



# Optimisation of a Workpiece Clamping Position with Reinforcement Learning for Complex Milling Applications

Chrismarie Enslin<sup>1</sup> , Vladimir Samsonov<sup>1</sup>  , Hans-Georg Köpken<sup>2</sup>,  
Schirin Bär<sup>1</sup>, and Daniel Lütticke<sup>1</sup> 

<sup>1</sup> Institute for Information Management in Mechanical Engineering,  
RWTH Aachen University, Aachen, Germany  
vladimir.samsonov@ima.rwth-aachen.de

<sup>2</sup> Digital Industries, Siemens AG, Erlangen, Germany

**Abstract.** Fine-tuning and optimisation of production processes in manufacturing are often conducted with the help of algorithms from the field of Operations Research (OR) or directly by human experts. Machine Learning (ML) methods demonstrate outstanding results in tackling optimisation tasks within the research field referred to as Neural Combinatorial Optimisation (NCO). This opens multiple opportunities in manufacturing for learning-based optimisation solutions. In this work, we show a successful application of Reinforcement Learning (RL) to the task of workpiece (WP) clamping position and orientation optimisation for milling processes. A carefully selected clamping position and orientation of a WP are essential for minimising machine tool wear and energy consumption. With the example of 3- and 5-axis milling, we demonstrate that a trained RL agent can successfully find a near-optimal orientation and positioning for new, previously unseen WPs. The achieved solution quality is comparable to alternative optimisation solutions relying on Simulated Annealing (SA) and Genetic Algorithms (GA) while requiring orders of magnitude fewer optimisation iterations.

**Keywords:** Reinforcement Learning · Supervised learning · Manufacturing · Process optimisation · Milling optimisation · Tool path

## 1 Introduction

This study looks into the adaptation of learning-based methods to an optimisation task in the context of mechanical engineering. The object of investigation is applying RL for optimising WP clamping position and orientation in a Computer Numerical Control (CNC) milling machine. The milling process involves the removal of material from the WP with a rotary cutting tool. The considered CNC milling machine belongs to a widely used type of milling machines with a rotary table and a swivelling spindle head allowing complex movements of the

cutting tool related to the WP. This enables the production of a wide variety of complex WP geometries on a single CNC milling machine.

Designing a new CNC milling process is a laborious task relying heavily on human expertise. Firstly, a Numerical Control (NC) program needs to be created based on the WP geometry, the chosen processing technology and the type of CNC machine. The resulting NC program defines the toolpath relative to the WP. A second step is the definition of a WP clamping position and orientation in the working space of the CNC machine. Different WP clamping positions and orientations result in different movements of the machine axes. Therefore, certain clamping positions and orientations will require higher accelerations of the heavy machine axes and a higher number of axes movements, directly influencing machine wear, machining accuracy and energy efficiency.

The standard approach to determine a suitable WP clamping position and orientation relies solely on human expertise gathered through experience. The preceding work [18] demonstrates the concept of formalising the task of finding the optimal WP clamping position and orientation for the milling process as an optimisation problem with subsequent use of RL. In this study, we considerably enhance the proposed approach and demonstrate the capability of the proposed method in addressing more challenging milling tasks while improving the quality of solutions with fewer search iterations. A direct comparison to alternative optimisation approaches, such as GA [14] and SA [20], demonstrates the capability of the proposed RL-solution to yield comparable results for WPs not seen during training, while requiring considerably fewer optimisation iterations.

## 2 ML Applications in Mechanical Engineering

Learning-based and data-driven methods are widely deployed to drive progress in the field of smart production and manufacturing [4, 15]. ML is considered the key enabling technology for further cost savings, quality improvement, and minimisation of waste in applications relying on the use of cutting processes along with heuristic optimisation approaches. At the same time, milling and turning receive the most attention from the research community [5].

Multiple studies concentrate on adopting learning-based methods for condition monitoring and machine tool diagnosis to enhance cutting processes. Wu et al. [22] demonstrate the applicability of simple regression models based on Random Forest (RF), Support Vector Machine (SVM) or Multilayer Perceptron (MLP) for the prediction of tool wear. Kothuru et al. [11] investigate the possibility of using SVM prediction models to estimate the condition of a cutting tool using only audible signals.

More advanced supervised ML models capable of processing sequential data demonstrated their efficiency for condition monitoring and quality prediction. Wang et al. [21] utilise recurrent predictive models on time series for tool wear monitoring. In this study, a Gated Recurrent Unit (GRU) predictive model demonstrates superior performance compared to conventional methods. For a similar Use Case (UC), Serin et al. [19] introduce the use of RL in combination

with an LSTM-based (Long Short-term Memory) control system. The LSTM-model is used as a memory base and can suggest optimal cutting parameters to the RL agent. The study of Yuan et al. [24] emphasises the advantages of enhancing recurrent predictive models with attention mechanisms for quality prediction in complex production processes.

RL methods are often used in manufacturing engineering for planning, controlling and iterative optimisation of the production process. Xanthopoulos et al. [23] deploy an RL agent for learning joint production control and maintenance strategies on a deteriorating production system. A trained RL agent can suggest maintenance schedules and production plans that are superior to methods currently implemented in practice. Pol et al. [16] use decentralised RL agents to perform online scheduling in flexible production systems with the advantage of generalising to uncertain situations that deviate from the plan. Meyes et al. [13] demonstrate the use of an RL agent for sample-efficient optimisation of a heavy plate rolling process. The application UC considered in the study contains many process-, material- and machine parameters interacting with each other. These different interactions of parameters produce different product quality levels, which relate to the height and the grain size of the finished products. A trained RL agent can estimate suitable pass schedules for heavy plate rolling to achieve the desired material characteristics.

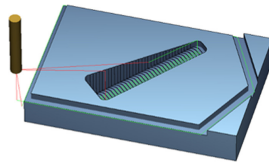
Bhinge et al. [1] demonstrate the importance of the tool path for the total energy consumption of the machine tool. Rangarajan et al. [17] emphasise the importance of the WP orientation in milling to minimise the processing time and drive loads. Campatelli et al. [3] propose a mathematical model to reduce energy consumption through optimal WP placement in the milling process. However, the proposed approach is applicable only for the finishing operations and is not capable of axis collisions avoidance. To the best of our knowledge, only the study from Samsonov et al. [18] demonstrates the application of RL methods to the task of optimal WP positioning in a machine tool.

### 3 Problem Statement

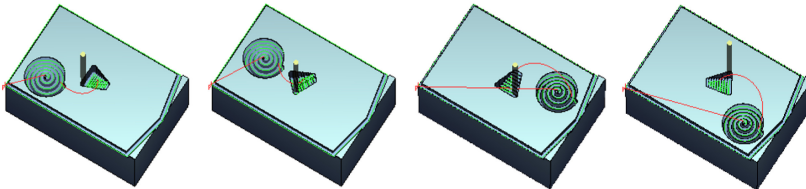
This study aims to find a near-optimal clamping position and orientation for a previously unseen WP in a CNC machine with RL. An optimal position refers to a placement that minimises the acceleration and the distance travelled in the axis directions of the machine while milling the WP. This placement should also accommodate the avoidance of all possible collisions of CNC machine parts caused by the movements of the machine axes. The task of finding WP position and orientation can be formalised as a continuous optimisation problem maximising the objective function equal to the reward function discussed in Sect. 4.1.

Two UCs of CNC milling are investigated in this work. Firstly, a simple WP geometry is considered with a groove along the perimeter of the part and the milling slot on top of it (see Fig. 1). Given shape requires a 3-axis milling process with possible movements in a front-to-back (X-axis), side-to-side (Y-axis), as well as up-and-down (Z-axis) directions by the cutting tool. A milling

slot is a widespread feature involving multiple changes of the tool movement direction. Having the groove along the perimeter of the WP increases the chance of axes collisions if the WP position and orientation are not selected correctly. A further element of simplicity is that the machine coordinate system and the WP coordinate systems coincide. Changing the orientation of the slot (slot angle) allows the generation of a set of WP geometries for training and evaluation of the proposed RL solution. This WP geometry is considered to a certain extent in [18] and is used in this work to demonstrate the achieved improvements on the search efficiency of the solution space.



**Fig. 1.** Visualisation of a 3-axis WP, slot angle =  $45^\circ$



(a) Slot angle =  $0^\circ$ , (b) Slot angle =  $45^\circ$ , (c) Slot angle =  $90^\circ$ , (d) Slot angle =  $210^\circ$ ,  
 Tower Z = 60mm, Tower Z = 60mm, Tower Z = 235mm, Tower Z = 235mm,  
 Tower X = 55mm    Tower X = 100mm    Tower X = 130mm    Tower X = 55mm

**Fig. 2.** Visualisations of 5-axis milling WPs

The second UC, and focus of the current study, is based on a 5-axis milling process and decoupling of the WP and machine coordinate systems. Apart from the milling slot and groove along the WP perimeter included in the first UC, a tower-shaped surface is added on the top of the WP (see Fig. 2 for various examples of WP geometries). The introduced tower feature is a complex spiral-formed shape with inclined sides towards the center. This makes the feature fairly representative for complex milling operations, requiring coordinated movements of multiple axes and constant change in the moving speed/accelerations. The milling process of considered 5-axis WP geometry involves a front-to-back movement (Z-axis), a side-to-side movement (X-axis), an up-and-down movement (Y-axis), a rotation movement along the Y-axis and pitch movement from

side-to-side of the cutting tool relative to the WP. In this case, the RL agent needs to learn how to handle non-trivial axis collision avoidance patterns and a different WP design. To vary the shape of the WP, the slot angle can be rotated  $360^\circ$  and the tower position can be moved continuously along the top and bottom of the WP. The NC program created to mill the WP is calibrated such that the slot and the tower will not overlap. Modifications of the slot and tower features change the energy exertion and wear on the main axes of the milling machine and the optimal placement is influenced.

The 3-axis UC provides a good framework to test modifications of the RL solution, with the goal of finally transferring and testing the results on the more complicated 5-axis UC.

## 4 RL Experiment Setup

The WP clamping position and orientation optimisation task is formalised as a fully observable Markov Decision Process (MDP). The RL agent is allowed to iteratively move the WP in the machine's working space to improve the reward from the WP clamping position and orientation. In the following sub-chapters the state space, action space and reward function for the RL agent are discussed, along with search space efficiency implementations. The training and validation scheme for the RL agent is outlined and the generation of WP data and an efficient way of using the milling simulation data by training an ML model is explained.

### 4.1 State Space, Action Space and Reward Function

The position of the WP in the machine space is described by axes coordinates  $(X, Y, Z)$  and the rotation angle of the WP with respect to the orientation of the machine. The RL agent can choose the location and orientation of the WP every time an action is required. For the 3-axis machine, the only possible variation in the shape of the WP is in terms of the milling slot angle. Therefore, each WP position and shape is uniquely captured by  $(X, Y, \textit{Rotation Angle}, \textit{Slot Angle})$ . The 5-axis machine includes the possibility of milling a tower, as well as a slot, and a combination of these two features uniquely specifies the WP. The parameters to describe the WP placement and shape to an RL agent in this case are  $(Z, X, Y, \textit{Rotation Angle}, \textit{Tower Z}, \textit{Tower X}, \textit{Slot Angle})$ . For the RL agent to know what kind of WP is currently being processed, it is important to provide this unique parameter set explicitly or to determine a proxy for the WP shape that could be provided.

After placement of the WP in the machine's working space at the specified coordinates, with the specified orientation and including the unique features of the WP, a milling process is carried out. From this milling process, the sum of the squared accelerations  $(e_Z, e_X, e_Y)$  and distances travelled along every machine axis  $(d_Z, d_X, d_Y)$  are recorded for further evaluation of the current WP placement and orientation.

The tuples for the state-action-reward representation found in the 5-axis UC are briefly summarised in Table 1. The state space includes the unique WP parameters, the location and orientation parameters in the working space of the milling machine, the resulting squared accelerations and distances travelled, the collision switch (*limit Z*) and the number of *steps left* before reaching the step limit in the episode. The latter parameter helps the agent with planning its search strategy and choosing step sizes while exploring the solution space. The action space consists of the changes to the WP location in terms of the machine coordinates, as well as the rotation angle of the WP and a decision to terminate the episode or not.

**Table 1.** Summary of the main parameters of the optimisation task.

State	$(Z, X, Y, \textit{Rotation Angle}, \textit{Tower X}, \textit{Tower Z}, \textit{Slot Angle}, d_Z, d_X, e_Z, e_X, \textit{limit Z}, \textit{steps left})$
Action	$(\Delta Z, \Delta X, \Delta Y, \Delta \textit{Rotation Angle}, \textit{Stop})$
Reward	$R = 0.7e + 0.3d$

The reward function provides feedback on the quality of the suggested placement of the WP in the machine space by the RL agent. Maximising the reward ensures WP positions and orientations close to the optimum. The 3-axis and 5-axis milling processes have similar reward functions, as the concept of minimising wear on the machine and energy consumption remains the same. Therefore, the reward function includes a component representing the sum of squared acceleration ( $e$ ) and the sum of distance travelled ( $d$ ) on the main axes. For 3-axis milling, the directions of travel are the X- and Y-axis, with the X-axis representing the heavier axis. For 5-axis milling the directions of travel include an X-axis, a Y-axis and a Z-axis. Here the Z-axis is the heaviest axis and the movement along the Y-axis is not included in the reward function, as the movement is minimal in this UC. To limit the wear of the machine as much as possible, the movement of the heavier axes should be one of the components minimised by the RL agent.

The squared acceleration ( $e$ ) is given higher importance than distance travelled ( $d$ ) by domain experts in finding a near-optimal position of the WP and therefore the reward function is formulated as:

$$R = \begin{cases} 0.7e + 0.3d & \text{no axis collision} \\ -1 & \text{axis collision.} \end{cases} \quad (1)$$

The weights for the optimisation function are also chosen by domain experts and reflect specific industry needs. If these weights were to be changed the RL agent would have to be retrained, but the optimisation problem would be formulated in the same way.

The terms concerning acceleration ( $e$ ) and distance ( $d$ ) are both accumulated terms across the different machine axes and over the entire WP milling run.

The components  $e_Z$  and  $d_Z$  are weighted twice as heavy as  $e_X$  and  $d_X$  in the combined reward components  $e$  and  $d$ , as advised by domain experts because the movement of the  $Z$ -axis should be avoided. The combined components are:

$$e = 2e_Z + e_X \quad (2)$$

$$d = 2d_Z + d_X, \quad (3)$$

where  $e_Z$ ,  $e_X$ ,  $d_Z$  and  $d_X$  represent normalised terms over the minimum and maximum observed values from multiple experiments. The normalisation function negates the  $e$ - and  $d$ -values, such that lower values of  $e$  and  $d$  will result in a higher reward. If any axis collisions are encountered, the lowest possible reward is returned. The intention of the optimisation algorithm should firstly be to learn collision avoidance, and thereafter an optimal WP placement.

## 4.2 Search Efficiency Modifications

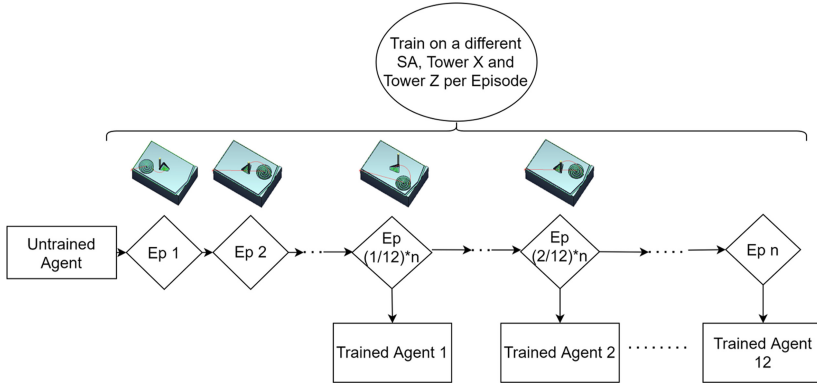
To reduce the number of optimisation iterations, we introduce several changes to the search process established in [18]. Firstly, for the 3-axis UC, we **extend the maximum range of the WP position change per step** (step size) along the X-axis, Y-axis and the rotation angle for orientation, from 40 mm, 40 mm and 35°, respectively, to 800 mm, 800 mm and 360°. This larger step size allows the RL agent to move the WP into any position within the working space of the milling machine. Similarly, for the 5-axis UC, we allow the RL agent to step into any position in the work space. Therefore, the step sizes are 300 mm, 200 mm and 50 mm for the Z-, X- and Y-axis, respectively, as well as a full 360° WP rotation.

Secondly, instead of assuming that the RL agent has to complete an entire episode before continuing to the next one, we introduce an additional action dimension referred to as **early stopping**. Early stopping allows the RL agent to stop the search process at any iteration step as soon as it assumes the current WP position is suitable. This updates the action space to  $(\Delta Z, \Delta X, \Delta Y, \Delta Rotation Angle, Stop)$ .

Thirdly, we switch from a dense reward calculated after every iteration of the search process to a **sparse reward**, returned only when the RL agent chooses to stop the search process or when the maximum allowed number of steps per search episode is reached. A sparse reward forces the RL agent to step to the optimum and terminate the episode as quickly as possible, to start collecting rewards. The sparse reward is a prerequisite for the early stopping implementation. The dense reward is not compatible with early stopping, since longer episodes generate higher cumulative rewards, incentivising the RL agent not to interrupt the episodes earlier.

## 4.3 RL Agent Training and Validation

The RL training routine is represented in Fig. 3. For every training episode in the 5-axis UC, a new WP is generated with a random slot angle, tower position Z



**Fig. 3.** Training setup for the RL agent

and tower position X. For the 3-axis UC, this is reduced to only varying the slot angle. The restrictions on these training WPs are as follows: slot angle between  $0^\circ$  and  $180^\circ$ , tower X position between 55 mm and 145 mm along the X-axis and tower Z position either 60 mm or 235 mm along the Z-axis. The decision to keep tower position Z fixed at only 60 mm or 235 mm is made to keep the tower and the slot from overlapping. To keep some WPs aside, purely for validation, the slot angles between  $40^\circ$  and  $50^\circ$  and tower X position between 90 mm and 100 mm are disallowed for training. The allowance of only two tower Z positions makes it impossible to keep certain tower Z positions aside for validation, therefore both positions are available for training and validation.

In the training procedure, episodes of different maximal lengths are investigated. The combination of larger step sizes and a sparse reward leads to a reduction in the maximum episode length required to find a near-optimal positioning of a WP. This drastically reduces the step count from 110 steps per episode to two steps per episode for 3-axis and one step per episode for 5-axis. These episodes have the option to terminate early if the agent believes it has reached the optimum. Therefore, an episode can terminate immediately after initialisation, if the agent finds that the initialisation position is an optimal position. All 5-axis RL agents are trained for 300.000 steps and the 3-axis RL agents are trained for 100.000 steps and this is repeated for three different random seeds in each case.

Throughout the training process 12 evaluation phases are equally spaced between training episodes, to account for the possibility that an intermediate version of the RL agent might be superior in performance to the RL agent in a later stage of training. During each evaluation phase, the current version of the trained RL agent for a given run solves the WP positioning task for 20 different initialisation points (WP positions) for each WP involved in the evaluation. This allows for the testing of the overall robustness and consistency of the evaluation scheme.



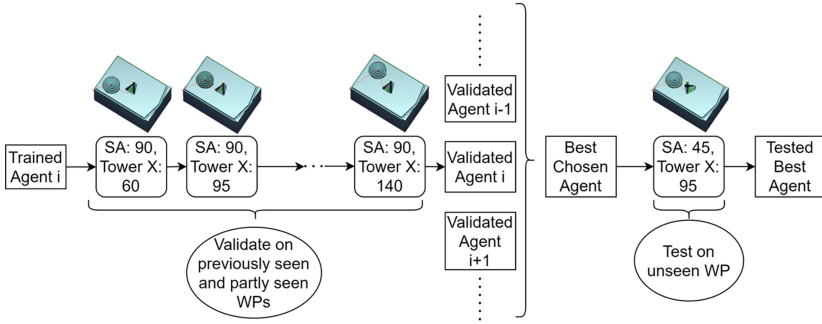


Fig. 4. Evaluation setup for the RL agent

The evaluation scenario for monitoring of the intermediate RL performance for the 3-axis UC is based on 11 WPs with slot angles that have been **seen** during training and are equally spaced throughout the  $180^\circ$  of allowed slot angles. The version of the RL agent from the evaluation phase with the best average reward is selected as a trained RL agent and is finally validated on the completely **unseen** WP with a slot angle of  $45^\circ$ . The rewards achieved on the unseen WP are used for the final estimation of the quality and generalisation ability of the RL agent, but it is not involved in choosing the best RL agent, as this can be seen as data leakage.

The 5-axis RL agent is regularly evaluated on 94 WPs with a combination of 6 different slot angles, 8 different tower X positions and the two available Z positions. These validation WPs are **seen**, or **partially seen**, during training, but the WP with a  $45^\circ$  slot angle and a tower located at the X coordinate of 95mm is kept away from the evaluation process. Similar to the 3-axis approach, the version of the RL agent from the evaluation phase with the best average reward is selected for the final validation on the **unseen** WP with slot angle  $45^\circ$  and tower X position of 95mm. The validation scheme for the 5-axis UC is summarised in Fig. 4.

The meta-heuristics GA [14] and SA [20] are used as baseline models for comparison to the RL agents. These heuristics are popular methods for solving optimisation problems and GA is often used in the context of production process optimisation as seen in [5]. Each meta-heuristic run is given 100.000 iterations to find the solution to the WP positioning task. Analogous to the RL evaluation, 20 independent initialisations and three random seeds are used. This results in 60 independent evaluation runs for each solution approach. The reward function is used as a cost function for the optimisation heuristics.

The RL agent used for the 3-axis and 5-axis UCs is the Soft Actor-Critic (SAC) introduced in [8]. In our study the stable baselines [10] implementation is used, to make it easily comparable to other RL implementations. The experiments are all performed in docker containers [12] for full reproducibility of experiments. The entire implementation of the milling simulation is done as an

OpenAI Gym environment [2]. The meta-heuristics are implementations from the mlrose package [9].

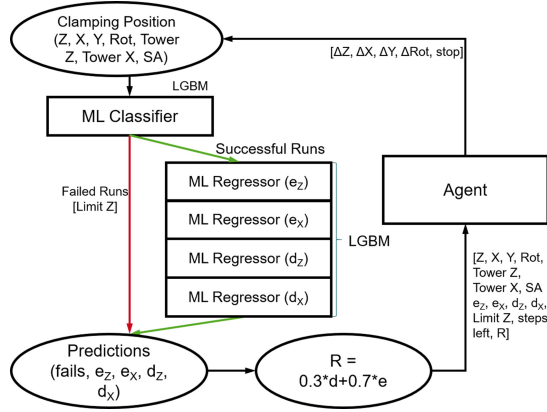
#### 4.4 Data Generation and Approximation of the Simulation with Machine Learning

Experiments in this investigation are performed on simulations created by SinuTrain software, reproducing the 3-axis and 5-axis milling processes performed on a CNC machine. For the purpose of RL training, the simulation is time intensive. It can take up to 4s to generate the output of a successful run for the 3-axis environment, with failed runs taking up to a second to generate outputs. We develop a set of ML models closely mimicking the behaviour of the simulation. ML models predict the output of a simulation run in a fraction of a second. As a result the development process of the RL solution is considerably accelerated. It is important to note that there are two layers of separation from the CNC machine and what the RL agent is trained on, namely the SinuTrain simulation and the ML models. Therefore the accuracy of the ML models are thoroughly validated to confirm the accuracy of this method, as summarised in Table 2 and Table 3. The details of the data generation process and the 3-axis ML model is stipulated in Samsonov et al. [18].

As this study also shifts to the more complex 5-axis milling process, another ML model is required to mimic the behaviour of the SinuTrain machine simulation. The ML models are even more justified in the 5-axis UC as a successful run can take up to 15s to generate the required outcomes. For the purpose of ML model training, 83.252 data points are generated from the SinuTrain simulation. The ML model ensemble in this study consists of five gradient boosting models (a success/fail classifier and a regression model for each of  $e_Z$ ,  $e_X$ ,  $d_Z$  and  $d_X$ ), fitted with the LightGBM package [7]. LightGBM is known for providing fast and efficient training of gradient boosting models. The training process, together with the ML models and the inputs and outputs of all components are outlined in Fig. 5.

The first model in the ensemble is a classification model, which distinguishes between input combinations that lead to a successful milling process or to an axis collision. The only axis collision possible in the training data is on the Z-axis. Table 2 summarises the F1-Score and the overall accuracy of the classifier model, as well as the size of the training datasets in each class, to demonstrate the support behind the different accuracy values.

The data points that are classified as successful milling process runs continue to the four regression models. These models produce estimates for  $e_Z$ ,  $e_X$ ,  $d_Z$  and  $d_X$ , respectively. As preprocessing of the data, the offset between the Z coordinate and the Tower Z position, as well as the offset between the X coordinate and the tower X position, is calculated and used as input values. The accuracy of the four models is summarised in Table 3, where the accuracy measure is chosen to be the  $R^2$ -values.


**Fig. 5.** ML model architecture

**Table 2.** Classifier accuracy

	F1-Score	Totals
Success	0.981	34943
Limit Z	0.987	48859
Weighted average	0.984	83252

**Table 3.** Regression models accuracy

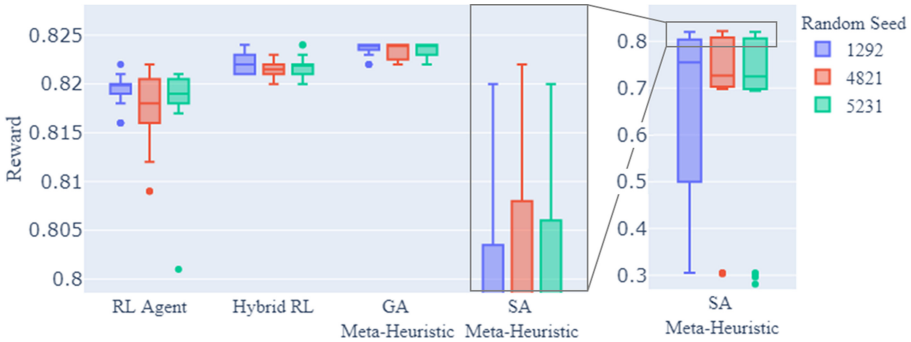
	$R^2$ -values
$e_z$	0.995
$e_x$	0.984
$d_z$	0.997
$d_x$	0.995

## 5 Experimental Results

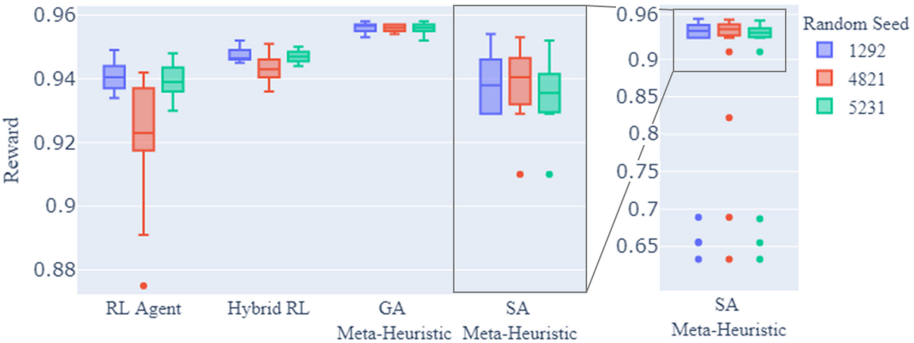
With a set of experiments, it is demonstrated to what extent the original WP positioning optimisation approach proposed by Samsonov et al. [18] can be enhanced and transferred to a more challenging 5-axis milling tasks. The use of the increased step size, early stopping and sparse reward, as described in Sect. 4.2, considerably improves the search efficiency of the trained RL agent. Averaged over all evaluation runs and random seeds, a trained RL agent solves the task with just one step in 95% of all cases for the 5-axis UC. For the 3-axis UC, the RL agent prefers to use two steps for solving the task in 90% of the observations. This allows us to introduce a hybrid approach, augmenting the best trained RL agent with a simple search heuristic to improve the absolute performance. For each WP positioning task, instead of solving the task once with an arbitrary initialisation point, the RL agent is allowed to conduct the search multiple times with different initialisation points and the solution with the best reward is selected. In our work this approach is referred to as Hybrid RL and the trained RL agent is given 20 attempts to find a near-optimal solution.

The runs of all three random seeds demonstrate the capability of the RL agent to generalise to unseen WP geometries, to consistently avoid axis collisions and to find good WP clamping positions, as seen in Fig. 6 and Fig. 7. The RL training

with random seed 4821 results in a noticeably lower performance across all 3- and 5-axis runs. Given the observed performance, the decrease is possibly related to bad initialisation weights of the policy- and value networks. The hybrid RL approach considerably boosts the absolute performance. The GA meta-heuristic demonstrates the best results, both in terms of the result consistency and the absolute reward value. The SA meta-heuristic could not match the performance of the RL-based or GA solutions in both 3-axis and 5-axis UCs. To conserve space, further evaluations of SA meta-heuristic are omitted.



**Fig. 6.** The achieved rewards for the 3-axis WP positioning search guided by the trained RL agent, hybrid RL approach, GA and SA meta-heuristics

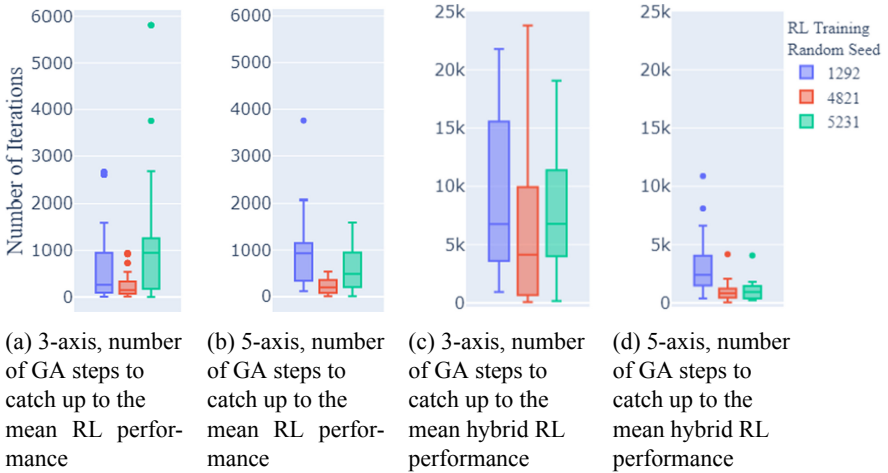


**Fig. 7.** The achieved rewards for the 5-axis WP positioning search guided by the trained RL agent, hybrid RL approach, GA and SA meta-heuristics

The conducted comparison of the absolute performance demonstrates that a well-tuned GA solver can surpass the proposed RL-based approaches. However, not only the absolute performance is essential for practical applications, the time required to find a viable solution is often a critical viability factor. A trained RL

agent needs between one to two steps per task to find a near-optimal positioning of a WP. The hybrid RL requires on average 21 and 36 steps per task for the 3-axis and 5-axis UC correspondingly.

To investigate how fast the GA meta-heuristic achieves results comparable to solutions found with the RL-based approaches, we track the performance at each interaction with the simulation environment during the meta-heuristic search. The pairwise comparison is always conducted between the GA and RL methods with identical random seeds used for the search/training. Figure 8a and Fig. 8b demonstrate that the GA meta-heuristic needs on average 660 steps for the 3-axis UC and 615 steps for the 5-axis UC to match the performance of the corresponding RL agent. GA meta-heuristic takes on average 7842 and 1756 steps on for the 3- and 5-axis UC correspondingly to surpass the performance of the hybrid RL approach (see Fig. 8c and Fig. 8d).



**Fig. 8.** The number of search steps the GA heuristic requires to match the mean performance of the trained RL agents and the hybrid RL approach

The considered evaluations are conducted on a fast ML-based environment to make extensive testing computationally viable. However, these ML models are only intended for the training of the RL agent. Building such an ML-based environment for RL training covers a finite number of WP geometries, thus keeping the general effort limited. During the deployment of a new WP in production, the search for optimal clamping parameters needs to be conducted directly in the SinuTrain simulation environment. A GA search involving 7842 steps could require 8,3h for 3-axis and similarly, a GA search involving 1756 steps could take 4,8h for 5-axis. This is in direct contrast to the possible 2s for 3-axis and 10s for 5-axis needed for the hybrid RL approach with matching performance.

## 6 Conclusion and Future Work

In this work, we extend the conceptual RL-based approach to optimise a WP clamping position in a CNC milling machine tool. We introduce several significant additions to the original approach [18]. The **first contribution** is the demonstration of the direct transfer to a more complex UC involving 5-axis milling, using the same hyperparameter set and training scheme. The RL agent successfully learns and avoids more elaborate collision patterns related to the 5-axis milling scenario, as well as consistently finds near-optimal WP clamping positions.

The **second contribution** is the introduction of early stopping, combined with larger action ranges and sparse rewards. As a result, a trained RL agent needs between one and two optimisation steps to find a near-optimal WP clamping position. Improved search efficiency allows us to make the **third contribution** by introducing the Hybrid RL approach where RL-guided optimisation search is enhanced with a simple heuristic. The agent is given 20 attempts with different initialisation points in the working space of the milling machine to solve one task. The solution with the highest reward is taken as the final WP clamping position. In this way, the final results are considerably more stable and primarily located at the upper bound of the observed RL performance, while still requiring not more than 40 optimisation steps per task.

Finally, we compare the performance of the enhanced RL-based and Hybrid RL optimisation approaches against the SA and GA meta-heuristics. Both learning-based methods surpass the SA meta-heuristic in terms of absolute performance and search efficiency. While the GA-heuristic demonstrates better absolute performance, it needs about 300 times more optimisation steps to match the performance of the RL-based approach and is two orders of magnitude less efficient compared to the Hybrid RL approach. In practice, it means that the RL-based optimisation methods are capable of solving the WP clamping position task in seconds instead of half-days of runtime in the SinuTrain simulation required by the considered meta-heuristics.

While demonstrating significant improvements, the proposed RL-based methods still remain in a prototype phase. In future work, we plan to address the current need for a handcrafted set of features describing the WP. A handcraft-free WP description can be achieved by representing the WP milling process as a set of vectors covering the change of the relative position, speed and acceleration of the milling tool tip point related to the WP during the milling process. A compact WP representation can be learned in an unsupervised manner, using an autoencoder, which feeds into the state space in an RL-based optimisation task. An additional direction of work is enhancing the RL-based optimisation methods by combining them with more advanced heuristics. Avoiding local optima is a common challenge while designing and applying meta-heuristics [6]. If a near-optimal WP position, determined by an RL-based method, is used as a starting point for an additional heuristic search, better absolute performance can be achieved while still maintaining the overall acceptable computation time.

## References

1. Bhinge, R., et al.: An intelligent machine monitoring system for energy prediction using a Gaussian Process regression. In: Lin, J. (ed.) 2014 IEEE International Conference on Big Data (Big Data 2014), pp. 978–986. IEEE, Piscataway (2014). <https://doi.org/10.1109/BigData.2014.7004331>
2. Brockman, G., et al.: OpenAI Gym (2016). <https://arxiv.org/pdf/1606.01540>
3. Campatelli, G., Scippa, A., Lorenzini, L., Sato, R.: Optimal workpiece orientation to reduce the energy consumption of a milling process. *Int. J. Precis. Eng. Manuf. Green Technol.* **2**(1), 5–13 (2015)
4. Cioffi, R., Travaglioni, M., Piscitelli, G., Petrillo, A., de Felice, F.: Artificial intelligence and machine learning applications in smart production: progress, trends, and directions. *Sustainability* **12**(2), 492 (2020). <https://doi.org/10.3390/su12020492>
5. Du Preez, A., Oosthuizen, G.A.: Machine learning in cutting processes as enabler for smart sustainable manufacturing. *Procedia Manuf.* **33**, 810–817 (2019). <https://doi.org/10.1016/j.promfg.2019.04.102>
6. Gandomi, A.H. (ed.): *Metaheuristic Applications in Structures and Infrastructures*, 1st edn. Elsevier Insights, Elsevier, London (2013)
7. Ke, G., et al.: LightGBM: A Highly Efficient Gradient Boosting Decision Tree (2017, undefined)
8. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor (2018). <http://arxiv.org/pdf/1801.01290v2>
9. Hayes, G.: mlrose: Machine Learning, Randomized Optimization and SSearch package for Python (2019). <https://github.com/gkhayes/mlrose>
10. Hill, A., et al.: Stable Baselines (2018)
11. Kothuru, A., Nooka, S.P., Liu, R.: Application of audible sound signals for tool wear monitoring using machine learning techniques in end milling. *Int. J. Adv. Manuf. Technol.* **95**, 3797–3808 (2017). <https://doi.org/10.1007/s00170-017-1460-1>
12. Merkel, D.: Docker: lightweight linux containers for consistent development and deployment. *Linux J.* **2014**(239), 2 (2014)
13. Meyes, R., et al.: Interdisciplinary data driven production process analysis for the internet of production. *Procedia Manufa.* **26**, 1065–1076 (2018). <https://doi.org/10.1016/j.promfg.2018.07.143>
14. Mitchell, M.: *An Introduction to Genetic Algorithms*. Complex Adaptive Systems. MIT, Cambridge and London (1996)
15. Nti, I.K., Adekoya, A.F., Weyori, B.A., Nyarko-Boateng, O.: Applications of artificial intelligence in engineering and manufacturing: a systematic review. *J. Intell. Manuf.* 1–21 (2021). <https://doi.org/10.1007/s10845-021-01771-6>
16. Pol, S., Baer, S., Turner, D., Samsonov, V., Meisen, T.: Global reward design for cooperative agents to achieve flexible production control under real-time constraints. In: *Proceedings of the 23rd International Conference on Enterprise Information Systems*. SCITEPRESS - Science and Technology Publications (2021). <https://doi.org/10.5220/0010455805150526>
17. Rangarajan, A., Dornfeld, D.: Efficient tool paths and part orientation for face milling. *CIRP Ann.* **53**(1), 73–76 (2004). [https://doi.org/10.1016/S0007-8506\(07\)60648-9](https://doi.org/10.1016/S0007-8506(07)60648-9)

18. Samsonov, V., Enslin, C., Köpken, H.G., Baer, S., Lütticke, D.: Using reinforcement learning for optimization of a workpiece clamping position in a machine tool. In: Proceedings of the 22nd International Conference on Enterprise Information Systems, pp. 506–514. SCITEPRESS - Science and Technology Publications (2020). <https://doi.org/10.5220/0009354105060514>
19. Serin, G., Sener, B., Ozbayoglu, A.M., Unver, H.O.: Review of tool condition monitoring in machining and opportunities for deep learning. *Int. J. Adv. Manuf. Technol.* **109**(3–4), 953–974 (2020)
20. van Laarhoven, P.J.M., Aarts, E.H.L.: Simulated annealing. In: van Laarhoven, P.J.M., Aarts, E.H.L. (eds.) *Simulated Annealing: Theory and Applications*, pp. 7–15. Springer, Dordrecht (1987). [https://doi.org/10.1007/978-94-015-7744-1\\_2](https://doi.org/10.1007/978-94-015-7744-1_2)
21. Wang, J., Yan, J., Li, C., Gao, R.X., Zhao, R.: Deep heterogeneous GRU model for predictive analytics in smart manufacturing: application to tool wear prediction. *Comput. Ind.* **111**, 1–14 (2019). <https://doi.org/10.1016/j.compind.2019.06.001>
22. Wu, D., Jennings, C., Terpenney, J., Gao, R.X., Kumara, S.: A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. *J. Manuf. Sci. Eng.* **139**(7) (2017). <https://doi.org/10.1115/1.4036350>
23. Xanthopoulos, A.S., Kiatipis, A., Koulouriotis, D.E., Stieger, S.: Reinforcement learning-based and parametric production-maintenance control policies for a deteriorating manufacturing system. *IEEE Access* **6**, 576–588 (2018). <https://doi.org/10.1109/ACCESS.2017.2771827>
24. Yuan, X., Li, L., Wang, Y., Yang, C., Gui, W.: Deep learning for quality prediction of nonlinear dynamic processes with variable attention-based long short-term memory network. *Can. J. Chem. Eng.* **98**(6), 1377–1389 (2020). <https://doi.org/10.1002/cjce.23665>