with respect to time as follows,

\[
s(\tau) = \begin{cases} 
  f^s \tau + \frac{1}{6} J^a \tau^3, & 0 \leq \tau < t_1 \\
  s_1 + f_1 \tau_2 + \frac{1}{2} A \tau_2^2, & t_1 \leq \tau < t_2, s_1 = f^s T_1 + \frac{1}{6} J^a T_1^3 \\
  s_2 + f_2 \tau_3 + \frac{1}{2} A \tau_3^2 - \frac{1}{6} J^a \tau_3^3, & t_2 \leq \tau < t_3, s_2 = s_1 + f_1 T_2 + \frac{1}{2} A T_2^2 \\
  s_3 + f_3 \tau_4, & t_3 \leq \tau < t_4, s_3 = s_2 + f_2 T_3 + \frac{1}{2} A T_3^2 - \frac{1}{6} J^a T_3^3 \\
  s_4 + f_4 \tau_5 - \frac{1}{6} J^d \tau_5^3, & t_4 \leq \tau < t_5, s_4 = s_3 + f_3 T_4 \\
  s_5 + f_5 \tau_6 - \frac{1}{2} D \tau_6^2, & t_5 \leq \tau < t_6, s_5 = s_4 + f_4 T_5 - \frac{1}{6} J^d T_5^3 \\
  s_6 + f_6 \tau_7 - \frac{1}{2} D \tau_7^2 + \frac{1}{6} J^d \tau_7^3, & t_6 \leq \tau \leq t_7, s_6 = s_5 + f_5 T_6 - \frac{1}{2} D T_6^2 
\end{cases}
\]  

(3.5)

where \( s_i \) expresses the displacement reached at the end of the corresponding time phase.

The important aspect of JLFP in this research is to analytically determine cycle time by summing all time phases based on mathematical formulations so far. Here, total time phases are summarized as one function \( T(f^s, f^e, J^a, A, J^d, D, L, F) \) as follows,

\[
T(f^s, f^e, J^a, A, J^d, D, L, F) = \sum_{i=1}^{7} T_i 
\]  

(3.6)

where \( L \) denotes the distance at the end of total time phases and \( f^e \) denotes the final
feedrate. Time phases $T_1$, $T_3$, $T_5$, and $T_7$ are derived based on boundary conditions in Eq. 3.3 as follows,

$$
T_1 = T_3 = \frac{A}{J^a^\text{a}} \\
T_5 = T_7 = \frac{D}{J^d^\text{d}}
$$

(3.7)

Then, $T_2$ is formulated based on Eq. 3.4 by considering $F$ is reached from $f^s$ at the end of the acceleration duration as follows,

$$
T_2 = \frac{1}{A} \left( F - f^s - \frac{1}{2} J^a T_1^2 - AT_3 + \frac{1}{2} J^a T_3^2 \right) \\
= \frac{1}{A} (F - f^s - AT_3)
$$

(3.8)

Likewise, $T_6$ is formulated based on Eq. 3.4 by considering $f^e$ is reached from $F$ at the end of deceleration duration as follows,

$$
T_6 = \frac{1}{D} \left( F - f^e - \frac{1}{2} J^d T_5^2 - DT_7 + \frac{1}{2} J^d T_7^2 \right) \\
= \frac{1}{D} (F - f^e - DT_7)
$$

(3.9)

Finally, $T_4$ is derived based on the distance traveled. $L$ is formulated by Eq. 3.5 that is composed of the travel length during acceleration, constant feedrate, and deceleration duration as follows,

$$
L = \frac{F + f^s}{2J^a A} \{ A^2 + (F - f^s)J^a \} + FT_4 + \frac{F + f^e}{2J^d D} \{ D^2 + (F - f^e)J^d \}.
$$

(3.10)
Eq. 3.10 is transformed as follows.

\[
T_4 = \frac{1}{F} \left[ L - \frac{F + f^s}{2J^oA} \left\{ A^2 + (F - f^s)J^o \right\} - \frac{F + f^e}{2J^dD} \left\{ D^2 + (F - f^e)J^d \right\} \right]. \tag{3.11}
\]

Derived equations in Eq. 3.7, 3.8, 3.9, and 3.11 constitute Eq. 3.6 that outputs cycle time based on given parametric information.

3.3 Implementation

According to the formulas explained in Section 3.2, JLFP is implemented for cycle-time prediction. Note that implemented JLFP also equips a way to modify given kinematic limits and commanded feedrate by following procedures described in [4].

Figure 3.1 shows an illustrative example of generated jerk-limited kinematic profiles by the JLFP for a single line segment \( L = 10 \text{[mm]} \) with given parameters shown in Table 3.1. During acceleration and deceleration duration, the trapezoidal acceleration profile can continuously change its value due to the step response of the jerk profile within provided kinematic limits, shown as green-dashed lines. It becomes the parabolic feedrate profile and cubic displacement profile. During constant feedrate duration, the feedrate profile maintain the commanded value represented as the green-dashed line. Finally, the displacement profile reaches 10 [mm] at the end of the deceleration duration. Cycle time can be defined the time when the feedrate planning process is completed along the given length of the line segment.
Figure 3.1: Illustrative example of jerk-limited kinematic profiles.

Table 3.1: Provided parameters into JLFP

<table>
<thead>
<tr>
<th>$F$ [mm/sec]</th>
<th>$f^a$ [mm/sec]</th>
<th>$f^e$ [mm/sec]</th>
<th>$A, D$ [mm/sec$^2$]</th>
<th>$J^a, J^d$ [mm/sec$^4$]</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0</td>
<td>0</td>
<td>1000</td>
<td>25000</td>
</tr>
</tbody>
</table>
Chapter 4: Cycle-Time Prediction using JLFP and ANN Techniques

4.1 Introduction

Accurate machining cycle-time prediction is challenging due to controller-dependent interpolation models that generate trajectories of given toolpaths. Such models construct kinematic profiles based on user-specified parameters with ambiguous weights such as jerk and acceleration limits, which are not clarified for end-users. Although cycle times can be roughly estimated by dividing the length of reference toolpaths by the commanded feedrate, the complex geometry provokes frequent changes of feedrate for changing their direction to move by slowing down and speeding up when necessary. As a result, there are non-negligible gaps between nominal and actual cycle times. In order to more accurately predict cycle times, it is essential to reveal such acceleration and deceleration behavior of interpolation models. This research proposes a machine-learning-based cycle-time prediction framework that indirectly captures interpolation models, learns quantitative information, and utilizes learned knowledge for cycle-time prediction. The key idea of this framework is how to clarify the interpolator dynamics from the pre-recorded information, such as kinematic profiles, and to apply them for cycle-time prediction. The framework equips interpolator dynamics prediction sequence composed of artificial neural networks (ANNs) that learn automatically identified interpolation models of target machines as quantitative
information. This chapter carefully discusses the definition of interpolation models, the overall framework including algorithm structures and automatic ground truth identification procedure for learning, and its performance against several simulators and an actual machine.

4.2 Interpolation Models of NC Systems

CNC machines equip various interpolation models to generate trajectory with the desired motion by following given waypoints while satisfying commanded feedrate and avoiding violation of commanded error tolerance. Such models enable to plan of continuous kinematic profiles; therefore, understanding the characteristics is critical to predicting cycle time by measuring the time domain of the kinematic profiles. However, equipped controllers are typically black-boxed, and also detailed modeling procedure lacks generalization ability against other controllers. Hence, a reasonable definition of generalized interpolation models is necessary. This section introduces three generalized interpolation models that would be used for any controllers in a broad perspective.

4.2.1 Point-to-Point Interpolation

The point-to-point (P2P) interpolation is a fundamental motion for trajectory generation. The machine generates trajectory by strictly following given toolpaths composed of discrete waypoints within user-specified error tolerance. As a result, the machine
produces trajectory that precisely matches reference toolpaths, as shown in Fig. 4.1. The disadvantage of P2P interpolation is that the machine must stop every junction point, or corner to change its direction to move. Although the machine keeps the commanded feedrate $F$ as possible to accelerate its productivity, the accumulation of full-stop motion due to numerous number of given toolpaths causes significant time delay. Therefore, it is necessary to realize non-stop motion around corner while keeping precision of trajectory as possible. The later two interpolation models tackle this problem: non-stop motion and precise corner trajectory within required quality.
4.2.2 Local Corner Interpolation

Local corner interpolation aims to accomplish continuous motion by allowing certain errors at every corner, unlike P2P interpolation. It slows down and speeds up to keep target feedrate as possible, as shown in Fig. 4.2 (a). It should be common in any machine to generate the desired trajectory since maintaining feedrate at the corner would produce significant error between nominal and actual toolpaths. In addition, sudden changes of feedrate cause huge jerk and acceleration; as a result, the machine would experience unacceptable fluctuation during trajectory generation. In order to avoid such situations, any controllers equip the way to reasonably translate from
Figure 4.3: Jerk-limited kinematic profile by JLFP.

the target feedrate to lower one, and vice versa. Hence, local corner interpolation is defined as one of the generalized models for the cycle-time prediction framework.

Fig. 4.3 shows that its kinematic profiles consist of acceleration, constant feedrate, and deceleration phase to generate a trajectory for one line segment. In order to plan such profiles using a jerk-limited feedrate planner (JLFP), following parameters should be reasonably identified from pre-recorded kinematic profiles.

- Initial feedrate $f^s$ and final feedrate $f^e$
- Jerk limit $J^a$ and acceleration limit $A$ for acceleration duration
- Jerk limit $J^d$ and acceleration limit $D$ for deceleration duration

These parameters embrace characteristics of machine-dependent interpolation dynamics as quantitative values, such as how amount drops down its feedrate around the corner and when to start slowing down or speeding up to achieve smooth motion along with toolpaths. Hence, ANNs are applied to predict these parameters by taking inputs that can be obtained before executing the machining process.

Another important characteristic is that the machine can eliminate to decrease its feedrate, as shown in Fig. 4.2 (b). It keeps the target feedrate as possible depending on the characteristics of provided toolpaths, commanded values such as error tolerance and target feedrate, and interpolation algorithms implemented in controllers. For example, Fig. 4.2 (b) shows the machine skips to decrease its feedrate due to the obtuseness of the cornering angle. This motion behavior is notable since the judgment of merging of successive line segments affects cycle-time prediction by producing a difference of feedrate profile as shown in Fig. 4.2 (a) and (b). Corner merging behavior is considered to be different from typical local corner interpolation due to toolpath geometry and interpolation algorithms; thus, it is essential to automatically classify which corner should be merged or not, in addition to predicting parameters for JLFP. The decision making of classifying merged corner is executed ANN classifier, then parameters for JLFP are predicted.
4.2.3 Global Corner Interpolation

Although previous sections describe interpolation models for the junction point between two successive line segments, merging more than two line segments to accomplish smooth motion is possible. For example, as shown in Fig. 4.4 (a), the machine generates a trajectory to follow given waypoints composed of long straight lines and a circular shape portion with multiple short line segments. While the circular shape portion has numerous short straight lines, where usually P2P or local corner interpolation are executed, the planned feedrate profile would drop its value only once and keep the condition until passing through the circular portion, as shown in Fig. 4.4 (a). This motion is defined as global corner interpolation since the model regards a circular shape with multiple line segments as one wide-ranging corner to avoid generating fluctuation of kinematic profiles. Note that such toolpath to construct a circular shape with multiple short line segments is broadly applied for part programs; thus, the global corner interpolation is remarkable to distinguish from P2P and local corner interpolations.

Global corner interpolation can be viewed as the extension of P2P and local corner interpolation, so the main foci for cycle-time prediction are similar to the previous two interpolation models, listed as follows.

- Parametric information controlling acceleration and deceleration behavior around the global corner
- Decision making of which global corner should be merged or not

The former denotes that it is necessary to predict parameters for JLFP, same as local
corner interpolation, except that the corner is composed of multiple line segments. Hence, ANNs are applied to predict these parameters while input information would be different from the other two interpolation models due to toolpaths characteristics. The latter expresses that the merging behavior of global corners should also be identified and learned for cycle-time prediction. Suppose that given toolpaths contain global corners consisting of multiple line segments with obtuse cornering angles. As shown in Fig. 4.4 (b), the machine would maintain its feedrate through the corners, depending on commanded information as target feedrate, error tolerance, and interpolation algorithms. The different decision of merging causes cycle-time gaps as shown in Fig. 4.4 (a) and (b); thus, the classification process is vital to predicting cycle-time.

Figure 4.4: Generated trajectory with corresponding feedrate profile by global corner interpolation.

(a) non-merged global corner  
(b) merged global corner
4.3 Cycle-Time Prediction Framework

Revealing controller-dependent interpolation dynamics against given toolpaths is necessary to predict machining cycle-time before executing trajectory generation. This research proposes the machine-learning-based cycle-time prediction framework that combines JLFP and ANN techniques to tackle this challenge. The objective of the proposed framework is to generate jerk-limited kinematic profiles using JLFP and measure its time domain to predict cycle time for given toolpaths. In order to install controller-dependent interpolation models into JLFP, the framework applies ANNs for predicting parameters of JLFP, such as corners being merged or not, cornering feedrate, and kinematic limits. ANN sequence takes inputs that can be obtained before trajectory generation, such as geometric information of toolpaths and user-specified parameters during their training and prediction phase. The corresponding outputs, which capture machine-specific interpolator dynamics, are automatically identified based on the pre-recorded kinematic profiles and learned their correspondence against input information for prediction. This section describes the details of the proposed framework step by step, including its motivation, challenges, and technical approach.

4.3.1 Algorithm Structure

The algorithm structure of the proposed framework is shown in Fig. 4.5. The framework takes inputs that can be accessed before trajectory generation and outputs cycle times that can be measured from kinematic profiles by JLFP with predicted parameters by ANNs. It consists of four main processes.
4.3.1.1 Geometric Analysis

The first step is called geometric analysis. The geometric analysis of given toolpaths aims to identify two corners: local and global corners. It is necessary to distinguish types of corners based on geometric characteristics since interpolator dynamics are independently predicted against each type of corner. For local corners, local corner interpolations is executed by dropping down its feedrate at the individual junction point between two line segments when necessary. In contrast, global corner interpolation generates trajectory for global corners by looking ahead at multiple line segments that make the circular-type shape. It lowers its feedrate when it enters such a wide-ranging corner and maintains its feedrate until passing through the corner to avoid fluctuation of kinematic profiles.

The technical approach of geometric analysis is processed for every junction point between consecutive line segments. The input information of the procedure is toolpaths with $M$ waypoints, and the output is identified $N$ corners registered as local or global ones. Note that the number of identified corners can be different from the number of total junction points of given toolpaths, since one global corner consists of at least two
Figure 4.6: Illustrative example of individual corner $C_i$ with its geometric information.

line segments or three local corners. An individual corner $C_i$ ($i = 1, \ldots, M-1$) consists of three consecutive waypoints as shown in Fig. 4.6. Its mathematical formulation is expressed as follows,

$$C_i = [P_{i-1}, P_i, P_{i+1}], \quad (4.1)$$

where $P$ expresses $x$- and $y$-axis positional information of the waypoints as $P = [x^p, y^p]^T$. Suppose at $C_i$, its cornering angle $\theta_i$ and line length $l_i$ are calculated as follows,

$$\theta_i = \arccos \left\{ \frac{(P_{i+1} - P_i) \cdot (P_i - P_{i-1})}{|P_{i+1} - P_i||P_i - P_{i-1}|} \right\}, \quad (4.2)$$

$$l_i = |P_{i+1} - P_i|. \quad (4.3)$$
The local corner is identified when the previous and current geometric characteristics at corners are sufficiently different in terms of the cornering angle and line length as follows.

\[
\begin{align*}
\theta_i &\neq \theta_{i-1} \\
l_i &\neq l_{i-1}
\end{align*}
\]  

(4.4)

On the other hand, when both of the following conditions are satisfied within user-specified thresholds, the corner is regarded as one of the components that consist of the global corner.

\[
\begin{align*}
\theta_i &\simeq \theta_{i-1} \\
l_i &\simeq l_{i-1}
\end{align*}
\]  

(4.5)

The above comparisons are repeatedly conducted against successive corners until the conditions are not satisfied. As a result, the geometric analysis identifies the local and global corners \( C_j \) \((j = 1, \ldots, N)\) with the corresponding position of waypoints as follows.

\[
C_j = \begin{cases} 
C^l_j = [P_{k-1}, P_k, P_{k+1}] \\
C^g_j = [P_{l-1}, \underbrace{P_l, \ldots, P_{l+m}}_{\text{cornering portion}}, P_{l+m+1}] 
\end{cases}
\]  

(4.6)

where \( C^l_j \) and \( C^g_j \) express the position of the junction point identified as the local corner, and the series of the position of the junction points identified as the global corner, respectively. Note that \( C^g_j \) consists of \( m \) line segments and the first and last
waypoints are recognized as straight line segments in the individual corner.

4.3.1.2 Feature Extraction

After geometric analysis, input feature information is prepared based on user-specified parameters into machines as commanded features and toolpaths geometry as geometric features. Commanded features are composed of the target feedrate $F$ and error tolerance $\varepsilon$. Note that commanded features are provided for all networks regardless of the types of corners since they control relative changes of interpolation models. Geometric features are extracted based on the identified local and global corners in Eq. 4.6. For local corners, the cornering angle $\theta_k$ and line lengths $l_{k-1}$ and $l_k$ are calculated using Eq. 4.2 and Eq. 4.3, which express individual corner information at $C_j^l$. For global corners, the average cornering angle $\bar{\theta}_j$ and line length $\bar{l}_j$ of series of corners at $C_j^g$ are extracted as follows,

$$\bar{\theta}_j = \frac{1}{m} \sum_{g=0}^{m-1} \theta_{l+g},$$  \hspace{0.5cm} (4.7)

$$\bar{l}_j = \frac{1}{m} \sum_{g=0}^{m-1} l_{l+g}.$$  \hspace{0.5cm} (4.8)

In addition, the curvature $\kappa_j$ is estimated by the least-squares method to use it as a geometric feature. Suppose the series of individual corners at $C_j^g$ is fitted by the circular shape where its center is at $(x_j^c, y_j^c)$ with radius $r_j$. The sum of the squared
difference between \( C_j^g \) and the circle is expressed as follows,

\[
\sum_{g=0}^{m-1} \left[ \left( x_{l+g}^p - x_j^c \right)^2 + \left( y_{l+g}^p - y_j^c \right)^2 - r_j^2 \right] = 0. \tag{4.9}
\]

Eq. 4.9 is transformed to reduce unknown variables as follows,

\[
\sum_{g=0}^{m-1} \left[ x_{l+g}^p \right]^2 + \left( y_{l+g}^p \right)^2 + \alpha x_{l+g}^p + \beta y_{l+g}^p + \gamma \right] = 0, \tag{4.10}
\]

where the variables \( \alpha, \beta, \) and \( \gamma \) are expressed as follows,

\[
\begin{align*}
\alpha &= -2x_j^c \\
\beta &= -2y_j^c \\
\gamma &= x_j^{c2} + y_j^{c2} - r_j^2
\end{align*} \tag{4.11}
\]

Partial derivatives of Eq. 4.10 are derived for \( \alpha, \beta, \) and \( \gamma \) as follows,

\[
\begin{align*}
\frac{\partial}{\partial \alpha} &= \alpha \sum_g x_{l+g}^p + \beta \sum_g x_{l+g}^p y_{l+g}^p + \gamma \sum_g y_{l+g}^p + \sum_g \left( x_{l+g}^p + y_{l+g}^p \right) = 0 \\
\frac{\partial}{\partial \beta} &= \alpha \sum_g x_{l+g}^p y_{l+g}^p + \beta \sum_g y_{l+g}^p + \gamma \sum_g y_{l+g}^p + \sum_g \left( x_{l+g}^p y_{l+g}^p + y_{l+g}^p \right) = 0 \\
\frac{\partial}{\partial \gamma} &= \alpha \sum_g x_{l+g}^p + \beta \sum_g y_{l+g}^p + \gamma \sum_g \left( x_{l+g}^p + y_{l+g}^p \right) = 0
\end{align*} \tag{4.12}
\]
Eq. 4.12 is solved in a matrix form to derive \((x^c_j, y^c_j)\) and \(r_j\) (Fig. 4.7) as follows,

\[
\begin{bmatrix}
\alpha \\
\beta \\
\gamma
\end{bmatrix}
= \begin{bmatrix}
\sum_g x^p_{l+g} \sum_g y^p_{l+g} \sum_g x^p_{l+g} \\
\sum_g x^p_{l+g} y^p_{l+g} \sum_g y^p_{l+g} \sum_g y^p_{l+g} \\
\sum_g x^p_{l+g} \sum_g y^p_{l+g} \sum_g 1
\end{bmatrix}^{-1}
\begin{bmatrix}
-\sum_g \left( x^p_{l+g} \right)^3 + x^p_{l+g} y^p_{l+g} \sum_g \left( x^p_{l+g} \right)^2 \\
-\sum_g \left( x^p_{l+g} y^p_{l+g} + y^p_{l+g} \right)^3 \\
-\sum_g \left( x^p_{l+g} \right)^2 + y^p_{l+g} \sum_g \left( x^p_{l+g} \right)^2
\end{bmatrix}.
\tag{4.13}
\]

Based on \(r_j\) solved by Eq. 4.11 and Eq. 4.13, \(\kappa_j\) is finally derived as follows,

\[
\kappa_j = \frac{1}{r_j}.
\tag{4.14}
\]

4.3.1.3 Interpolator Dynamics Prediction Sequence

These extracted features are then provided for interpolator dynamics prediction sequence based on ANNs. The prediction sequence is two-folded: corner merging networks and corner parameter prediction networks.
Figure 4.8: Prediction sequence of corner merging networks (CM-Nets).

The former networks, corner merging networks (CM-Nets), aim to classify which corner should be merged or not. In other words, CM-Nets sort out corners where parameters for JLFP are predicted in the next step for dropping down its feedrate, or corners where machines keep their target feedrate as if they were smoothing a straight line segment. CM-Nets play a role in estimating what types of interpolation models described in Section 4.2 are applied for every corner. The algorithmic structure of CM-Nets is shown in Fig. 4.8. The sequence equips two ANNs for each corner type. Based on the geometric analysis and feature extraction procedure, input features of local and global corners are provided to corresponding networks. Local CM-Net takes a vector of input features at $C_j^l$, besides global CM-Net takes a vector of input features at $C_j^g$ as follows,
The output of target corners expressed as Eq. 4.15 is obtained from the output layer of networks, which is composed of the following sigmoid function,

\[
\tilde{y}_j = \frac{1}{1 + \exp\left(-\sum_h w_h z_h\right)},
\]

where \(w_h\) and \(z_h\) denote the weight and output variable obtained from the hidden neuron, respectively. Two distinct classes are predicted as corners are merged or not based on the outcome of the sigmoid function \(\tilde{y}_j\) that is regarded as a probability of being in each class as follows,

\[
\tilde{C}_j = \begin{cases} 
\tilde{C}_j^\text{merged} & \text{if } 0 \leq \tilde{y}_j < 0.5 \\
\tilde{C}_j^\text{non-merged} & \text{if } 0.5 \leq \tilde{y}_j \leq 1 
\end{cases}
\]

The latter networks, corner parameter prediction networks, aim to predict parameters of JLFP for corners where machines slow down and speed up to follow given toolpaths while subjecting to the pre-specified error tolerance. The algorithmic structure of corner parameter prediction networks is shown in Fig. 4.9. The network sequence equips corner feedrate networks (CF-Nets) that predict minimum feedrate, and kinematic limits networks (KL-Nets) that predict jerk and acceleration limits around corners. By combining prediction results from CF-Nets and KL-Nets, the network sequence estimates the dynamic behavior of machines along given toolpaths.
with complex geometry as quantitative information.

Recall that geometric features of local and global corners are prepared in the feature extraction step, then corner merging behaviors are predicted by CM-Nets. Thus, corner parameter prediction networks can predict target parameters by taking inputs at corners where CM-Nets classify as non-merged corners $\tilde{C}_{\text{non-merged}}^n$ ($n = 1, \ldots, O$) that consist of $\tilde{C}_{\text{d,non-merged}}^n$ and $\tilde{C}_{\text{g,non-merged}}^n$. By using such extracted information, the corner parameter prediction step begins with corner feedrate prediction by CF-Nets. Local CF-Net takes a vector of input features at $\tilde{C}_{\text{d,non-merged}}^j$, besides global CF-Net takes a vector of input features at $\tilde{C}_{\text{g,non-merged}}^j$ as follows,
The output of target corners is obtained from the output layer of networks, which is composed of the following identity function,

\[ \tilde{f}_c^n = \sum_h w_h z_h, \]  

(4.19)

where \( \tilde{f}_c^n \) expresses predicted corner feedrate. Then, KL-Nets predict kinematic limits for the corresponding corners. Local KL-Net takes a vector of input features at \( \tilde{C}^{l,\text{non-merged}}_j \), besides global CF-Net takes a vector of input features at \( \tilde{C}^{g,\text{non-merged}}_j \) as follows,

\[ \mathbf{x}_n = \begin{cases} 
[\theta_k, l_{k-1}, l_k, F, \varepsilon] \mathbf{T} & \text{if } \tilde{C}^{l,\text{non-merged}}_j, \\
[\bar{\theta}_j, \bar{I}_j, \kappa_j, F, \varepsilon] \mathbf{T} & \text{if } \tilde{C}^{g,\text{non-merged}}_j.
\end{cases} \]  

(4.20)

In Eq. 4.20, \( \tilde{f}_c^n \) is subtracted from \( F \) to indicate the difference between the minimum and target feedrate, since kinematic limits are used for changing feedrate of machines to fill the gap in feedrate profile. The output vectors of kinematic limits are obtained from the output layer of networks, which is composed of the following identity function,

\[ \begin{bmatrix} \tilde{J}^{d,n-1}_k, \tilde{D}^{\text{an}}_n, \tilde{J}^{\text{an}}_n, \tilde{A}_n \end{bmatrix} \mathbf{T} = \mathbf{W} \mathbf{z}, \]  

(4.21)

where \( \tilde{J}^{d,n-1}_k, \tilde{D}^{\text{an}}_n, \tilde{J}^{\text{an}}_n, \) and \( \tilde{A}_n \) denote jerk and deceleration limits before entering the target corners and jerk and acceleration limits after passing the target corners,
respectively. The output layer owns multiple neurons so that the weights and output variables obtained from the hidden layer are expressed as a matrix form $W$ and a vector form $z$, respectively.

4.3.1.4 Feedrate Planning

The final step for cycle-time prediction is feedrate planning by JLFP with analyzed and predicted information in previous steps. While the kinematic profiles are typically generated for a single line segment that connects two consecutive waypoints, the implemented feedrate planner in the framework utilized processed geometric information to reproduce interpolator dynamics with predicted information. The planner repeatedly generates kinematic profiles for traveling distance between non-merged local or global corners, from $\tilde{C}_n^{\text{non-merged}}$ to $\tilde{C}_{n+1}^{\text{non-merged}}$.

The technical approach of integrating JLFP and predicted information starts from the registration of predicted corner parameters from CF-Nets and KL-Nets. Initial feedrate $f^s$ and final feedrate $f^e$ are applied as follows,

$$\left\{ \begin{array}{l} f^s = \tilde{f}_n^c \\ f^e = \tilde{f}_{n+1}^c \end{array} \right. \quad (4.22)$$

Also, jerk and acceleration limits of acceleration and deceleration duration are applied.
as follows,

\[
\begin{align*}
J^a &= \tilde{J}^a_n \\
A &= \tilde{A}_n \\
J^d &= \tilde{J}^d_n \\
D &= \tilde{D}_n
\end{align*}
\]

These six parameters are the regular information regardless of local and global corners. After parameter registration, the travel distance \( L \) from \( \tilde{C}^\text{non-merged}_n \) to \( \tilde{C}^\text{non-merged}_{n+1} \) for feedrate planning is calculated. First, the travel distance \( L^s \) where JLFP generate kinematic profiles as dealing with straight lines within seven-time duration \( T_q \) \((q = 1, \ldots, 7)\) is calculated as sum of total lengths of line segments between predicted two consecutive local or global corners, from \( \tilde{C}^\text{non-merged}_n \) to \( \tilde{C}^\text{non-merged}_{n+1} \), as one single line segment to reproduce the corner merging behavior of target machines. In addition, the framework considers lengths of global corners to simulate constant corner feedrate duration defined in Section 4.2.3. Remember that the global corner interpolation is expressed as the behavior that drops down its feedrate before entering global corners, keeps the value within the corners, and speeds up after passing through the corners. Hence, it is necessary to reflect such a travel distance where machines keep lower feedrate, unlike the behavior at local corners. The travel distance \( L^c \) for constant corner feedrate duration is formulated as follows,
\[
L^c = \begin{cases} 
0 & \text{if } \tilde{C}_{n}^\text{non-merged} \in \tilde{C}^l \\
\sum_{g=0}^{m} l_{l+g} & \text{if } \tilde{C}_{n}^\text{non-merged} \in \tilde{C}^g 
\end{cases} \quad (4.24)
\]

The starting corner \( \tilde{C}_{n}^\text{non-merged} \) has the travel distance before acceleration duration \( L_{c,a} \) while the ending corner \( \tilde{C}_{n+1}^\text{non-merged} \) has the travel distance after deceleration duration \( L_{c,d} \). Two travel distances are expressed as follows, under the assumption that the transition occurs at the midpoint of corners,

\[
L_{c,a}^c = \begin{cases} 
0 & \text{if } \tilde{C}_{n}^\text{non-merged} \in \tilde{C}^l \\
\frac{L_{n}}{2} & \text{if } \tilde{C}_{n}^\text{non-merged} \in \tilde{C}^g 
\end{cases} \quad (4.25)
\]

\[
L_{c,d}^c = \begin{cases} 
0 & \text{if } \tilde{C}_{n+1}^\text{non-merged} \in \tilde{C}^l \\
\frac{L_{n+1}}{2} & \text{if } \tilde{C}_{n+1}^\text{non-merged} \in \tilde{C}^g 
\end{cases} \quad (4.26)
\]

As a result, the input travel distance \( L \) for feedrate planning is expressed as follows,

\[
L = L_{c,a}^c + L^s + L_{c,d}^c. \quad (4.27)
\]

Based on \( L \) and acquired parameters in Eq. 4.22 and Eq. 4.23, JLFP generates kinematic profiles following mathematical formulas explained in Chapter 3. Therefore,
the predicted cycle time against given toolpaths $\tilde{T}_{\text{total}}$ is expressed as follows,

$$\tilde{T}_{\text{total}} = \sum_{n=1}^{O} \sum_{q=0}^{8} \tilde{T}_{q,n},$$

(4.28)

where $\tilde{T}_{0}$, $\tilde{T}_{1}$ to $\tilde{T}_{7}$, and $\tilde{T}_{8}$ from $\tilde{C}_{n}^{\text{non-merged}}$ to $\tilde{C}_{n+1}^{\text{non-merged}}$ are derived as follows,

$$\begin{align*}
\tilde{T}_{0} &= \frac{L_{c,a}}{f_{s}} \\
\sum_{q=1}^{7} \tilde{T}_{q} &= T(f_{s}, f_{e}, J^{a}, A, J^{d}, D, L^{s}, F) \quad \text{(4.29)} \\
\tilde{T}_{8} &= \frac{L_{c,d}}{f_{e}}
\end{align*}$$

The framework eventually outputs $\tilde{T}_{\text{total}}$ with generated kinematic profiles during feedrate planning.

4.3.2 Corner Parameter Identification

As mentioned in Section 4.3, the proposed framework predicts cycle times by combining analytical and data-driven approaches to learn controller-dependent interpolation models based on pre-recorded kinematic profiles. ANNs output prediction results as described in Eq. 4.17, Eq. 4.19, and Eq. 4.21; therefore, it is essential to identify corresponding ground-truth information that is used for training ANNs prior to the prediction. The framework equips the identification procedure that takes inputs as toolpaths geometry and the kinematic profiles, outputs necessary parameters. By utilizing identified parameters, JLFP can reconstruct kinematic profiles with the same
cycle time as actual kinematic profiles.

### 4.3.2.1 Corner Feedrate Identification

As the initial step of parameter identification, types of corners where machines change their feedrate or not are identified for local and global corners with corresponding corner feedrates. The process starts from finding local minima in the given feedrate profile $f(t)$, then ties up identified minimum feedrates to the corners. Suppose that $f(t)$ owns $Q$ sections in total, where consist of acceleration, constant feedrate, and deceleration duration as shown in Fig. 4.10, the minimum feedrate $f_r^\text{min}$ ($r = 1, \ldots, Q - 1$) and the corresponding $x$- and $y$-axis positional information $P_r^\text{min} = [x_r^\text{min}, y_r^\text{min}]^T$ in the recorded trajectory $x(t)$ and $y(t)$ are measured as follows,

$$
\begin{align*}
    f_r^\text{min} &= f(t_r^\text{min}) \\
    x_r^\text{min} &= x(t_r^\text{min}) \\
    y_r^\text{min} &= y(t_r^\text{min})
\end{align*}
$$

where $t_r^\text{min}$ expresses the time between when the deceleration duration starts at $t_r^d$ and the acceleration duration ends at $t_r^a$, as follows,

$$
    t_r^\text{min} = \arg \min_{t_r^d \leq t \leq t_r^a} f(t).
$$

(4.31)
(a) trajectory

(b) kinematic profile

Figure 4.10: Illustrative example of identified corner information based on its geometry and kinematic profile.

\( \mathbf{P}_r^{\min} \) is then linked with the closest corner \( C_r \) by identifying its number \( i' \) as follows,

\[
i' = \arg \min_{1 \leq i \leq M} \left| \mathbf{P}_i - \mathbf{P}_r^{\min} \right|
\]  

(4.32)

Based on the identified \( C_r \), local corners \( C^l \) and global ones \( C^g \), that can be obtained from the geometric analysis described in Section 4.3.1.1, are labeled as follows,

\[
C^l_j = \begin{cases} 
C^l_{j, \text{non-merged}} & \text{if } C_r \in C^l \\
C^l_{j, \text{merged}} & \text{otherwise}
\end{cases}
\]  

(4.33)

\[
C^g_j = \begin{cases} 
C^g_{j, \text{non-merged}} & \text{if } C_r \in C^g \\
C^g_{j, \text{merged}} & \text{otherwise}
\end{cases}
\]  

(4.34)
Note that merged local and global corners do not have corner feedrate values but keep commanded ones to be regarded as straight-line portions. Acquired information by Eq. 4.33 and Eq. 4.34 are utilized as ground truth for training CM-Nets.

After identifying non-merged corners $C_{\text{non-merged}}$, corner feedrates $f_c$ for corresponding corners are calculated. For local corners, identified $f_{r\text{min}}$ is simply used as the local ones. For global corners, there might be multiple $f_{r\text{min}}$ since machines would try to generate accurate trajectories while slowing down their feedrate at every individual corner depending on installed interpolator dynamics. Therefore, the average value of minimum feedrates measured within the global corner is treated as the corner feedrate. Based on the two different conditions, $f_n^c$ are expressed as follows,

$$f_n^c = \begin{cases} 
    f_{r\text{min}} & \text{if } C_{n}^{l,\text{non-merged}} \\
    \frac{1}{s} \sum_{r'} f_{r'\text{min}} & \text{if } C_{n}^{g,\text{non-merged}} 
\end{cases}$$

(4.35)

Identified $f^c$ are provided for training CF-Nets.

4.3.2.2 Kinematic Limits Identification

In the next step of parameter identification, kinematic limits $J^a$, $A$, $J^d$, and $D$ are identified for generating jerk-limited kinematic profiles with the same cycle time as actual kinematic profiles. Such parameters can be varied depending on equipped controllers and also geometric characteristics of given toolpaths. Hence, kinematic limits are identified for every line segment $L^s$ where JLFP generates a trapezoidal acceleration profile based on solving the optimization problem by the sequential
quadratic programming[19].

The problem formulation starts from setting the objective function. Suppose that kinematic limits for JFLP to generate the profiles from $C_n^{\text{non-merged}}$ to $C_{n+1}^{\text{non-merged}}$ should be identified. Ideally, $L_n^s$ should be the same as the travel length derived analytically as follows,

$$L_n^s = \frac{F + f_n^s}{2J_n^a A_n} \left\{ A_n^2 + (F - f_n^a) J_n^a \right\} + FT_{4,n} + \frac{F + f_n^e}{2J_n^d D_n} \left\{ D_n^2 + (F - f_n^e) J_n^d \right\},$$

where $F$, $f_n^s$, and $f_n^e$ are the commanded feedrate, and already identified corner feedrates at the starting and ending position in Section 4.3.2.1, respectively. $T_{4,n}$ also denotes the time duration of the constant feedrate phase that can be measured from pre-recorded kinematic profiles. Then, the objective function is set based on Eq. 4.36 as follows,

$$\arg\min_{J_n^a, A_n, J_n^d, D_n} L(J_n^a, A_n, J_n^d, D_n) - L_n^s$$

(4.37)
subject to \( T_{1,n} = T_{3,n} = \frac{A_n}{J_n^a} > 0 \) \hspace{1cm} (4.38a)
\( T_{5,n} = T_{7,n} = \frac{D_n}{J_n^d} > 0 \) \hspace{1cm} (4.38b)
\( T_{2,n} = \frac{F - f_n^a}{A_n} - \frac{A_n}{J_n^a} \geq 0 \) \hspace{1cm} (4.38c)
\( T_{6,n} = \frac{F - f_n^c}{D_n} - \frac{D_n}{J_n^d} \geq 0 \) \hspace{1cm} (4.38d)
\( T_n^a = \frac{F - f_n^s}{A_n} + \frac{A_n}{J_n^a} \) \hspace{1cm} (4.38e)
\( T_n^d = \frac{F - f_n^c}{D_n} + \frac{D_n}{J_n^d} \) \hspace{1cm} (4.38f)

The qualitative description of Eq. 4.37 and Eq. 4.38 is to minimize the difference of travel distances between analytically solved and measured ones while satisfying cycle-time conditions. Eq. 4.38a and 4.38b are applied for fulfilling the conditions of JLFP that acceleration and deceleration exist at every line segment. \( T_{2,n} \) and \( T_{6,n} \) express constant acceleration phases derived by JLFP, and they should not be less than zero; thus, Eq. 4.38c and Eq. 4.38d are added as inequality constraints. The last two equality constraints as shown in Eq. 4.38e and Eq. 4.38f express that analytically derived acceleration and deceleration duration must be equal to the measured acceleration duration \( T_n^a \) and the deceleration one \( T_n^d \). It is important to follow these constraints for identifying kinematic limits that do not violate conditions of JLFP, and also are able to generate kinematic profiles with \( T_n = T_n^a + T_{n,4} + T_n^d \). Based on solving the above optimization problem, ground truth information \( \hat{J}_n^a, \hat{A}_n, \hat{J}_n^d, \hat{D}_n \) are identified to train KL-Nets.
While the sequential quadratic programming [20] solves the optimization problem in Eq. 4.37 and 4.38, it is necessary to also consider that the measured feedrate profile does not have constant feedrate duration $T_{4,n}$. In such case, Eq. 4.36 is modified as follows,

$$L^s_n = \frac{F'}{2J_n^a A_n} \{ A_n^2 + (F' - f_n^s)J_n^a \} + \frac{F'}{2J_n^d D_n} \{ D_n^2 + (F' - f_n^e)J_n^d \},$$

(4.39)

where $F'$ denotes the arbitrary peak feedrate. The optimization problem is set similar to Eq. 4.37 as follows with the constraints in Eq. 4.38,

$$\arg \min_{J_n^a, A_n, J_n^d, D_n} L'(J_n^a, A_n, J_n^d, D_n, F') - L^s_n$$

(4.40)

After solving Eq. 4.40, identified $\hat{J}_n^a, \hat{A}_n, \hat{J}_n^d, \hat{D}_n$ are substituted for $L'(J_n^a, A_n, J_n^d, D_n, F')$ to calculate the difference against $L^s_n$. The derived error is used to identify $\hat{F}_n'$ from $F^{\text{lower}}$ and $F^{\text{upper}}$, which are applied as lower and upper bounds to find the best value in terms of solving the following problem,

$$\arg \min_{J_n^a, A_n, J_n^d, D_n, F'} L'(J_n^a, A_n, J_n^d, D_n, F') - L^s_n$$

(4.41)
Chapter 5: Experiments

5.1 Introduction

The performance of the proposed cycle-time prediction framework described in Chapter 4 is verified against the actual CNC machine. In order for its execution, training toolpaths are generated with random geometric information. Then, the CNC machines execute their trajectory generation and output recorded kinematic profiles. These profiles are utilized to train ANN sequences, and the framework with the learned ANN sequences finally predicts cycle times against given toolpaths prior to trajectory generation. This chapter describes how to design toolpaths for network training, shows cycle-time prediction results against realistic toolpaths, and discusses its performance both qualitatively and quantitatively.

5.2 Training Toolpath Design

As the first step of cycle-time prediction by the proposed framework, data collection is conducted. Data collection aims to prepare various pre-recorded kinematic profiles of given toolpaths, which involve the dynamic behavior of machines by generating trajectories using NC systems. The profiles are then used for corner parameter identification described in Section 4.3.2 that condensates interpolator dynamics as quantitative information; hence, it is essential to reasonably collect data in terms
of covering distributions of characteristics of interpolator dynamics. In order to investigate the distributions, two types of toolpaths are prepared with randomized information.

The first type is defined as local corner toolpaths, as shown in Fig. 5.1. The objective of preparing this toolpath is to capture the local corner interpolation as mentioned in Section 4.2.2. Any machine has to slow down and speed up around every corner except when the machine eliminates junction points. Although the motion of changing its feedrate can be reproduced by JLFP with cornering feedrate and kinematic limits, their values would be different relying on the geometry of toolpaths. Its geometric characteristics can be expressed as the magnitude of cornering angles $\theta$ and length of straight-line segments $l$ that connect consecutive waypoints. In order to cover the change of corner parameters with respect to different geometric characteristics, local corner toolpaths are generated with randomly set $\theta$ and $l$. 

Figure 5.1: Illustrative example of randomly generated local corner toolpath.
The second type is defined as global corner toolpaths, as shown in Fig. 5.2. The objective of preparing this toolpath is to capture the global corner interpolation as mentioned in Section 4.2.3. In addition to the local corner interpolation, machines would globally interpolate circular shape toolpaths with multiple short straight lines while avoiding the fluctuation of kinematic profiles instead of slowing down every corner. Although this motion can also be reproduced by JLFP, it is necessary to change its geometric characteristics to cover the distribution of the global corner interpolator dynamics. Here, global corner toolpaths are generated as follows,

1. Local corner $C_i$ is generated with randomized $\theta_i$, $l_{i-1}$, and $l_i$.
2. A circle tangent to $l_{i-1}$ and $l_i$ is derived with randomized radius $r$.
3. Waypoints are generated along the circle with the randomized interval $l_{\text{interval}}$.
4. Smoothed global corner $C'_i$ is returned while eliminating the original junction point.
5.3 Experiments using Actual Machine

This section demonstrates the cycle-time prediction performance of the proposed framework against realistic toolpaths using an actual machine. The procedure starts from gathering recorded kinematic profiles of designed toolpaths as described in Section 5.2 from the machine. Then, ANN sequences in the proposed framework are trained to learn the correspondence between geometric characteristics and interpolator dynamics. Finally, the framework outputs kinematic profiles based on JLFP with predicted information and cycle-time prediction results against given toolpaths that are not used during the network training phase.
5.3.1 Experimental Setup

The experiments to demonstrate the cycle-time prediction is executed using a DMG Mori machine tool with Heidenhain NC system, as shown in Fig. 5.3. The machine executes trajectory generation process to gather recorded kinematic profiles of given designed toolpaths for network training, summarized as Table 5.1. Note that local corner toolpaths are prepared with different types of line length and corner angle conditions, such as short, long, and obtuse, to avoid imbalanced data distribution. For each type of toolpath, three sets of randomly generated G-codes are prepared and sent to the machine. The machine generates their trajectories and outputs corresponding kinematic profiles while changing user-specified parameters $F = 3500, 4000, 4500$ [mm/min] and $\varepsilon = 10, 20, 30$ [micron].

<table>
<thead>
<tr>
<th>Type of toolpath</th>
<th>local – short</th>
<th>local – long</th>
<th>local – obtuse</th>
<th>global</th>
</tr>
</thead>
<tbody>
<tr>
<td>line length ($l$) [mm]</td>
<td>$0.2 &lt; l &lt; 1$</td>
<td>$1 &lt; l &lt; 20$</td>
<td>$0.2 &lt; l &lt; 1$</td>
<td>$20 &lt; l &lt; 50$</td>
</tr>
<tr>
<td>corner angle ($\theta$) [deg]</td>
<td>$0 &lt; \theta &lt; 160$</td>
<td>$0 &lt; \theta &lt; 160$</td>
<td>$0 &lt; \theta &lt; 50$</td>
<td>$20 &lt; \theta &lt; 160$</td>
</tr>
<tr>
<td># of local corners</td>
<td>1000</td>
<td>1000</td>
<td>2000</td>
<td>–</td>
</tr>
<tr>
<td># of global corners</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>1000</td>
</tr>
</tbody>
</table>

After obtaining kinematic profiles, ANN sequences in the proposed framework are trained with identified corner parameters described in Section 4.3.2. Here, the number of hidden neurons are selected as shown in Table 5.2.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>
Figure 5.4: Three toolpaths utilized for evaluating cycle-time prediction performance.

(a) contour toolpath

(b) trochoidal toolpath

(c) pocketing toolpath
Figure 5.4 summarizes prepared three types of toolpaths to evaluate cycle-time prediction performances of the proposed framework. The first type shown in Fig. 5.4 (a) is named contour toolpath consisting of 29 line segments in total. This toolpath has several local corners and two global corners with multiple line segments. The next type, shown in Fig. 5.4 (b), is named trochoidal toolpath composed of 649 line segments. Unlike the former case, it mainly consists of circular portions with a large number of short line segments. The third type of toolpath is also shown in Fig. 5.4 (c), which is named pocketing toolpath. This toolpath has 5013 line segments in total and is composed of a combination of multiple local and global corners. The next sections show the cycle-time prediction results against these toolpaths and assess the performance of the proposed framework.

5.3.2 Cycle-Time Prediction against Contour and Trochoidal Toolpaths

As the first trial, cycle-time prediction against the contour toolpath is conducted. The trajectories of the target toolpath are generated with $F = 3500, 4000, 4500$ [mm/min] and $\epsilon = 10$ [micron]. Table 5.3 summarizes cycle times and corresponding prediction results.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3500</td>
<td>9.57</td>
<td>–</td>
<td>7.93</td>
<td>9.66</td>
<td>0.87</td>
</tr>
<tr>
<td>4000</td>
<td>8.70</td>
<td>–</td>
<td>6.94</td>
<td>8.80</td>
<td>1.16</td>
</tr>
<tr>
<td>4500</td>
<td>8.04</td>
<td>–</td>
<td>6.17</td>
<td>8.15</td>
<td>1.28</td>
</tr>
</tbody>
</table>

Table 5.3: Cycle time prediction result against contour toolpath.
errors of actual, CAM-estimated, and predicted by the proposed framework. The proposed method can predict its cycle time about 1 [\%] error, while the CAM-estimated cycle time has about 20 [\%] error. Based on the results with different commanded feedrate, it can be said that the proposed framework has the ability to precisely predict cycle times regardless of the changes of commanded feedrates. Figure 5.5 shows the visualized result of predicted corner merging behavior by CM-Nets and Figure 5.6 compares kinematic profiles of measured and generated by JLFP with predicted parameters when the commanded feedrate is set as 3500 [mm/min]. First, as shown in Fig. 5.5, CM-Nets predicts all local and global corners should not be merged for this toolpath. The classification results are then projected to predict corner parameters and generate kinematic profiles as shown in Fig. 5.6. Local CF-Net accurately predicts cornering feedrate for every local corner, and Global CF-Net predicts cornering feedrate value. While the actual profile fluctuates at the global
Figure 5.6: Measured (black) and predicted (red) kinematic profiles with $F = 3500$ [mm/min] and $\varepsilon = 10$ [micron] of contour toolpath.
corners, Global CF-Net successfully predicts a reasonable value that is close to the average value of minimum feedrates of junction points within the global corners. In addition, as shown in the jerk and acceleration profiles, KL-Nets provide jerk and acceleration limits that are similar to the peaks in the measured profiles. By combining predicted information and JLFP, the proposed framework can precisely predict cycle time for the simplest toolpath by reproducing the dynamic behavior of interpolation models.

The next trial is to predict cycle time against the trochoidal toolpath. The trajectory of the target toolpath is generated with $F = 3500, 4000, 4500 \text{ [mm/min]}$ and $\varepsilon = 10 \text{ [micron]}$. Table 5.4 summarizes cycle times and corresponding prediction errors of actual, CAM-estimated, and predicted by the proposed framework. Compared to the previous case, the prediction performance drops as ~7.2 $\%$, but also CAM estimation result is dramatically worse as about 60 ~70 $\%$ error. The primary reason

![Figure 5.7: Classification result of CM-Nets against trochoidal toolpath.](image)
Figure 5.8: Measured (black) and predicted (red) kinematic profiles with $F = 3500$ [mm/min] and $\varepsilon = 10$ [micron] of trochoidal toolpath.
for the inaccuracy is that the multiple circular shapes in the toolpath suppress the increase of feedrate to the commanded value. The predicted interpolation models by CM-Nets are visualized in Fig. 5.7, and the measured and predicted motion behaviors can be described in Fig. 5.8 when the commanded feedrate is set as 3500 [mm/min]. As shown in Fig. 5.7, Local CM-Net predicts corner merging behavior where its cornering angle is close to zero, and Global CM-Net estimates that circular shapes in the toolpath should be regarded as non-merged global corners. Then, JLFP generates kinematic profiles while incorporating predicted information by the corner parameter prediction sequence. From Fig. 5.8 (a), the consistent cycle-time error can be confirmed until passing through consecutive circular portions. The one reason for the consistent error is the predicted cornering feedrate at every circular portion is a bit lower than the average cornering feedrate. Another reason comes from the difference of reaching peak feedrate at a short line segment between consecutive circular shapes, as shown in Fig. 5.8 (b). Recall that the framework calculates travel distances based on the given toolpath information such as the travel distance at the global corner as $L_c$ and at the straight line segment as $L_s$, where feedrate value is constant, and its

<table>
<thead>
<tr>
<th>feedrate [min/sec]</th>
<th>cycle time [sec]</th>
<th>actual</th>
<th>CAM-estimated</th>
<th>proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>3500</td>
<td></td>
<td>29.0</td>
<td>11.5</td>
<td>30.8</td>
</tr>
<tr>
<td></td>
<td>error [%]</td>
<td>–</td>
<td>60.4</td>
<td>6.41</td>
</tr>
<tr>
<td>4000</td>
<td></td>
<td>28.7</td>
<td>10.0</td>
<td>30.8</td>
</tr>
<tr>
<td></td>
<td>error [%]</td>
<td>–</td>
<td>65.0</td>
<td>7.20</td>
</tr>
<tr>
<td>4500</td>
<td></td>
<td>28.5</td>
<td>8.92</td>
<td>30.3</td>
</tr>
<tr>
<td></td>
<td>error [%]</td>
<td>–</td>
<td>68.6</td>
<td>6.49</td>
</tr>
</tbody>
</table>

Table 5.4: Cycle time prediction result against trochoidal toolpath.
value is planned by JLFP with predicted parameters. Although the calculated $L^c$ during the prediction phase is the same as the sum of total short line segments at the global corner, the actual machine would start to increase feedrate before passing through the circle, and also decrease its value before entering the next circle due to the implemented corner blending algorithm in the machine. Thus, the corresponding $L^c$ for the measured kinematic profiles would be shorter than the sum of total small line segments and the machine can be allowed to increase its feedrate than the predicted kinematic profiles. In summary, while the result still has the error compared to the result against the simpler toolpath, the proposed framework has far better cycle-time prediction performance relative to the conventional CAM-estimated method against the toolpath with multiple non-merged global corners.

5.3.3 Cycle-Time Prediction against Pocketing Toolpath

This section shows cycle-time prediction results against the pocketing toolpath, which consists of various lengths of line segments and different types of corners where interpolation models are used, as shown in Fig. 5.4 (c). Table 5.5 summarizes cycle

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3500</td>
<td>99.5</td>
<td>78.3</td>
<td>100.9</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>21.3</td>
<td>1.37</td>
</tr>
<tr>
<td>4000</td>
<td>93.1</td>
<td>68.5</td>
<td>94.2</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>26.4</td>
<td>1.13</td>
</tr>
<tr>
<td>4500</td>
<td>88.6</td>
<td>60.9</td>
<td>89.7</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>31.3</td>
<td>1.26</td>
</tr>
</tbody>
</table>
times and corresponding prediction errors of actual, CAM-estimated, and predicted by the proposed framework. Based on the prediction results shown in Table 5.5, the proposed framework accurately predicts cycle times about 1 \% error regardless of the commanded feedrate is changed. This performance is notable since the proposed method can maintain accurate cycle-time performance even when the given toolpath becomes more complex from the simpler toolpaths in terms of the amount and variety of geometric information.

The predicted information is visualized as shown in Fig. 5.9 and Fig. 5.10. The first prediction procedure is to classify which corners should be regarded as straight line segments, or applied interpolation models, as shown in Fig. 5.9. The outside of the toolpath basically consists of long-straight lines whereas the inside possesses multiple circular-shape portions. Based on the training results that provide interpolator dynamics of the target machine, Local CM-Net identifies several merged local lines, and Global CM-Net classifies merged and non-merged global corners. One example of different types of global corners is zoomed-in Fig. 5.9. Here, the red-colored portion is classified as the non-merged one, and the green-colored portions are merged ones. This decision-making would be reasonable since the red portion forms an acute change of the direction to move while it is composed of multiple line segments. In addition, each individual corner in the green portions has an obtuse corner angle and a longer line segment compared to the red one so that the machine could maintain a higher feedrate due to the small amount of the changes of the direction to move. In general, Global CM-Net classifies acute global corners with shorter line lengths as non-merged one and vise versa.
Figure 5.9: Classification result of CM-Nets against pocketing toolpath.
Figure 5.10: Measured (black) and predicted (red) kinematic profiles with $F = 3500$ [mm/min] and $\varepsilon = 10$ [micron] of pocketing toolpath.
Then, the framework generates jerk-limited kinematic profiles with predicted corner parameters for classified non-merged local and global corners, as shown in Fig. 5.10. In a broad sense, the framework outputs kinematic profiles with similar trends to the measured ones in terms of when and how to slow down and speed up its feedrate. Thus, it can finally predict cycle times with about 1 [%] error. However, there are several different characteristics against the motion of the actual machine. First, the predicted corner feedrates by CF-Nets are more consistent than the actual ones, as shown in the feedrate profile in the displacement domain. Recall that Global CF-Net takes geometric features of global corners based on its circular shapes, such as average cornering angles, lengths of line segments, and curvatures. While this information contributes to predicting corner feedrates, it would be also necessary to add cornering angle information for global corners, because it shows how amount the machine can spend time to change direction. This can be denoted as the central angle of the estimated circular shapes of each global corner so that Global CF-Net could predict corner feedrate more accurately. Next, some amount of the error comes from over and underestimation of peak feedrate when the machine passes obtuse global corners, as shown in Fig. 5.11 (a) and (b). The prediction procedure decides green-colored global corners as straight line segments and JFLP increases its feedrate to the commanded value. However, the measured profile describes that the machine does not increase the feedrate to the maximum allowed value. Instead, it keeps lower value consistently within these corners. It is assumed that these global corners should be classified as non-merged global ones to suppress peak feedrate. Similar trends can be confirmed when the framework generates kinematic profiles for circle portions,
as shown in Fig. 5.11 (c) and (d). The machine keeps its feedrate lower than the commanded one whereas the prediction result increases its value as possible. One interesting characteristic of measured kinematic profiles is that the machine would change peak feedrate depending on the radius of the circle. The former and latter peaks in the feedrate profile correspond to outer and inner circles in the zoomed toolpath, respectively, and there are differences about 10 [mm/sec] in their constant feedrate duration. From this behavior, it is considered that the curvature of global corners clearly affects the allowable feedrate value for the interpolation models.

In addition to the above reasoning of inaccuracies, incorrect prediction occurs due to missing the portion where the machine greatly lower its feedrate, as shown in Fig. 5.12. Although the measured feedrate profile greatly changes its value from the commanded one to under 10 [mm/sec], as shown in Fig. 5.12 (b) and (d), the framework completely misses such corners by classifying the merged global ones, or overestimate its value. The reason comes from the framework that regards these corners as global ones then predicts the existence of corner feedrates and kinematic limits, while the actual machine treats them as local corners. As shown in Fig. 5.12 (a) and (b), these corners have local corners, but also consecutive small line segments immediately after that. The sets of line segments are classified as global corners during the geometric analysis process because these line segments have similar trends in terms of their cornering angle and lengths information but should be handled as part of a long single line. The geometric analysis process should be improved to identify such a series of small line segments regarded as the part of local corners when cornering angles are close to zero.
Figure 5.11: Zoomed pocketing toolpath and corresponding feedrate profiles in displacement domain where incorrect prediction occurs (case 1 and case 2).
Figure 5.12: Zoomed pocketing toolpath and corresponding feedrate profiles in displacement domain where incorrect prediction occurs (case 3 and case 4).
5.4 Discussion

This section discusses the performance of the proposed cycle-time prediction framework based on the experimental results using the actual CNC machine. As described in Section 5.3.2 and Section 5.3.3, it has the ability to outperform the conventional CAM-based estimation method. The prediction accuracy can be improved since the framework captures the dynamics of interpolation models, which are defined in Section 4.2, based on a data-driven approach. In addition, the framework has robustness against various types of toolpath geometry by clearly defining local and global interpolation models, applying different prediction sequences inside the framework, and produce jerk-limited kinematic profiles. This is the advantage compared to the previous research presented in [3]. Furthermore, it can be expected that the framework can predict cycle times for different machines by learning their dynamic behaviors using ANNs since the definition of interpolation models and the use of geometric information, as well as kinematic profiles, are common in any machine.

Several limitations and reasoning of inaccuracies are found based on experimental results. The first limitation can be seen in the prediction results against the trochoidal toolpath. The prediction results are worse than the other toolpaths, since JLFP generates kinematic profiles without considering corner smoothing of the CNC machine, and the accumulation of the cycle-time error occurs. This problem should be resolved by also predicting positional information of the trajectory in addition to the parameters for JLFP. The next reasoning of inaccuracies is the incorrect classification by CM-Nets and corner feedrate prediction by CF-Nets cause gaps between actual and predicted
kinematic profiles. Further improvement of prediction accuracy should be done by
turning parameters of ANN such as the number of hidden units, adding other input
features to describe geometric characteristics in detail, or increasing the amount of
training data. Also, during cycle-time prediction for the pocketing toolpath, incorrect
results of the geometric analysis procedure are found due to the consecutive small line
segments form a part of the straight-line segment while the process regards them as
one of the global corners. It should be modified by adding functionality to judge local
or global corners in detail with the threshold related to cornering angle information.
Chapter 6: Conclusion & Future Work

This thesis presents a new framework to accomplish accurate machining cycle-time prediction by capturing controller-dependent dynamic behavior of CNC machines. The proposed cycle-time prediction framework consists of JLFP to model high-level interpolator dynamics analytically and ANN sequences to capture machine-specific dynamic behavior based on collected data obtained from the target CNC machine. It takes toolpaths with user-specified parameters to output corresponding jerk-limited kinematic profiles with predicted corner parameters, and cycle times can be predicted by measuring the time domain of the kinematic profiles. Experimental evaluation using the actual CNC machine shows that the framework has the ability to predict cycle time up to 99 [%] accuracy against realistic toolpaths. Compared to the cycle-time prediction works so far, several advantages can be seen as follows. First, it enables the elimination of machine-specific treatments by utilizing data-driven techniques to empirically capture interpolation models. Next, cycle-time prediction via kinematic profiles can effectively utilize learned dynamic behaviors compared to the direct cycle-time prediction approaches. Furthermore, JLFP plays a role to describe the detailed motion of slowing down and speeding up around corners so that it has robust prediction performance against complex toolpaths with various geometric characteristics. In summary, the proposed framework can accomplish accurate cycle-time prediction by utilizing the advantages of both analytical modeling and a data-driven approach.
Future work plans to evaluate the performance against other CNC machines using either simulators or actual machines and compare its results with the presented performances in this thesis to validate its generalization ability.
Bibliography


