A. Duo, D. Reguera-Bakhache, U. Izaguirre and J. Aperribay, "Active Power Optimization of a Turning Process by Cutting Conditions Selection: A Q-Learning Approach," 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), 2022, pp. 1-6.

DOI: https://doi.org/10.1109/ETFA52439.2022.9921714.

Active Power Optimization of a Turning Process by Cutting Conditions Selection: A Q-Learning Approach

Aitor Duo, Daniel Reguera-Bakhache, Unai Izaguirre and Javier Aperribay
Faculty of Engineering, Electronics and Computing
Mondragon Unibertsitatea
Arrasate-Mondragón, Spain
{aduo,dreguera,uizaguirre,japerribay}@mondragon.edu

***Abstract*—In the context of Industry 4.0, the optimization of manufacturing processes is a challenge. Although in recent years many of the efforts have been in this direction, there is still improvement opportunities in these processes. The optimisation of the power consumed by the processes can be improved by means of the parameters of control. To date, this challenge has been addressed by Multi-Objective optimization techniques, however, Reinforcement Learning based approaches are raising with promising results in many industrial fields.**

In this paper, we propose a Reinforcement Learning (RL) based approach to optimize the active power consumption of a machining process by the cutting conditions selection. Through the application of Q-Learning algorithm, the agent self-learns the optimal solution through interacting with the environment. The approach was validated in three different scenarios demonstrating the feasibility of RL application to determine the cutting conditions values in order to optimize the active power consumption.

***Index Terms*—Reinforcement Learning, Q-Learning, Manufacturing, Cutting Process**

I. Introduction

To boost competitiveness and meet changing customer demands, the manufacturing sector is taking advantage of Information and Communication Technologies (ICT). Machining processes are no exception, as they move towards a smarter, connected network to become part of an industrial digital ecosystem.

Despite the advances made to date, there are still considerable opportunities for improvement because of the complexity of machining processes. In this context, extracting and analysing data from machining operations can provide valuable information to optimise the control of these complex systems.

In machining, the geometrical specifications of a component are produced by the relative movements of the tool and the workpiece. These processes can be presented as a set of input elements and output elements. The input parameters are physical components and quantitative parameters that define the process behaviour (i.e.: Cutting conditions, lubricant, workpiece, tool, etc.), and two groups define the output of the process, the industrial parameters and the scientific parameters. Industrial parameters are those which are desired to control or improve the actual industrial process i. e.: tool life-cost, workpiece surface quality, energy consumption, etc. While scientific parameters comprise intrinsic physical properties of the system i.e.: spindle power, vibrations, cutting forces, etc.

The development of unmanned processes capable of performing human operator tasks, can deliver significant improvements in productivity, cost and quality. Machining processes are a key factor in the manufacture of different parts, so it is required to develop intelligent systems to facilitate better decision making. Such processes are created in human-machine collaborative environments, that allow machines to gain autonomy from the operator experience through the use of appropriate Machine Learning (ML) tools.

Reinforcement Learning (RL) can be defined as a type of ML where the model is implemented as an agent that explores an unknown state space through "trial and error". The agent, with the aim to reach a goal state, determines which actions must take to move to different states where the only feedback is a scalar reward [1].

Sutton et al. divided in [2] the two main categories of RL methods: (i) Model-based methods where the core component depends on planning forward steps of the environment using a physical model and (ii) Model-free methods which mainly rely on the learning capability of the agent without planning. Model-free algorithms take into account feedback provided from the environment and never use calculated predictions of next state and next reward to change agent's behaviour, whereas model-based algorithms leverage the predictions of next state and reward with the aim of selecting the best actions. One of the most widespread example of model-free off-policy RL algorithm for temporal difference learning is Q-Learning.

Q-Learning algorithm is a straightforward way for the agent to self learn how to act efficiently in controlled Markovian environments [3]. This algorithm works in an iterative manner improving the appropriateness of the action a taken by the agent from a discrete action space A at the different evaluated states s from a defined state space S, where $a \in A$ and $s \in S$.

To date, RL is being increasingly used in optimization use cases. However, it is a technique rarely used in industrial environments. The agent should be trained on a simulated

controlled environment to gain knowledge from the process [4]. Thus, the lack of such simulators of complex industrial systems is the main bottleneck for RL application in manufacturing sectors.

It is however feasible to convert a machining process into a RL problem. In such machining processes, the states can be defined by scientific parameters mapped indirectly from industrial parameters to achieve operator-defined objectives. Therefore, it can be depicted as shown in Fig. 1. For a material-tool combination, the action is performed on the cutting conditions of the process. For each action, the process will return an observation measured by the scientific parameters and a reward, which indicates to the agent whether the action performed has been adequate or not.

The main objectives of the present work are: (i) the creation of an environment that simulates the behaviour of an industrial machining process and it's suitable for a RL approach. (ii) To tune the conditions that minimize the active power consumption of the machining process using RL techniques.

II. Related Work

RL can be broadly described as a learning paradigm where an agent is able to autonomously learn behaviour in a dynamic environment based on rewards and penalties [5]. In recent years, there has been growing interest in the application of RL to different research areas. For example in Robotics to self learn and adapt different routines [6], [7] or in Smart Buildings to improve energetic efficiency autonomously [8], [9].

The use of RL techniques in different industrial fields has been examined in a number of recent works. In [10], a Multi-Objective Reinforcement Learning based approach for prescriptive analysis was proposed and then validated in a real industrial scenario. The designed algorithm was able to generate more accurate insights to each operator by single optimal solution selection. In [11], authors presented a methodology for real time production scheduling in smart factories. Based on composite rewards the system self learn efficiently to achieve multiple objectives in industrial production processes. In [12], authors studied and then validated the application of the Q-Learning algorithm to increase the performance of parameter optimization in a dynamic job shop scheduling problem taking into account process information such us machine breakdowns and variations in shop floor conditions. Real-time scheduling problem in manufacturing scenarios was also addressed in [13] by the application of Q-Learning algorithm to optimize production performance.

The field of Manufacturing is complex and dynamic. Several authors have made a study of ML techniques and its application to solve different optimization problems. So far, however, there has been little discussion about the application of RL from a theoretical point of view with same objective in mind. In [14], a combination of ML and RL techniques was presented to address the complexity of multi-pass CNC turning by designing and implementing a method for multi-task parametric optimisation enhancing computation efficiency. In

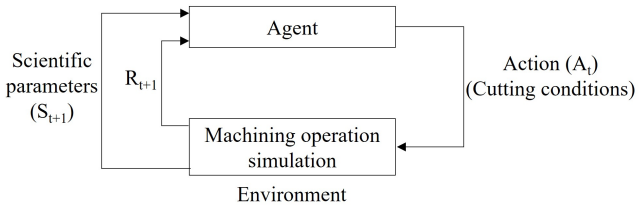


Fig. 1: Generalized Reinforcement Learning model in machining

[15], by the combination of nearest-neighbors algorithm and RL the authors presented a novel algorithm to suitable select process parameters to enhance the productivity in milling processes. Milling process optimization was also approached with RL techniques in [16]. In this study, authors demonstrated a successful application of RL to optimize workpiece clamping position and orientation in complex milling processes.

Therefore, in this work we present a RL based approach for a industrial process condition tuning. The main idea is to present an approach to minimize the active power consumption of the industrial process based on two actionable parameters.

III. Problem description

The combination of the working conditions of industrial processes generates an observation or combination of observations that needs to optimize. Some of the conditions may be fixed, or may have a higher cost of change (i. e. tool, material, coolant, the machine itself, etc.). Other conditions are variable: cutting speed (Vc), feed speed (f), depth of cut (ap), and these are those in which we can take particular actions to optimize one or more of the industrial parameters. Fig 2 shows an example of a turning process in which these parameters have a direct influence on energy consumption. In theory, machine energy consumption should be transferred completely to mechanics. In practice, taking into account the efficiency of the machine, there will be power losses because of frictional factors. Thus, with an adaptive automatic modification of cutting parameters the idea is to reduce the power required for material removal on these type of processes, as could be milling or drilling.

In the problem in progress, a first approach is made with a modifiable environment represented in Fig. 3 with possible applications in the machining area. The mesh can be rotated or the noise level increased or decreased to have different starting points and different target states.

For this work, two different scenarios are proposed. (i) The first one is the original mesh rotated by $\theta = 20°$ and $\tau = 0\%$ of Gaussian noise (Fig. 3 a). (ii) And the second one rotated by $\theta = 70°$ and $\tau = 8\%$ of Gaussian noise (Fig. 3 b). The exploring starting point depends on the rotation of the mesh. In the former there is more than one possible solution, while in the latter there are fewer possible solutions given the noise introduced.

For the current case study, the active power per chip flow generated W by the machining process must be minimised.

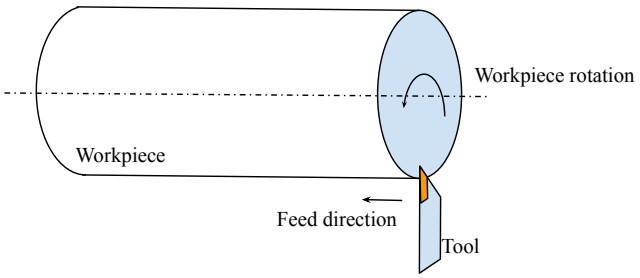
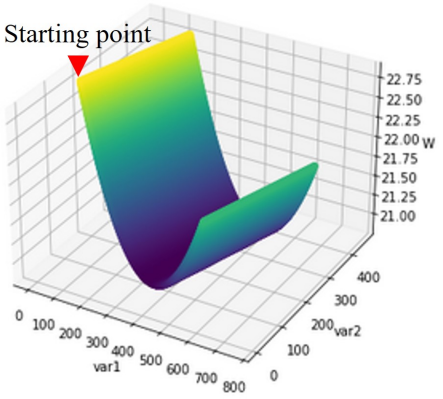
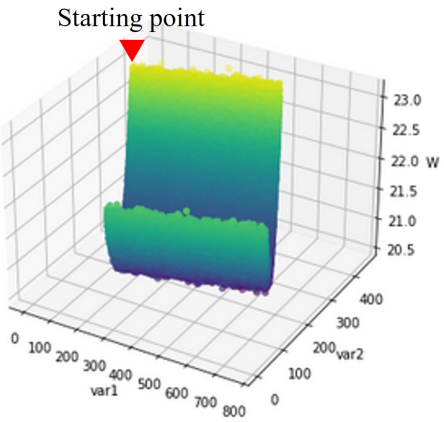


Fig. 2: Machining turning process.



(a) $\theta = 20°$, $\tau = 0\%$



(b) $\theta = 70°$, $\tau = 8\%$

Fig. 3: Environment representation by a mesh. The var1 and var2 are min-max scaled values of original axes, θ is the rotation degrees of the mesh and τ is the noise level introduced to the mesh.

However, in order to simplify the problem, the simulation environment returns a value (W) by combining the variables presented on Fig. 3. It should be made clear that this is a first approach to the automatic selection of conditions for machining processes. The data were obtained from a real process and although the ranges of the observations (W) are correct, the actionable conditions were scaled (min-max) for a better interpretation of the actions taken by the agent. Each point of the mesh represents an observation of the environment.

IV. Methodology

In this section we present the designed approach to train the agent in order to minimize the active power consumption, based on the Q-Learning algorithm application (Algorithm 1). Relied on Temporal Difference (TD) learning, the algorithm performs the following sequential process until the goal state is reached:

1) Initialize the Q-Table values arbitrarily, in the proposed scenario Q-Table is initialized to zeros.
2) Observe the current state s (where $s \in S$).
3) For the current state s and depending on $\mathcal{E}$, which defines the exploration/exploitation ratio, select a random action $a \in A$ (Exploration phase) or select the action $a \in A$ with max(Q-value) (Exploitation phase).
4) Once the action a is selected and taken, observe the reward R and the following state s_{t+1}.
5) Update the Q-Table values for the state s using Bellman equation (described in equation 2).
6) Set the state s to the new state s_{t+1}.

In the current study, the action to be taken by the agent during the training process is determined by the parameter $\mathcal{E}$. This parameter determines whether an action will be performed by the learned Q-values or whether an action is selected randomly (exploration/exploitation ratio). When $\mathcal{E} = 1$ the agent explores the environment taking random actions, whereas when $\mathcal{E} = 0$ the agent takes the action with maximum Q value for a specific state. The value of this parameter is decayed as a function of the equation 1. This means that in the initial stages of training, the value of $\mathcal{E}$ will approach $\mathcal{E}_{max}$ by exploring the entire state space S. This value will decrease exponentially until it reaches $\mathcal{E}_{min}$ by the decay rate (d) per each episode.

$$\mathcal{E} = \mathcal{E}_{min} + (\mathcal{E}_{max} - \mathcal{E}_{min}) \cdot e^{-d \cdot episode} \quad (1)$$

Once the action a to be taken on the environment has been determined, the environment will return the state s_{t+1}, the reward R and if the objective has been reached or not. With this information the $Q(s, a)$ is updated according to equation 2.

$$Q(s,a) = (1-\alpha) \cdot Q(s,a) + \alpha \cdot (R + \gamma \cdot Q(s_{t+1}, a_{t+1})) \quad (2)$$

Where α is the learning rate, which takes into account the immediate reward, ranging between [0,1], and γ is the discount factor, also between values [0,1]. The value γ is the one that computes the future reward.

The problem presented in this work is a discrete state space, the value functions are stored in a Q-Table, with the x- and y-axes being the states and actions. A visual representation of the Q-Table for a system with 143647 states and 4 actions is shown in table I.

Algorithm 1 Q-Learning Agent

```
Initialize Q-Table
for each episode in episodes do
    Initialize s
    while s is not goal state do
        for each t in episode do
            Select (exploration or exploitation) with ℰ
            if exploration then
                a ⇐ select random action a ∈ A
            else
                a ⇐ select action a with max(Q(s, a))
            end if
            R, s_{t+1} ⇐ Observe(a)
            Q(s, a) = Q(s, a) + α · [R + γ · maxQ(s_{t+1}, a_{t+1}) − Q(s, a)]
            s ⇐ s_{t+1}
        end for
    end while
end for
```

The action space is $A = \{a_1, a_2, a_3, a_4\}$, where each of the actions are, a_1 : increase var1, a_2 : decrease var1, a_3 : increase var2, a_4 : decrease var2. The state space is represented by the combination of the number of observation and the 4 actions that can be applied on each observation $S = \{1, 2, ..., 143647\}$.

Three tests have been performed to validate the feasibility of Q-Learning Algorithm application. The agent hyperparameters (γ, α, $\mathcal{E}$, d) of each of the tests and environment specifications defined by θ, mesh rotation angle, and τ, Gaussian noise can be shown on Table II.

The agent searches through the state space finding all possible solutions to reach the goal state. Fig. 4 shows the agent exploration process for the first, thirteenth and the one hundredth episodes in Test #2. It can be seen how the agent moves through the state space until it reaches the possible solutions. The black areas shown in Fig. 4 are those that the agent has not yet explored and the white areas are those that the agent has explored throughout the episodes. In episode 1 the agent is not able to find the combination of parameter modifications needed to reach one of the possible targets. As the agent explores the entire state space, the number of white areas grows until the agent explores an increasing number of possible solutions. At this point the agent starts to reduce the number of epochs needed per episode to reach the required solution in the most optimal way.

As the agent explores the entire state space, the Q-Table is updated, and the agent uses the learned Q-values by a ratio of $1 - \mathcal{E}$. The convergence of Q-Table during training phase was measured by the $return/epochs$ per episode.

V. Results and discussion

This paper has evaluated the possible application of reinforcement learning in machining environments. The results indicate that this type of application can contribute to the optimisation of the active power consumed by these processes through the autonomous selection of the cutting conditions.

TABLE I: Q-Table example for the state space (S) and action space (A) combination

S \ A	a_1	a_2	a_3	a_4
1	$Q_{[1,1]}$	$Q_{[1,2]}$	$Q_{[1,3]}$	$Q_{[1,4]}$
2	$Q_{[2,1]}$	$Q_{[2,2]}$	$Q_{[2,3]}$	$Q_{[2,4]}$
...	$Q_{[...,1]}$	$Q_{[...,2]}$	$Q_{[...,3]}$	$Q_{[...,4]}$
143647	$Q_{[143647,1]}$	$Q_{[143647,2]}$	$Q_{[143647,3]}$	$Q_{[143647,4]}$

TABLE II: Agent hyper-parameters, and environment specifications defined by mesh rotation and Gaussian noise

Test	γ	α	$\mathcal{E}$	d	θ	τ
#1	0.96	0.81	0.7	0	0	0
#2	0.96	0.81	1-0.01	0.01	0	0
#3	0.96	0.81	1-0.01	0.01	20	8

The convergence rates ($return/epochs$) of the training phase are shown in Fig. 5 for the three tests performed in this work. Test #1 and Test #2 were run with the same agent and the same environment and the $\mathcal{E}$ parameter was modified to compare the training performance regarding this $\mathcal{E}$ parameter. While for Test #3 a different environment was used and the hyperparameters of the best performing test between Test #1 and Test #2 were used.

Keeping the $\mathcal{E}$ fixed (Test #1), convergence rate and the stabilization of the curve is given earlier (around 13000 episodes). The Test #2 having a decaying $\mathcal{E}$, from 1 to 0.01 takes approximately 17000 episodes to converge. The last test (Test #3), having a lower number of possible objective states, and with a decaying $\mathcal{E}$, also from 1 to 0.01 (same hyperparameters as Test #2), takes around 40000 episodes to reach the optimum convergence rate.

The number of episodes needed to converge can be seen in the table III. Also the number of epochs needed to reach an optimum target state of the state space during the evaluation phase.

Although the Tests #1 and #2 were carried out on the same environment, in Fig. 5 it can be seen that the indicator $return/epochs$ is around 45 for Test #1 and 195 for Test #2. With both tests requiring a similar number of epochs during the evaluation phase, the return obtained in Test #2 is higher than in Test #1. So decaying the $\mathcal{E}$ parameter gets a higher return than keeping it fixed. With reference to the Test #3, in which the same hyperparameters are used as in the Test #2, a higher number of epochs are needed to reach the target state during evaluation phase.

The optimal trajectories obtained in each of the tests performed can be seen in Fig. 6. The orange dots in each of the figures represent the space of possible solutions for each of the environments tested in this work. The black rectangle in each of the graphics represents the observation space, and the white trajectory represents the changes on the variables to achieve the most optimal solution of the process in the evaluation phase.

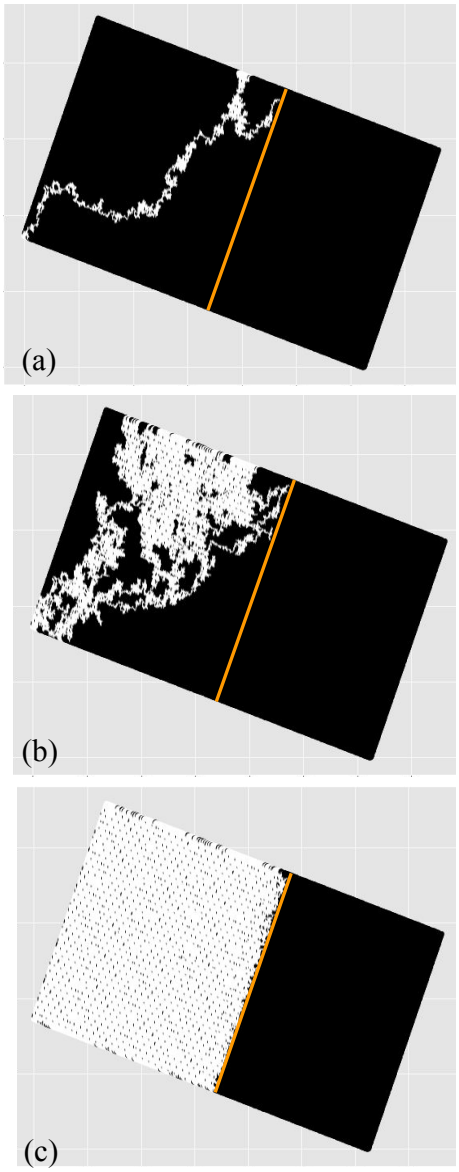


Fig. 4: Agent's exploration phase over different episodes for the **Test #2** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = [1 - 0.01]$, $d = 0.01$, $\theta = 20°$ and $\tau = 0\%$). a) Episode #1, b) Episode #30 c) Episode #100. White areas are those explored by the agent, whereas black areas are those that have not yet been explored.

TABLE III: Number of episodes to convergence (Ce) and number of epochs needed by the agent to reach the goal during evaluation phase.

Test	Ce	epochs
#1	13000	505
#2	17000	509
#3	40000	634

The trajectories of Fig. 6 a) and Fig. 6 b) are close to each other. The trajectory taken by Test 3 is different given the limited number of possible solutions. In this case the minimum

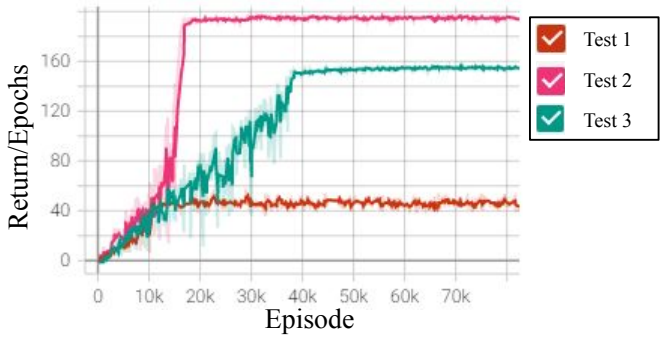


Fig. 5: Convergence rate during training phase of the agent in the different tests: i) **Test #1** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = 0.7$, $d = 0$, $\theta = 20°$ and $\tau = 0\%$), ii) **Test #2** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = [1 - 0.01]$, $d = 0.01$, $\theta = 20°$ and $\tau = 0\%$) and iii) **Test #3** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = [1 - 0.01]$, $d = 0.01$, $\theta = 70°$ and $\tau = 8\%$).

point of the mesh presented in Fig. 3 is at a centred point of the state space S. Therefore, a larger number of epochs is needed to arrive at the optimal solution.

Signal acquisition to control the cutting process has gained importance in recent decades. In addition to reducing manpower, it allows information to be obtained on the status of the cutting process. This means cost and time reduction.

VI. Conclusions and Future Lines

Reinforcement Learning (RL) is a learning paradigm that has the potential to face optimization problems for a wide range of applications. Agent's ability to self learn by interaction with the environment through the only feedback of the reward can lead greater efficiency and power consumption optimization in the machining processes. Hence, the development of this kind of approaches can contribute to face the energy crisis that most companies are experiencing nowadays.

In the present paper we have tested and validated the feasibility of applying RL techniques, more specifically Temporal Difference (TD) learning by Q-Learning algorithm, to select the most proper cutting conditions of a turning process. For this purpose, an environment with a large number of states has been created, indeed, a 143647 state space that has been tested in three different scenarios (Test #1, Test #2 and Test #3) demonstrating the validity of the proposed approach. Data gathered from the experimental phase proves that the agent self learns in the different scenarios, obtaining the cutting conditions values that optimizes the power consumption.

Further research could be undertaken to further enhance the proposed approach. One potential area of investigation is the application of Deep Reinforcement Learning (DRL) techniques to improve the convergence rate in such environments with a large number of states. Furthermore, the state space exploration process could be enhanced by the application of a multi-agent approach. This would deliver, a more effective method to explore the state space, and thus, the optimization of machining process power consumption.

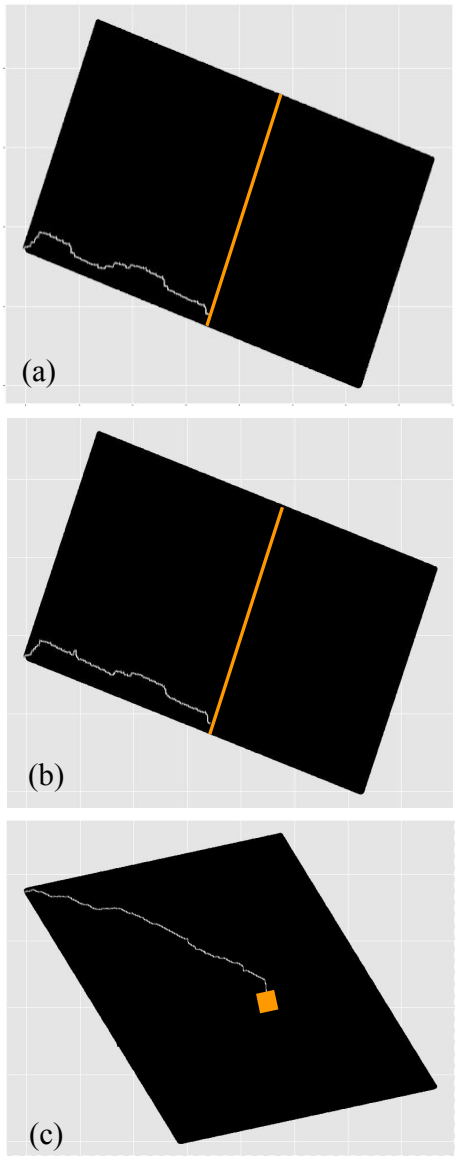


Fig. 6: Solution found by the Q-Learning Algorithm in the different conducted tests: a) **Test #1** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = 0.7$, $d = 0$, $\theta = 20°$ and $\tau = 0\%$), b) **Test #2** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = [1 - 0.01]$, $d = 0.01$, $\theta = 20°$ and $\tau = 0\%$) and c) **Test #3** ($\gamma = 0.96$, $\alpha = 0.81$, $\mathcal{E} = [1-0.01]$, $d = 0.01$, $\theta = 70°$ and $\tau = 8\%$).

Optimisation of production systems through digitisation is currently focused on reducing production time, costs, and increasing production quality. For a wider use of the proposed approach and to obtain more robust models it is necessary to carry out a larger number of tests and obtain a larger amount of data. It is also necessary to validate the proposed approach on a real machining process in order to determine the energy cost savings of implementing the proposed methodology.

ACKNOWLEDGMENT

This work has been developed by the Intelligent Systems for Industrial Systems research group (IT1676-22) and the High-performance Machining research group (IT-1315-19) of Mondragon Unibertsitatea supported by the Department of Education, Universities and Research of the Basque Country.

REFERENCES

[1] M. A. Wiering and M. Van Otterlo, "Reinforcement learning," *Adaptation, learning, and optimization*, vol. 12, no. 3, p. 729, 2012.
[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
[3] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.
[4] R. Nian, J. Liu, and B. Huang, "A review On reinforcement learning: Introduction and applications in industrial process control," *Computers and Chemical Engineering*, vol. 139, p. 106886, 2020. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2020.106886
[5] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.
[6] P. Kormushev, S. Calinon, and D. G. Caldwell, "Reinforcement learning in robotics: Applications and real-world challenges," *Robotics*, vol. 2, no. 3, pp. 122–148, 2013.
[7] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: Applications on robotics," *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 153–173, 2017.
[8] X. Ding, W. Du, and A. Cerpa, "Octopus: Deep reinforcement learning for holistic smart building control," in *Proceedings of the 6th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, 2019, pp. 326–335.
[9] A. Mathew, A. Roy, and J. Mathew, "Intelligent residential energy management system using deep reinforcement learning," *IEEE Systems Journal*, vol. 14, no. 4, pp. 5362–5372, 2020.
[10] K. Lepenioti, M. Pertselakis, A. Bousdekis, A. Louca, F. Lampathaki, D. Apostolou, G. Mentzas, and S. Anastasiou, "Machine learning for predictive and prescriptive analytics of operational data in smart manufacturing," in *International Conference on Advanced Information Systems Engineering*. Springer, 2020, pp. 5–16.
[11] T. Zhou, D. Tang, H. Zhu, and L. Wang, "Reinforcement learning with composite rewards for production scheduling in a smart factory," *IEEE Access*, vol. 9, 2021.
[12] J. Shahrabi, M. A. Adibi, and M. Mahootchi, "A reinforcement learning approach to parameter estimation in dynamic job shop scheduling," *Computers & Industrial Engineering*, vol. 110, pp. 75–82, 2017.
[13] Y.-R. Shiue, K.-C. Lee, and C.-T. Su, "Real-time scheduling for a smart factory using a reinforcement learning approach," *Computers & Industrial Engineering*, vol. 125, pp. 604–614, 2018.
[14] 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.
[15] J. Friedrich, J. Torzewski, and A. Verl, "Online learning of stability lobe diagrams in milling," *Procedia CIRP*, vol. 67, pp. 278–283, 2018.
[16] C. Enslin, V. Samsonov, H.-G. Köpken, S. Bär, and D. Lütticke, "Optimisation of a workpiece clamping position with reinforcement learning for complex milling applications," in *International Conference on Machine Learning, Optimization, and Data Science*. Springer, 2021, pp. 261–276.