BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES
DOI: 10.24425/bpasts.2024.148940

CNC Machine Control Using Deep Reinforcement Learning

Dawid Kalandyk[1], Bogdan Kwiatkowski[2], and Damian Mazur[2]

[1] Doctoral School of the Rzeszów University of Technology, Powstańców Warszawy Ave. 12, 35-959 Rzeszow, Poland, d.kalandyk@prz.edu.pl
[2] Department of Electrical and Computer Engineering Fundamentals, Rzeszow University of Technology, W. Pola str. 2, 35-959 Rzeszów, Poland; b.kwiatkowsk@prz.edu.pl, mazur@prz.edu.pl

Abstract. Optimization of industrial processes such as manufacturing or processing of specific materials is a point of interest for many researchers, and its application can lead not only to speeding up the processes in question, but also to reducing the energy cost incurred during them. This article presents a novel approach to optimizing the spindle motion of a computer numeric control (CNC) machine. The proposed solution is to use deep learning with reinforcement to map the performance of the Reference Points Realization Optimization (RPRO) algorithm used in industry. A detailed study was conducted to see how well the proposed method performs the targeted task. In addition, the influence of a number of different factors and hyperparameters of the learning process on the performance of the trained agent was investigated. The proposed solution achieved very good results, not only satisfactorily replicating the performance of the benchmark algorithm, but also, speeding up the machining process and providing significantly higher accuracy.

Key words: deep reinforcement learning; cnc machining; machining optimization

1. INTRODUCTION

The ability to carve parts of complex shapes with high accuracy not only allows the creation of robots capable of performing many tasks, but also enables the development of technology. For a long time, therefore, computer numeric control (CNC) machines has been the subject of many studies [1], [2]. Fields being developed include general machine control [3], estimation, and minimization of machining errors [4], [5]. The topic attracting the most attention is spindle motion path planning and determination of spindle motion control signals in successive time steps [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. Artificial intelligence methods actively developed recently are also eagerly used for the previously mentioned tasks [16], [17], [18], [19], [20]. A lot of attention has also been paid by researchers to the reinforcement learning (RL) algorithm [21], which, due to its versatility, can be applied to various types of control tasks [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33]. Due to the operating characteristics of the shop floor, or even the machining process itself (time steps), this algorithm is also widely used for various tasks: manufacturing floor process control [34], [35], [36], [37], [38], damage prediction [39], equipment overhaul management [40], [41], selection of equipment settings [42], [43], [44], [45], [46], [47], [48] and optimizing the spindle motion path and determining the g-code [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59] that determines spindle motion. In the paper [60], the authors proposed to use fuzzy logic systems taught by the particle swarm method to replicate the operation of the g-code determination algorithm used in industry [6]. In this way, they wanted to linearize the computational complexity of this algorithm and make its operation independent of CNC machine dynamics parameters such as maximum spindle velocity or maximum linear acceleration of the spindle. The present work builds on the aforementioned research by using a deep learning algorithm with reinforcement to map the performance of the Reference Points Realization Optimization (RPRO) algorithm.

Contributions of this work are as follows:

- a novel approach that involves using a deep reinforcement learning algorithm to mimic the work of the RPRO algorithm,
- study of a number of different configurations of parameters of the learning process,
- testing the accuracy of mimicking the operation of the RPRO algorithm,
- optimization of the machining process

In section 2. the proposed solution will be characterized. Then, in Section 3. the conducted research will be described and the results of the performed experiments will be presented and analyzed. Section 4. will summarize the paper and indicate the direction of further work.

2. Proposed framework

According to the premise, the work involves applying a deep reinforcement learning algorithm to the task of generating the g-code that controls the operation of the CNC machine. On its form depends not only the duration of the machining of the fabricate, but also the accuracy of the production of the final workpiece, as well as the level of tool wear and electricity consumption. The task is to train a neural network to respond to the given input signals in such a way as to mimic the behavior of another algorithm. According to the authors, this allows to linearize the process of generating the g-code - while the RPRO algorithm for each time step must perform checks for a certain number of forward steps, the proposed solution directly generates the desired output signal. The proposed system of learning and operation of the algorithm is shown in Figure 1.

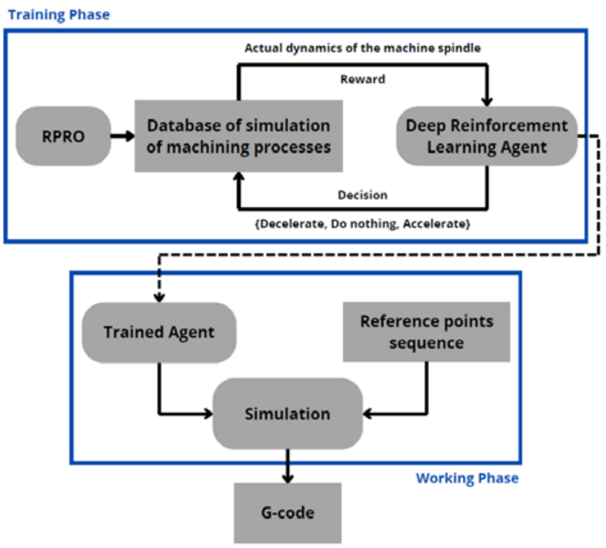

FIGURE 1 Proposed framework

A. Used base algorithm

The RPRO algorithm is aimed at optimizing the g-code in terms of machining time keeping in mind, however, to ensure the best possible workpiece manufacturing accuracy at a given time. It is worth noting that the mentioned workpiece machining accuracy should be understood as the average accuracy of the spindle reaching each reference point. For each time step of 2ms, the algorithm decides whether the spindle should accelerate, decelerate, or perhaps should move with unchanged dynamics. The calculations are carried out off-line, allowing different versions of the g-code to be checked going forward and manipulated accordingly. The disadvantage of this approach is the inability to operate in real time, receiving as input the actual data describing the dynamics of the machine and the spindle, rather than those from the simulation. During the simulation and operation of the algorithm, the current spindle position, spindle velocity and spindle acceleration are calculated based on equation (1). The exact way the algorithm works is described in [6].

$$\vec{J}(t) = \frac{da}{dt} = \dot{\vec{a}}(t) = \frac{d^2v}{dt^2} = \ddot{\vec{v}}(t) = \frac{d^3r}{dt^3} = \dddot{\vec{r}}(t) \qquad (1)$$

B. Reinforcement learning

The reinforcement learning algorithm is characterized by stepwise action - the agent decides in successive time steps t what action a should be taken to maximize the sum of rewards r granted to him after each choice. His learning is accomplished through his interaction with the environment in which he moves and updating information about it. According to currently accepted methods, this knowledge can be represented in two ways. First, by a value function $V^{\pi}(s)$ (2) specifying the expected total value of the reward to be gained starting from a given state s_0. The second way is by an action value function $Q^{\pi}(s,a)$ (3) specifying the expected total value of the reward to be earned when an agent starts from a given state s_0 and performs an action a_0 in it. The discount factor γ is also an important aspect controlling the learning process and the final behavior of the agent. It is also referred to as the agent's foresight factor and takes values in the range (0;1).

$$V^{\pi}(s) = E_{\pi}[\sum_{t=0}(\gamma^t \cdot r^t)] \; where \; s_0 = s \qquad (2)$$

$$Q^{\pi}(s,a) = E_{\pi}[r_0 + \sum_{t=0}(\gamma^t \cdot r^t)] \; where \; s_0 = s, a_0 = a \qquad (3)$$

For the simplest model of the environment and for many applications, a basic tabular representation of the functions mentioned is sufficient. However, if the problem domain is not finite, then some method of approximating the values of these functions can be used. One such method is to use a neural network for this purpose, the so-called deep learning with reinforcement (DRL). Such an action provides the possibility of learning an agent that is likely to be able to perform satisfactorily even in the case of small changes in the environment in which it moves.

C. Database

To conduct the experiments, a previously prepared database consisting of stored simulations of machining processes for different reference point paths and a number of combinations of machine dynamics parameters was used. These simulations were prepared using the RPRO algorithm. The values of the various parameters are shown in Table 1. High density of reference points means that the distances between them are less than 1mm, medium density means that the distance between them is greater than 1mm but less than 10mm, while low density means that the distances between successive reference points are between 10mm and 100mm.

TABLE 1 Database Parameters Values [60]

Parameter	*Possible Values*	*Combinations Factor*
Trajectory length	$\{15, 50. 100\}$	3
Reference points density	$\{\text{Low}, \text{Medium}, High\}$	3
Maximum velocity $\left[\frac{m}{min}\right]$	$\{2.5, 4.0. 6.0. 8.0\}$	4
Maximum acceleration $\left[\frac{m}{s^2}\right]$	$\{1.5, 1.8, 2.0. 2.5, 3.0\}$	5
Jerk $\left[\frac{m}{s^3}\right]$	$\{10. 20. 30\}$	3
Target precision $[mm]$	0.01	1
Time step duration $[s]$	0.002	1

3. Experiments

The experiments conducted included testing the impact of neural network architectures of different complexity for a number of combinations of hyperparameters of the learning process. The differences in architectures concerned two aspects, namely the number of neurons in successive layers and the type of activation function. The exact set of architectures studied is shown in Table 2, while their general scheme is shown in Figure 2. It was determined that the input of the network would be the following signals: normalized current spindle velocity, normalized current spindle acceleration and normalized distance to the next reference point. Normalization of the data was performed with respect to, accordingly : the maximum allowed spindle velocity, the maximum allowed spindle acceleration and the largest distance between adjacent reference points. Double Deep Q-Learning (DDQL) method was selected as the learning algorithm. It is also necessary to specify what signals the agent will receive after performing a particular action. A very simple system of assigning reinforcement was established, namely, when the agent performed an action in accordance with the decision of the RPRO algorithm the reward was equal to 0, otherwise he received a penalty equal to -0.1. Exact form of reinforcement signal is described by formula (4).

$$r^t = \begin{cases} 0, & if\ a_{agent} = a_{RPRO} \\ -0.1 & otherwise \end{cases}, \tag{4}$$

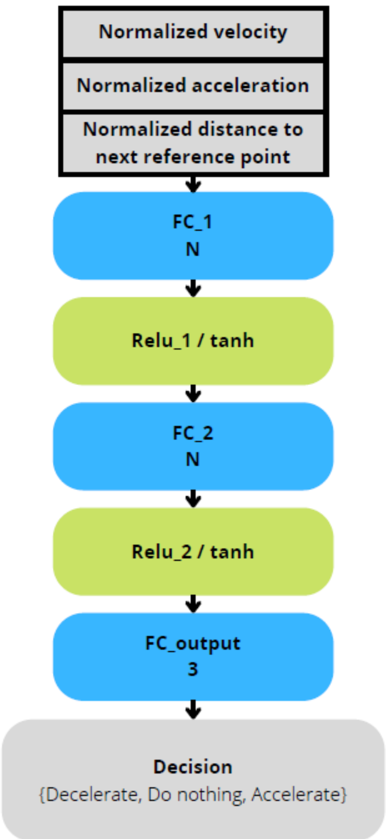

FIGURE 2 Diagram of the used neural network

TABLE 2 Evaluated architecture parameters

Evaluated Architecture Number	Layerwise Neuron Count (N)	Layerwise Activation Function
1	24	ReLU
2	24	tanh
3	20	ReLU
4	20	tanh
5	16	ReLU
6	16	tanh
7	12	ReLU
8	12	tanh
9	8	ReLU
10	8	tanh
11	6	ReLU
12	6	tanh

At first, it was decided to evaluate the proposed solution depending on the density of the accumulation of reference points. Thus, the study was carried out for 3 subsets of the database corresponding to combinations of other parameters for trajectories of 15 points. Each subset contained 600 recorded machining processes. Learning was repeated on successive 5 pairs from the combinations for 10 trajectories, resulting in 120 processes in the training set and 480 in the test set. Each learning process was repeated 5 times to significantly reduce the impact of randomness in the learning process. All experiments were performed using the Matlab 2022a environment extended with the appropriate toolboxes. Subsequent parameters of the learning process were set as follows: *maxEpisodes* (500), *TargetSmoothFactor* (0.05), *TargetUpdateFrequency* (5), *MiniBatchSize* (64), *ExperienceBufferLength* (5000), *MaxStepsPerEpisode* (300). For convenience, the listed parameters are summarized in Table 3.

TABLE 3 Learning process hyperparameters values.

Parameter name	Parameter value
maxEpisodes	500
TargetSmoothFactor	0.05
TargetUpdateFrequency	5
MiniBatchSize	64
ExperienceBufferLength	5000
MaxStepsPerEpisode	300

A measure of the level of replication of the RPRO algorithm's behavior by the proposed solution can be expressed by the Pearson correlation coefficient between the number of steps in successive test machining processes. An interesting addition to it is also the average accuracy of the machining process expressed in micrometers. The listed metrics describing the results of the experiment are shown in Table 4 and Table 5. Note that these values are the medians from all repetitions and training pairs for specific combinations of architecture and learning process hyperparameters values. In situations where, the algorithm failed to achieve the intended effect (learning process did not reach convergence), it was not possible to determine the correlation coefficient, which is indicated in the tables by a triple pause (---). Due to the large number of failures for a subset of the base with sparsely distributed reference points, the corresponding part of the results was omitted and will be focus of future study.

TABLE 4 Results of the first experiment - median correlation coefficient between the number of steps in the processing performed by the RPRO algorithm and the trained agent

Steps Count Correlation		Discount Factor					
		0.99			0.999		
Ref. Points Density	Network Arch. Number	Learning Rate			Learning Rate		
		1e-3	1e-4	1e-5	1e-3	1e-4	1e-5
High	1	0.847	0.750	0.848	0.647	0.844	0.844
	2	0.855	0.852	0.855	0.847	0.842	0.846
	3	0.844	0.855	0.848	0.839	0.777	0.847
	4	0.852	0.851	0.849	0.687	0.848	0.849
	5	0.849	0.856	---	0.844	0.853	---
	6	0.849	0.853	---	0.855	0.852	---
	7	0.825	0.857	---	0.847	0.846	---
	8	0.850	0.851	0.849	0.850	0.836	0.848
	9	0.821	0.856	---	0.845	0.850	---
	10	0.852	0.855	0.846	0.846	0.848	0.845
	11	0.842	0.851	---	0.848	0.848	---
	12	0.855	0.849	---	0.844	0.845	---
Medium	1	0.773	0.777	0.774	0.772	---	---
	2	0.777	0.774	0.774	---	0.472	0.538
	3	0.774	0.774	0.774	---	---	---
	4	0.771	0.777	0.775	---	0.773	0.726
	5	0.774	0.774	---	---	---	---
	6	0.771	0.775	0.664	---	0.386	0.237
	7	0.777	0.774	0.578	---	---	0.041
	8	0.775	0.776	0.774	---	0.171	0.172
	9	0.774	0.774	---	---	---	---
	10	0.774	0.775	0.774	---	0.009	0.112
	11	0.772	0.774	0.543	---	---	0.054
	12	0.774	0.775	---	---	---	---

TABLE 3 Results of the first experiment - median of average error of the machining process

Mean Error		Discount Factor					
		0.99			0.999		
Ref. Points Density	Network Arch. Number	Learning Rate			Learning Rate		
		1e-3	1e-4	1e-5	1e-3	1e-4	1e-5
High	1	2.38	2.32	2.56	2.47	2.46	2.55
	2	2.45	2.43	2.51	2.46	2.44	2.63
	3	2.4	2.46	2.49	2.44	2.45	2.58
	4	2.45	2.49	2.51	2.52	2.52	2.46
	5	2.56	2.5	3.11	2.41	2.48	3.29
	6	2.47	2.43	4.69	2.48	2.35	4.81
	7	2.32	2.37	4.45	2.46	2.52	4.57
	8	2.49	2.47	2.56	2.49	2.43	2.57
	9	2.4	2.5	3.16	2.41	2.51	3.38
	10	2.45	2.48	2.54	2.39	2.55	2.54
	11	2.47	2.51	3.86	2.47	2.49	4.48
	12	2.41	2.51	435	2.46	2.48	437
Medium	1	5.53	5.9	5.41	2708	4496	4450
	2	5.79	5.31	5.14	4835	2661	2926
	3	5.54	5.03	1561	4481	4799	3580
	4	4.99	5.47	5.34	4708	2531	816
	5	5.37	6.2	7.08	4653	4742	4599
	6	5.08	5.56	2445	4852	1814	3607
	7	5.02	5.19	2414	3727	4741	3581
	8	5.42	5.14	5.06	4698	3369	2443
	9	4.83	5.14	3344	3568	3742	4455
	10	5.4	5.34	5.13	4677	4286	4366
	11	5.55	5.91	2446	4450	4622	3472
	12	5.21	5.32	4421	4736	2635	4832

The observed results show that for a subset of reference paths with dense reference points, the proposed method achieves a high level of correlation in the number of steps during individual machining processes. It is also interesting to note that the proposed solution achieves significantly better results in terms of machining process accuracy. However, in order to be able to verify whether this is by chance at the expense of the time spent on machining, it is necessary to analyze the median of the steps performed in the individual machining processes. For the RPRO algorithm, this value is 110 for densely spaced reference points and 500 for moderately spaced reference points, respectively. The corresponding medians for the proposed solution are shown in Table 6. It can be observed that for both subsets of the base the median number of steps is slightly smaller than for the RPRO algorithm. Thus, the learned agent does not behave identically to the benchmark algorithm, but it achieves significantly better processing accuracy than it in a slightly shorter processing time.

TABLE 4 Results of the first experiment - the median of number of steps during the machining process

Steps Count		Discount Factor					
		0.99			0.999		
Ref. Points Density	Network Arch. Number	Learning Rate			Learning Rate		
		1e-3	1e-4	1e-5	1e-3	1e-4	1e-5
High	1	85	88	80	86	85	80
	2	83	83	81	83	82	82
	3	85	83	82	84	85	83
	4	82	82	80	84	80	82
	5	82	81	74	87	82	75
	6	83	84	69	82	84	72
	7	89	82	71	85	86	71
	8	82	83	80	81	88	81
	9	86	80	75	85	80	75
	10	83	81	80	83	83	82
	11	85	80	76	84	81	72
	12	84	80	1	83	80	1
Medium	1	471	469	476	179	71	101
	2	475	482	484	1	293	204
	3	478	482	446	73	1	101
	4	489	467	478	1	322	453
	5	482	465	431	1	1	101
	6	486	480	410	1	263	234
	7	487	482	406	87	1	101
	8	481	485	483	1	290	472
	9	488	486	367	159	82	109
	10	479	481	482	1	280	243
	11	476	471	487	68	1	258
	12	478	481	71	14	297	1

In addition, at this point it is also necessary to analyze the decrease in the quality of the learned agent's work with an increase in the interval between the reference points as seen in Table 4, Table 5. and Table 6. The intention of the first experiment was to test the ability of the studied algorithm to replicate the performance of the RPRO algorithm. Wanting to ensure the constancy of as many parameters as possible while reducing the computation time, the authors assumed that the aforementioned maximum length of the episode would be 300. Less frequently distributed reference points obviously lengthen the machining process, and limiting the episode to the given value meant that for an average distribution of them, the agent was not able to achieve as good results as for densely distributed points, since it did not have the opportunity to experience the end of the machining process during learning. For densely spaced reference points, on the other hand, the agent was unable to achieve satisfactory results at all having had the opportunity to experience only a small initial portion of the entire machining process (300 out of 4446 steps or less than 7%). The last noteworthy fact is the indication of the presented results that potentially the best values of hyperparameters could be learning rate equal to 0.001 and discount factor equal to 0.99, respectively. All tested architectures performed very well with a slight advantage for those using sigmoidal activation function. This is very good news in terms of the development of the idea and the complication of the environment to achieve control in multiple dimensions of motion.

TABLE 5 Results of the second experiment - median correlation coefficient between the number of steps in the processing performed by the RPRO algorithm and the trained agent

Steps Count Correlation		Network Architecture Number											
Learning Rate	**Train Trajectories Count**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0.001	2	0.827	0.839	**0.853**	---	0.807	0.832	0.828	0.833	0.846	0.835	**0.855**	0.844
	3	**0.858**	0.846	0.851	0.850	0.801	0.838	0.799	**0.856**	**0.858**	**0.858**	0.848	0.840
	4	**0.843**	0.840	0.836	0.827	0.842	0.833	0.798	0.835	0.814	**0.848**	0.831	0.832
	5	0.803	0.816	0.810	0.808	0.805	**0.821**	0.810	0.807	0.807	0.820	**0.822**	0.808
	6	0.819	0.813	0.814	0.812	0.818	---	0.811	**0.824**	0.815	0.821	0.817	**0.825**
	7	0.793	0.834	0.828	0.831	0.796	**0.837**	0.799	0.833	0.835	0.835	0.828	**0.838**
	8	---	**0.855**	0.844	0.848	0.717	0.850	0.842	0.851	**0.851**	0.850	0.844	0.850
0.0001	2	**0.855**	0.837	0.842	0.840	**0.852**	0.843	0.838	0.847	0.840	0.845	0.835	0.829
	3	**0.861**	0.841	0.849	0.843	0.852	0.849	0.846	0.842	0.842	**0.854**	0.839	0.836
	4	0.837	**0.844**	0.838	0.827	**0.839**	0.837	**0.839**	0.824	0.831	0.828	0.832	0.822
	5	0.811	0.815	**0.821**	0.808	0.815	0.808	0.812	**0.819**	**0.819**	0.812	0.815	0.804
	6	0.820	0.819	0.817	0.815	**0.827**	0.823	0.813	**0.826**	0.818	0.812	0.822	0.811
	7	0.837	0.836	**0.838**	0.834	**0.842**	0.835	0.828	0.833	0.831	0.835	0.834	0.828
	8	0.849	**0.854**	0.848	0.847	---	0.845	0.852	0.849	0.845	**0.853**	0.844	0.841
0.00001	2	0.827	**0.832**	---	0.829	---	---	---	**0.834**	---	0.831	---	---
	3	0.837	**0.853**	---	0.842	---	---	---	**0.843**	---	0.840	---	---
	4	0.824	**0.842**	---	**0.835**	---	---	---	0.824	---	0.827	---	---
	5	0.805	**0.827**	0.803	0.806	---	---	---	**0.810**	---	0.808	---	---
	6	**0.818**	**0.824**	0.815	0.812	---	---	---	0.810	---	0.809	---	---
	7	0.833	**0.855**	0.828	**0.840**	---	---	---	0.837	---	**0.840**	---	---
	8	0.848	**0.859**	**0.853**	0.842	---	---	---	0.849	---	0.849	---	---

TABLE 6 Results of the second experiment - median of average error of the machining process

Mean Error		Network Architecture Number											
Learning Rate	**Train Trajectories Count**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0.001	2	2.44	2.04	2.42	2.10	1.81	2.19	1.93	2.44	2.29	2.51	2.43	2.48
	3	2.38	2.46	2.47	2.55	2.47	2.59	2.24	2.48	2.64	2.51	2.58	2.58
	4	2.48	2.54	2.28	2.58	2.47	2.51	2.59	2.50	2.47	2.47	2.59	2.58
	5	2.30	2.55	2.36	2.53	2.38	2.48	2.58	2.48	2.54	2.49	2.48	2.19
	6	2.59	2.59	2.65	2.48	2.48	3.03	2.59	2.58	2.69	2.58	2.59	2.59
	7	2.49	2.59	2.63	2.61	2.05	2.53	2.30	2.59	2.59	2.59	2.59	2.59
	8	2.19	2.36	2.48	2.41	2.27	2.21	2.27	2.24	2.36	2.31	2.39	2.27
0.0001	2	2.47	2.47	2.47	2.48	2.44	2.56	1.80	2.47	2.51	2.58	2.43	2.51
	3	2.47	2.56	2.57	2.49	2.47	2.58	2.13	2.58	2.57	2.45	2.51	2.58
	4	2.58	2.47	2.48	2.59	2.59	2.59	2.40	2.48	2.60	2.58	2.48	2.58
	5	2.59	2.48	2.58	2.57	2.56	2.58	2.50	2.52	2.55	2.54	2.48	2.58
	6	2.56	2.59	2.66	2.60	2.59	2.59	2.56	2.58	2.59	2.59	2.52	2.58
	7	2.59	2.59	2.59	2.59	2.59	2.59	1.68	2.60	2.59	2.59	2.59	2.59
	8	2.19	2.48	2.25	2.32	2.71	2.47	2.16	2.48	2.48	2.43	2.42	2.41
0.00001	2	2.55	2.51	2.28	2.47	3.15	5.15	5.47	2.50	3.21	2.55	4.59	427.33
	3	2.56	2.58	2.21	2.47	3.10	5.10	4.14	2.47	2.86	2.53	5.58	448.67
	4	2.57	2.58	2.47	2.51	2.98	5.20	5.63	2.59	3.09	2.59	4.18	438.00
	5	2.58	2.54	2.62	2.48	3.06	5.25	5.77	2.48	3.06	2.47	3.51	427.33
	6	2.54	2.59	2.59	2.51	3.06	5.13	6.17	2.66	3.06	2.59	3.65	427.33
	7	2.66	2.65	2.59	2.62	2.76	4.92	5.12	2.63	2.76	2.59	3.22	427.33
	8	2.55	2.53	2.48	2.30	2.69	4.28	4.81	2.48	2.69	2.36	2.69	448.67

In the second experiment, the authors decided to investigate whether increasing the diversity of reference point trajectories in the learning dataset has a positive effect on the quality of the trained agent's match with the RPRO algorithm's behavior. Thus, it was decided to gradually increase the number of trajectories in the learning set starting with 2 and ending with 8. This time, the virtual window of the training set moved sequentially through a number of trajectories resulting in 8, 7, 6, ..., 2 combinations of trajectories, respectively. The results were again combined by calculating the corresponding medians, which are shown in Table 7. and Table 8. In addition, in each row of Table 7, both the largest values of the correlation coefficient of the number of steps, as well as its second best values, are marked in bold. It should also be noted that for this experiment the maximum number of possible episodes was reduced by 20%, resulting in a value of 400. Analyzing the data presented, it can be indicated that the three best performing architectures are those numbered 2, 8 and 10, respectively. In accordance with previous observations, their common feature is the use of sigmoidal activation functions. The results also indicate that increasing the diversity of reference point paths in the learning dataset had no noticeable effect on the quality of the trained agent's performance. Thus, it can be conjectured that as few as two paths are sufficient for an agent to learn decision-making to a satisfactory degree while optimizing the machining process. It can be also

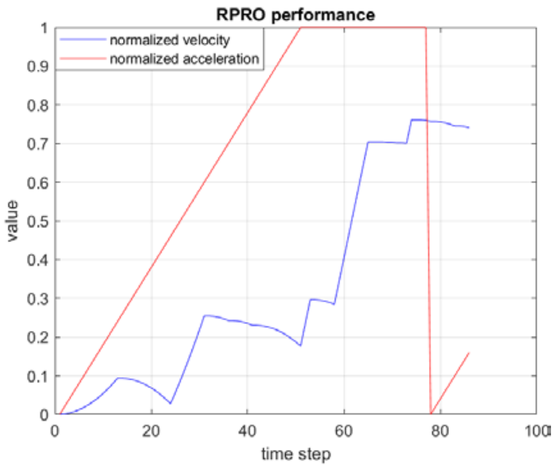

FIGURE 3 RPRO algorithm performance on exemplary machining process

observed that the choice of a learning rate coefficient equal to 1e-5 led to a disturbance in the stability of the learning process by, in several cases, failing to successfully complete a single trial. This clearly indicates that it should be avoided, such small values of this coefficient in the studied case.

At the end of this section is a comparison of the simulation run of an example processing performed by the RPRO algorithm and in the way proposed by the authors on the Figure 3 and Figure 4, respectively. The values shown on them are both the normalized spindle velocity and its normalized acceleration in successive time steps. The presented case shows the observations described so far, where the solution proposed by the authors creates a g-code that improves the machining

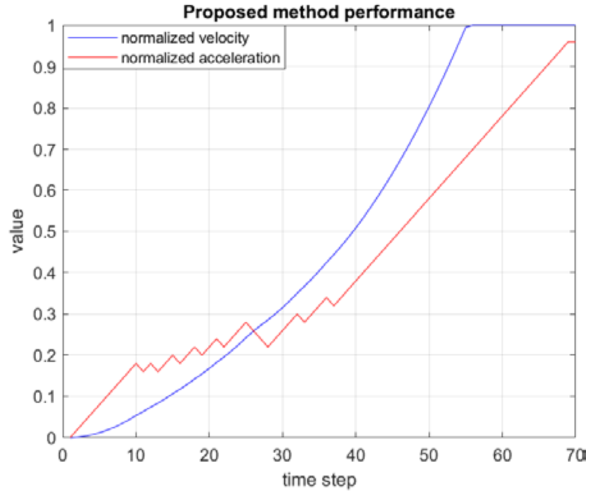

FIGURE 4 Proposed method performance on exemplary machining process

process (realizing it in fewer steps and with higher accuracy). The RPRO algorithm completed the task during 86 steps achieving an average accuracy of 47.94 micrometers , while the proposed solution completed the task during 69 time steps achieving an average accuracy of 32.98 micrometers. Another interesting aspect is the way in which the dynamics of the spindle's movement changes, namely, unlike the RPRO algorithm, which pursued the highest possible acceleration, only to later reduce it just as sharply and reduce the speed as well, the solution obtained by the algorithm proposed by the authors relies on the gradual acceleration of the spindle until it reaches its maximum speed, taking into account minor adjustments that allow it to hit the reference points more accurately.

4. CONCLUSIONS

The authors presented a novel approach to the problem of optimizing the motion dynamics of a cnc machine. It consisted of using a deep learning algorithm with reinforcement to map the operation of the RPRO algorithm used in industry. The presented solution achieved very good results - it mapped the operation of the RPRO algorithm to a satisfactory degree, and, in addition, it accelerated the machining process and provided a much higher accuracy (much lower average error). However, in order for the proposed solution to be put into industrial use, some improvements still need to be made and movement in multiple axes needs to be integrated simultaneously, which will be the main focus of the authors' further research. Attention will also be paid to optimizing other parameters of the machining process such as smoothing of acceleration and deceleration cycles, which has a direct impact on extending both machine service intervals and tool life.

REFERENCES

[1] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, “Time-optimal control of robotic manipulators along specified paths,” *Int J Rob Res*, vol. 4, no. 3, pp. 3–17, 1985.

[2] J. H. Lee, Y. Liu, and S.-H. Yang, “Accuracy improvement of miniaturized machine tool: geometric error modeling and compensation,” *Int J Mach Tools Manuf*, vol. 46, no. 12–13, pp. 1508–1516, 2006.

[3] S. Z. Mansour and R. Seethaler, “Feedrate optimization for computer numerically controlled machine tools using modeled and measured process constraints,” *J Manuf Sci Eng*, vol. 139, no. 1, p. 011012, 2017.

[4] Q. Bi, N. Huang, C. Sun, Y. Wang, L. Zhu, and H. Ding, “Identification and compensation of geometric errors of rotary axes on five-axis machine by on-machine measurement,” *Int J Mach Tools Manuf*, vol. 89, pp. 182–191, 2015.

[5] X. Li, H. Zhao, X. Zhao, and H. Ding, “Interpolation-based contour error estimation and component-based contouring control for five-axis CNC machine tools,” *Sci China Technol Sci*, vol. 61, pp. 1666–1678, 2018.

[6] B. Kwiatkowski, T. Kwater, D. Mazur, and J. Bartman, “An offline application that determines themaximum accuracy of the realization ofreference points from G-code for givenparameters of CNC machine dynamics,” *Bulletin of the Polish Academy of Sciences Technical Sciences*, pp. e147345–e147345.

[7] J. M. Langeron, E. Duc, C. Lartigue, and P. Bourdet, “A new format for 5-axis tool path computation, using Bspline curves,” *Computer-Aided Design*, vol. 36, no. 12, pp. 1219–1229, 2004.

[8] Y. Sun, S. Sun, J. Xu, and D. Guo, “A unified method of generating tool path based on multiple vector fields for CNC machining of compound NURBS surfaces,” *Computer-Aided Design*, vol. 91, pp. 14–26, 2017.

[9] M. Chen and Y. Sun, “A moving knot sequence-based feedrate scheduling method of parametric interpolator for CNC machining with contour error and drive constraints,” *The International Journal of Advanced Manufacturing Technology*, vol. 98, pp. 487–504, 2018.

[10] B. Pękala, E. Rak, B. Kwiatkowski, A. Szczur, and R. Rak, “The use of concave and convex functions to optimize the feed-rate of numerically controlled machine tools,” in *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, IEEE, 2020, pp. 1–8.

[11] M. Rahaman, R. Seethaler, and I. Yellowley, “A new approach to contour error control in high speed machining,” *Int J Mach Tools Manuf*, vol. 88, pp. 42–50, 2015.

[12] S. D. Timar, R. T. Farouki, T. S. Smith, and C. L. Boyadjieff, “Algorithms for time–optimal control of CNC machines along curved tool paths,” *Robot Comput Integr Manuf*, vol. 21, no. 1, pp. 37–53, 2005.

[13] J. Dong and J. A. Stori, “A generalized time-optimal bidirectional scan algorithm for constrained feed-rate optimization,” 2006.

[14] Z. Shiller and H.-H. Lu, “Robust computation of path constrained time optimal motions,” in *Proceedings., IEEE International Conference on Robotics and Automation*, IEEE, 1990, pp. 144–149.

[15] S. D. Timar and R. T. Farouki, “Time-optimal traversal of curved paths by Cartesian CNC machines under both constant and speed-dependent axis acceleration bounds,” *ROBOTICS AND COMPUTER INTEGRATED MANUFACTURING*, vol. 24, no. 1, pp. 16–31, 2008.

[16] C. Wang, X. P. Tan, S. B. Tor, and C. S. Lim, “Machine learning in additive manufacturing: State-of-the-art and perspectives,” *Addit Manuf*, vol. 36, p. 101538, 2020.

[17] A. Molina, H. Ponce, P. Ponce, G. Tello, and M. Ramírez, “Artificial hydrocarbon networks fuzzy inference systems for CNC machines position controller,” *The International Journal of Advanced Manufacturing Technology*, vol. 72, pp. 1465–1479, 2014.

[18] T. Kar, N. K. Mandal, and N. K. Singh, “Multi-response optimization and surface texture characterization for CNC milling of inconel 718 alloy,” *Arab J Sci Eng*, vol. 45, pp. 1265–1277, 2020.

[19] S. Datta, S. S. Mahapatra, B. C. Routara, and A. Bandyopadhyay, “The fuzzy inference system approach to a multi-performance characteristic index for surface quality improvement in CNC end milling,” *International Journal of Experimental Design and Process Optimisation*, vol. 2, no. 3, pp. 265–282, 2011.

[20] M. F. Alam, M. Shtein, K. Barton, and D. J. Hoelzle, “Autonomous manufacturing using machine learning: A computational case study with a limited manufacturing budget,” in *International Manufacturing Science and Engineering Conference*, American Society of Mechanical Engineers, 2020, p. V002T07A009.

[21] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, “Deep reinforcement learning: A brief survey,” *IEEE Signal Process Mag*, vol. 34, no. 6, pp. 26–38, 2017.

[22] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[23] G. Lample and D. S. Chaplot, “Playing FPS games with deep reinforcement learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[24] D. Kalandyk, “Reinforcement learning in car control: A brief survey,” in *2021 Selected Issues of Electrical Engineering and Electronics, WZEE 2021*, 2021. doi: 10.1109/WZEE54157.2021.9576838.

[25] J. Marquez, C. Sullivan, R. M. Price, and R. C. Roberts, “Hardware-in-the-Loop Soft Robotic Testing Framework using an Actor-Critic Deep Reinforcement Learning Algorithm,” *IEEE Robot Autom Lett*, 2023.

[26] Q. Su, B. Li, C. Wang, C. Qin, and W. Wang, “A power allocation scheme based on deep reinforcement learning in HetNets,” in *2020 international conference on computing, networking and communications (ICNC)*, IEEE, 2020, pp. 245–250.

[27] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, “Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning,” *IEEE Internet Things J*, vol. 6, no. 3, pp. 4005–4018, 2018.

[28] K. Li, Y. Zhang, K. Li, and Y. Fu, “Adversarial feature hallucination networks for few-shot learning,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13470–13479.

[29] X. Wang *et al.*, “Dynamic scheduling of tasks in cloud manufacturing with multi-agent reinforcement learning,” *J Manuf Syst*, vol. 65, pp. 130–145, 2022.

[30] G. Velusamy and R. Lent, “Evaluating reinforcement learning methods for bundle routing control,” in *2019 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, IEEE, 2019, pp. 1–4.

[31] A. M. Seid, G. O. Boateng, S. Anokye, T. Kwantwi, G. Sun, and G. Liu, “Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: A deep reinforcement learning approach,” *IEEE Internet Things J*, vol. 8, no. 15, pp. 12203–12218, 2021.

[32] C. Li, P. Zheng, Y. Yin, B. Wang, and L. Wang, “Deep reinforcement learning in smart manufacturing: A review and prospects,” *CIRP J Manuf Sci Technol*, vol. 40, pp. 75–101, 2023.

[33] B. Fernandez-Gauna, I. Ansoategui, I. Etxeberria-Agiriano, and M. Graña, “Reinforcement learning of ball screw feed drive controllers,” *Eng Appl Artif Intell*, vol. 30, pp. 107–117, 2014.

[34] S. Baer, J. Bakakeu, R. Meyes, and T. Meisen, “Multi-agent reinforcement learning for job shop scheduling in flexible manufacturing systems,” in *2019 Second International Conference on Artificial Intelligence for Industries (AI4I)*, IEEE, 2019, pp. 22–25.

[35] Y.-C. Wang and J. M. Usher, “Application of reinforcement learning for agent-based production scheduling,” *Eng Appl Artif Intell*, vol. 18, no. 1, pp. 73–82, 2005.

[36] T. Zhou, D. Tang, H. Zhu, and Z. Zhang, “Multi-agent reinforcement learning for online scheduling in smart factories,” *Robot Comput Integr Manuf*, vol. 72, p. 102202, 2021.

[37] X. Jing, X. Yao, M. Liu, and J. Zhou, “Multi-agent reinforcement learning based on graph convolutional network for flexible job shop scheduling,” *J Intell Manuf*, pp. 1–19, 2022.

[38] K. Chang, S. H. Park, and J.-G. Baek, “AGV dispatching algorithm based on deep Q-network in CNC machines environment,” *Int J Comput Integr Manuf*, vol. 35, no. 6, pp. 662–677, 2022.

[39] J. Yao, B. Lu, and J. Zhang, “Tool remaining useful life prediction using deep transfer reinforcement learning based on long short-term memory networks,” *The International Journal of Advanced Manufacturing Technology*, pp. 1–10, 2022.

[40] R. Lamprecht, F. Wurst, and M. F. Huber, “Reinforcement learning based condition-oriented maintenance scheduling for

flow line systems," in *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, IEEE, 2021, pp. 1–7.
[41] M. L. R. Rodríguez, S. Kubler, A. de Giorgio, M. Cordy, J. Robert, and Y. Le Traon, "Multi-agent deep reinforcement learning based Predictive Maintenance on parallel machines," *Robot Comput Integr Manuf*, vol. 78, p. 102406, 2022.
[42] A. Mishra and V. S. Jatti, "Reinforcement learning based approach for the optimization of mechanical properties of additively manufactured specimens," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, pp. 1–9, 2023.
[43] D. Limoge, A. Sunstrom, V. Pinskiy, and M. Putman, "Defending Industrial Production Using AI Process Control," in *2020 IEEE Systems Security Symposium (SSS)*, IEEE, 2020, pp. 1–4.
[44] Y. Zhang, Y. Li, and K. Xu, "Reinforcement learning–based tool orientation optimization for five-axis machining," *The International Journal of Advanced Manufacturing Technology*, vol. 119, no. 11–12, pp. 7311–7326, 2022.
[45] Q. Xiao, C. Li, Y. Tang, and L. Li, "Meta-reinforcement learning of machining parameters for energy-efficient process control of flexible turning operations," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 1, pp. 5–18, 2019.
[46] W. Li *et al.*, "A novel milling parameter optimization method based on improved deep reinforcement learning considering machining cost," *J Manuf Process*, vol. 84, pp. 1362–1375, 2022.
[47] H. A. Taha, S. Yacout, and Y. Shaban, "Deep Reinforcement Learning for autonomous pre-failure tool life improvement," *The International Journal of Advanced Manufacturing Technology*, vol. 121, no. 9–10, pp. 6169–6192, 2022.
[48] X. Liu and D. Chang, "An Improved Method for Optimizing CNC Laser Cutting Paths for Ship Hull Components with Thicknesses up to 24 mm," *J Mar Sci Eng*, vol. 11, no. 3, p. 652, 2023.
[49] M. F. Alam, M. Shtein, K. Barton, and D. Hoelzle, "Reinforcement learning enabled autonomous manufacturing using transfer learning and probabilistic reward modeling," *IEEE Control Syst Lett*, vol. 7, pp. 508–513, 2022.
[50] L. Jonath, J. Luderich, J. Brezina, A. M. Gonzalez Degetau, and S. Karaoglu, "Improving the Thermal Behavior of High-Speed Spindles Through the Use of an Active Controlled Heat Pipe System," in *International Conference on Thermal Issues in Machine Tools*, Springer, 2023, pp. 203–218.
[51] G. Singh and R. Sharma, "Cnc Machine Handling for Holes Servicing through Programming," 2023.
[52] F. Jaensch, A. Csiszar, J. Sarbandi, and A. Verl, "Reinforcement learning of a robot cell control logic using a software-in-the-loop simulation as environment," in *2019 Second International Conference on Artificial Intelligence for Industries (AI4I)*, IEEE, 2019, pp. 79–84.
[53] Q. Chen, B. Heydari, and M. Moghaddam, "Leveraging task modularity in reinforcement learning for adaptable industry 4.0 automation," *Journal of Mechanical Design*, vol. 143, no. 7, p. 071701, 2021.
[54] A. A. Apolinarska *et al.*, "Robotic assembly of timber joints using reinforcement learning," *Autom Constr*, vol. 125, p. 103569, 2021.
[55] B. Li, H. Zhang, P. Ye, and J. Wang, "Trajectory smoothing method using reinforcement learning for computer numerical control machine tools," *Robot Comput Integr Manuf*, vol. 61, p. 101847, 2020.
[56] Y. Jiang, J. Chen, H. Zhou, J. Yang, P. Hu, and J. Wang, "Contour error modeling and compensation of CNC machining based on deep learning and reinforcement learning," *The International Journal of Advanced Manufacturing Technology*, pp. 1–20, 2022.
[57] C. Dripke, S. Höhr, A. Csiszar, and A. Verl, "A concept for the application of reinforcement learning in the optimization of CAM-generated tool paths," in *Machine Learning for Cyber Physical Systems: Selected papers from the International Conference ML4CPS 2016*, Springer, 2017, pp. 1–8.
[58] V. Samsonov, E. Chrismarie, H.-G. Köpken, S. Bär, D. Lütticke, and T. Meisen, "Deep representation learning and reinforcement learning for workpiece setup optimization in CNC milling," *Production Engineering*, vol. 17, no. 6, pp. 847–859, 2023.
[59] Z. Lin, T. Chen, Y. Jiang, H. Wang, S. Lin, and M. Zhu, "B-Spline-Based Curve Fitting to Cam Pitch Curve Using Reinforcement Learning.," *Intelligent Automation & Soft Computing*, vol. 36, no. 2, 2023.
[60] D. Kalandyk, B. Kwiatkowski, and D. Mazur, "Application of Mamdani Fuzzy Logic Inference System to Optimise CNC Machine Motion Dynamics," in *IEEE International Conference on Fuzzy Systems*, 2023. doi: 10.1109/FUZZ52849.2023.10309802.