CIRP Journal of Manufacturing Science and Technology 60 (2025) 248–259

Contents lists available at ScienceDirect

CIRP Journal of Manufacturing Science and Technology

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

Application of dynamic pricing for variant production using reinforcement learning

Florian Stamer [*], Matthias Henzi, Gisela Lanza

wbk Institute of Production Science, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

ARTICLE INFO

Keywords:
Variant production
Volatility
Dynamic pricing
Reinforcement learning
Capacity levelling

ABSTRACT

In the context of variant production, the increasing volatility and customer requirements challenge the profitability of manufacturers. A promising approach to mitigate these challenges could be a dynamic pricing. An intelligent design of a continuous delivery-time-price function allows customers to choose based on their preferences and demand may be shifted to level any peaks. This way, profit, service level, and capacity usage could be improved. This work develops a dynamic pricing model based on reinforcement learning applied to a use case of the automation industry. The results show that the dynamic pricing model performs better than current methods in practice.

1. Introduction

Diversified customer demands as well as a shortening of product life cycles lead to an increase in product variety [1]. This changed manufacturing systems towards so called variant production [1]. Currently, increasing challenges threaten the economic success of variant production [2,3]. On the one hand, increasing customer requirements demand for shorter delivery times, cheaper prices and higher quality [4]. On the other hand, disturbances in the supply chain lead to fluctuations of the production capacity usage and influence the capability of a production system to produce at the right time [3,5]. In the centre of the conflict between the capability to produce and customer requirements, the manufacturer has to decide about which order should be processed in what time and for what price [6]. The profitability of the manufacturer is significantly influenced by their pricing and delivery strategies. Setting prices too high, or failing to meet delivery expectations, can deter customers, resulting in fewer orders. Conversely, low prices may attract more orders, but this is only beneficial if it aligns with the manufacturer's capability to utilize their machinery and materials efficiently. The key is achieving a balance between demand and supply. An imbalance can lead to customer dissatisfaction, caused either by delays in delivery or by prices that do not reflect the value or urgency of the product.

A dynamic delivery time pricing model could be an effective solution for the manufacturer's challenges. This approach allows prices to be adjusted based on a customer's specific delivery time request. Such a dynamic pricing strategy has the potential to enhance overall profitability. However, to be successful, it must integrate real-time data from the production system, like machine availability and current order queues, as well as the customer's willingness to pay for a product at a particular delivery time. Addressing these factors is crucial for the model to effectively balance demand, supply, and price optimization. Manufacturers dealing with product variants face a complex production system, making behavior prediction challenging. Additionally, customer preferences are typically unknown and can only be inferred from their interactions. This work aims to develop and assess a dynamic pricing model tailored for variant production that addresses these challenges. The objective is to evaluate the model's performance against current industry practices, thereby demonstrating its practical viability. For a comprehensive assessment, various scenarios will be examined to test the robustness and adaptability of the dynamic pricing model in different market conditions.

2. State of the art

This section on the state of the art is divided into two distinct parts. Initially, we will delve into the methodological foundations, providing readers with a comprehensive understanding of the techniques employed later on. Subsequently, we will focus specifically on the current state of dynamic pricing in the context of variant production, offering an in-depth exploration of its application and relevance in this particular field.

* Correspondence to: wbk Institute of Production Science, Karlsruhe Institute of Technology (KIT), Kaiserstraße 12, Karlsruhe, Germany.
E-mail address: Germany.florian.stamer@kit.edu (F. Stamer).

https://doi.org/10.1016/j.cirpj.2025.05.004
Received 25 June 2023; Received in revised form 10 May 2025; Accepted 16 May 2025
Available online 26 May 2025

2.1. *Methodological foundation*

In this study, a thorough understanding of specific methods is essential. The dynamic pricing challenge can be approached from various mathematical perspectives. Here, we classify the dynamic pricing issue as a Markov Decision Problem (MDP). The key criterion for this classification is the principle that any change in state is influenced only by the current environmental state and the action taken within that context [7]. This means that all information from previous environmental states that is relevant for the future must be contained in the current environmental state. The environmental state then satisfies the Markov criterion or the Markov property [8]. MDP can be tackled with reinforcement learning (RL) [8]. In the conceptual model of RL, an agent can observe certain parameters of the environment, denoted as S, at a given time t for an interaction, also called an action. The agent then selects an action from the action Space A based on this observation [8]. In addition, the state transition probabilities *prob*, a reward function R and a discounting factor γ are important influencing factors for the action selection. The transition probabilities *prob* of the given environment are not known, as the system underlies a stochasticity. The goal of RL is to maximise the profit over time by selecting the right actions in specific given states.

This can be done using neural networks as they are general function approximators [9]. A neural network consists of neurons connected by edges. Each neuron has weighted inputs and a bias factor [10]. The sum of the weighted input values and the bias factor is translated into an output value using an activation function [10]. The neurons of a network can be arranged in layers, and multiple layers can be connected in succession to solve more complex approximations [10].

The training of neural networks can take a lot of data and time. Within this time, the actions of the RL agent could be harmful to the environment. So, practitioners want to make sure that the RL agent works. A method for the training and testing of RL agents can be the simulation. Simulation allows (statistical) hypotheses to be tested and scenarios to be explored without influencing the system under consideration [11].

2.2. *State of the art for dynamic pricing in variant production*

Dynamic pricing is a revenue management technique that proves particularly beneficial for manufacturers with limited capacities and diverse customers [12]. In the context of this work, the method is called dynamic delivery time pricing to emphasise the dependence of the price on the delivery time. However, in literature the term dynamic pricing is commonly used aggregating all sorts of approaches setting prices dynamically based on selected influencing factors. When applied, dynamic delivery time pricing can offer several benefits, such as:

- Customers can place more individualized orders by selecting their preferred price and delivery time in addition to their desired product.
- Manufacturers can improve key performance indicators such as capacity utilization by incentivizing demand shifts that are advantageous to them.

In literature, different approaches exist. First of all, approaches of the retail and service business must be considered which could be seen as the main domain for dynamic pricing. Dynamic pricing especially gained momentum since the introduction of online retail [13]. Accordingly, a vast amount of different work can be found. A relevant field of work is the inventory control, e.g. for perishable products [14,15]. Different pricing strategies based on manually developed mathematical heuristics are evaluated and the increase of profitability is shown [14, 15]. In general, it is a common approach in this domain to investigate the performance of dynamic prices under changing different variables of the pricing [13]. Besides retail, other domains have also adopted dynamic pricing already. In the domain of energy trading, Lu et al. (2018) develop an algorithm based on Q-Learning, which dynamically determines the end customer price in the energy sector [16]. The authors argue the choice of algorithm with the necessary ability to adapt to environmental changes. Another approach of the energy sector was developed by Bahrami et al. (2021) [17]. They develop an actor-critic RL model to determine energy prices based on the current demand profile of the customers. By doing so, the peak demand is shifted towards periods of low demand. However, it is important to point out the differences between classic domains like retail compared to a production context: An inventor is less complex than a production system. Also, the order of one product in retail typically does not have an impact on other products which can even change over time depending on machine availability. Recent work is examining the impact of dynamic pricing on the customer in terms of trust and fairness [18–20]. There are specific influencing factors like the price fluctuation range which are relevant for customers.

Considering the motivation, approaches from the domain of variant production are especially relevant and are presented in the following. Ata & Olsen develop a method to offer an individual delivery-time-price menu to each incoming customer [21]. A customer can then decide which delivery-time-price pair matches their preferences best and select this pair respectively. The preferences of a customer are not known by the manufacturer a priori and are modelled by an ‘S-shaped delay’ cost function. However, only two types of customers are considered. The manufacturer is obligated to meet the delivery time that was offered to the customer. The biggest shortcoming of this approach is that only one product is included in the model. Also, the approach lacks experimentability and scalability. Another concept is pursued by Garmdare et al. (2018) where the customer receives only one delivery-time-price pair [22]. The dynamically offered delivery-time-price pair influences the total demand and respectively the market share for one company. In general, the model does not seem to be applicable for the industrial practice, as the authors don’t consider information asymmetry and state that their algorithm is too computationally expensive for realistic models. Zhao et al. (2012) compare the method of a uniform price-lead-time offer with the method of a differentiated price-lead-time offer for manufacturers [23]. To assess which method would be better for a company, price-sensitive and delivery-time-sensitive customers are considered. The offer is determined analytically and takes the form of a price-lead-time menu for the differentiated case. As a result, the authors establish four theorems that specify which method is better in which situation. The model only applies to one product and simplifications in the representation of the production are made. Also, it is assumed that each service class has its own machine simplifying the problem but also increasing flexibility cost. Baykasoglu et al. (2020) on the other hand model a production system with capacities that are not fixed [24]. The authors focus on the capacity usage of a bottleneck resource and the whole production system for the dynamic price and delivery time decision. Thereby, only one delivery-time-price pair is quoted instead of a menu or function of both variables. Looking for relevant work outside the manufacturing industry, two interesting approaches exist for energy pricing: The approach of Stamer & Lanza (2023) picks up the use of actor-critic RL for a dynamic delivery time pricing but adapts it to variant production [25]. Nonetheless, the approach currently demonstrates its efficacy from a learning perspective, indicating that the agent can acquire meaningful behaviours. However, it has not been set up to showcase tangible real-world advantages. To achieve that, one must incorporate data from a specific use case, fine-tune the methodology and parameters in close alignment with the practical context, and then compare the method’s performance with that of existing real-world approaches. Subsequently, this approach is adopted, expanded upon, and implemented in a specific use case to evaluate its effectiveness.

3. Actor-critic reinforcement learning approach considering a production system

Manufacturers producing a variety of product variants face significant challenges due to increasing complexity and volatility in supply chains and customer demand. To balance production capacity with customer satisfaction, implementing a dynamic pricing strategy can be an effective solution. By adjusting prices and delivery times based on real-time production capacity and demand, manufacturers can influence customer ordering behavior, encouraging them to place orders that align better with production schedules and optimize capacity utilization.

This approach involves navigating complex trade-offs. Accepting more orders can boost revenue but risks overloading the production system, leading to delays, increased lead times, and potential penalties for late deliveries. Limiting orders maintains production within capacity limits, ensuring timely delivery but may result in lost revenue from unfulfilled demand. Offering lower prices for longer delivery times can encourage customers to accept delivery schedules that better fit available capacity, smoothing out demand but may not appeal to those needing urgent delivery. Prioritizing customer preferences, e.g. short term deliveries, enhances satisfaction and loyalty but may lead to suboptimal production schedules and increased costs. Optimizing production schedules improves efficiency and reduces costs but may require customers to accept less favorable terms, risking dissatisfaction. Additionally, the manufacturer often lacks precise knowledge of individual customer preferences and utility functions, making it challenging to tailor individual pricing and delivery time offers that will be accepted while optimizing production. Dynamic market conditions add further complexity, with fluctuations in supply chain availability, material costs, and competitor pricing strategies requiring the pricing approach to be adaptable to remain effective. The complete dilemma is also represented by Fig. 1.

To manage these trade-offs, an adaptive solution is necessary - one that can learn from interactions, handle complex and stochastic environments, and balance short-term and long-term objectives. In other words, the solution tries to cover the whole area of the distribution shown in the middle of Fig. 1 by utilizing the open capacity slots long term as well as short term shown on the right side of Fig. 1.

An actor-critic RL approach is proposed as a viable method for implementing dynamic pricing in this complex environment. This method allows the agent to output continuous pricing and delivery time offers, learn optimal policies through interaction with customers and the production system, balance exploration of new strategies with exploitation of known effective ones, and adapt continuously to shifts in customer behavior and production capacity (Fig. 2).

The first part is structured by the well-known and established general model of RL consisting of an environment, an agent, and the interaction

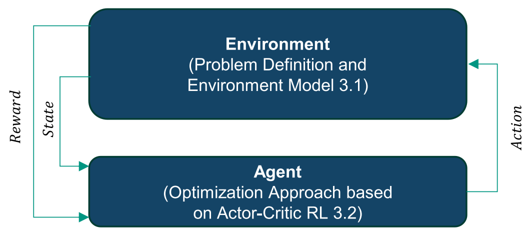

Fig. 2. General model of RL.

between these two [8].

In order to setup the environment model the core problem of dynamic pricing is elaborated in more detail. The agent model represents the optimization approach to tackle the problem. In line with this structure, the 'Problem Definition and Environment Model' (Section 3.1) and 'Agent Model' (Section 3.2) are introduced, with the interactions between the agent and the environment further outlined in the 'Agent Model' section. In 3.3, a KPI system is developed to identify, quantify and evaluate the results of the model in the application.

3.1. *Problem definition and environment model*

The optimization problem revolves around the manufacturer and the customer. In this problem we take the perspective of the manufacturer while assuming certain business rules and behaviour of the customer: The process begins when one of several customers has a demand for a product in a certain quantity. The manufacturer takes the request. Now, it is assumed that the manufacturer can offer a menu of prices and delivery times. This menu is defined for a product and quantity by a continuous delivery-time-price function $p_{i,l}(x_{i,l}, K_{i,l}, L_{i,l}, N_{i,l})$ which is elaborated in Section 3.2. By designing this function, the manufacturer tries to maximise the profit g which is the central optimization goal. This profit function takes into account penalties for orders produced either too early or too late, integrating earliness and lateness penalties into the manufacturer's overall costs.

$$g = \sum_{i,l} \left((p_{i,l} - c_l) \cdot m_{i,l} - c_{Tardiness,i,l} - c_{Earliness,i,l} \right) - c_{Cap} - c_{Op} \tag{1}$$

As a result, the manufacturer needs to consider the production schedule when designing a delivery-time-price function. The customer uses the delivery-time-price function $p_{i,l}$ in conjunction with his utility function u_i to determine a delivery time $x_{i,l}$ at a price $p_{i,l}$ by maximising his expected utility u_i. The utility function will be explained in Section 3.1.2 and the delivery-time-price function $p_{i,l}$ will be elaborated in Section 3.2.1. The index $i \in I$ represents the specific customer, while $l \in L$ denotes the specific order. If the customer's utility u_i is negative, he does

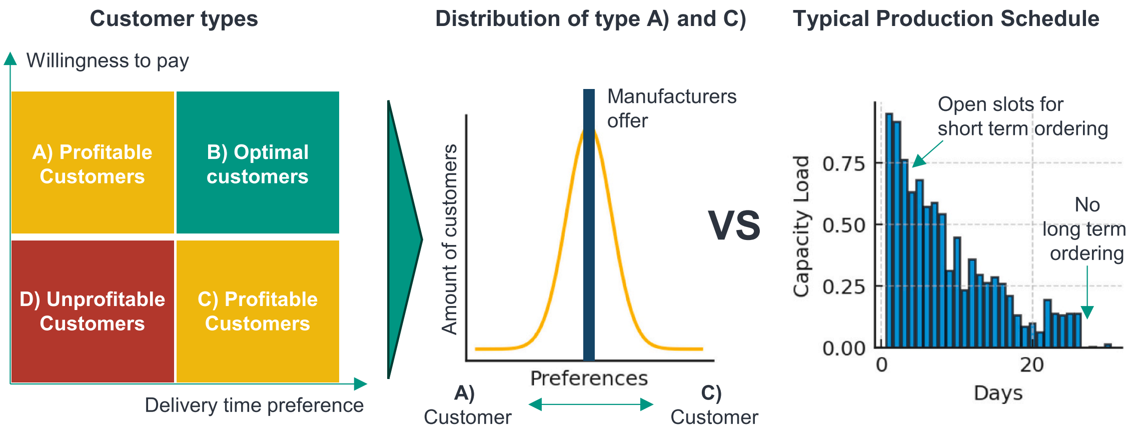

Fig. 1. Dilemma of static delivery-time-pricing.

not order. In case of a positive utility, the customer places an order. The manufacturers' goal must be to maximise the profit by adjusting the delivery-time-price function without losing the customer nor delaying any resulting agreed orders.

Given this problem definition, the models of the production planning and control as well as the customer will be examined in more detail in the following subsections.

3.1.1. Production planning and control

The production planning and control (PPC) is responsible for the sequencing of incoming orders. If there is a delivery-time-price combination with positive utility and the customer places the order, the PPC must check whether this order can be integrated into the existing production schedule without causing infeasible shifts or delays. Each product variant $v \in V$ requires multiple process steps $ps \in PS$. These process steps $ps \in PS$ can be processed on a subset of stations, as defined by a product-specific route sheet $R(v, ps)$. The route sheet is a mapping between all necessary process steps $ps \in PS$ (e.g. milling, then drilling, then assembling) of a product variant v and the stations $j \in \mathscr{M}$ capable of executing the process step. The route sheet can be expressed mathematically as:

$$R(v, ps) \subseteq \mathscr{M} \forall v \in V, ps \in PS \tag{2}$$

Stations can be manual workstation, a CNC machine, or a robot cell. Each station has a virtual waiting queue of scheduled orders with a specific processing step.

To ensure that the offered delivery times are actually manageable, the PPC needs to verify feasibility. This involves checking if the order (i, l) with the specific processing step fits into available capacities without violating constraints. Let $\mathscr{O}_t$ be the set of agreed orders at run time t, and $pt_{(i,l),v,ps,j}$the processing time of process step ps of product variant v of order (i, l) on machine j. $pt_{(i,l),v,ps,j}$is a random variable to account for processing uncertainties where values are drawn from a random distribution. Assigning process step ps of product variant v of order (i, l) to machine j is represented by a binary variable $y_{i,l,v,ps,j}$, where $y_{i,l,v,ps,j} = 1$ if assigned, and 0 otherwise. The PPC must then ensure overall:

$$\sum_{j \in \mathscr{M}} y_{i,l,v,ps,j} = h_{i,l} \quad \forall (i, l) \in \mathscr{O}_t, \quad v, ps, t \tag{3}$$

ensuring each processing step of each variant within an agreed order $h_{i,l}$ is assigned exactly once. Then, a capacity constraint needs to be considered:

$$\sum_{(i,l) \in \mathscr{O}_t, v, ps} pt_{(i,l),v,ps,j} \cdot y_{i,l,v,ps,j} \leq T_{cap,j} \quad \forall j \in \mathscr{M} \tag{4}$$

ensuring that total processing times do not exceed the available capacities $T_{cap,j}$.

These constraints help maintain a feasible schedule, which directly affects the manufacturer's global optimization goal (cf. formula 1). A feasible schedule reduces the risk of tardiness, while also minimizing any unnecessary earliness costs.

Many studies use FIFO rules or assign an incoming order to the machine with the shortest queue. However, such approaches are ill-suited for scenarios where negotiating delivery times can increase flexibility. Instead, a modified slack-based rule is applied. Let $T\mathscr{D}_{i,l}$ denote the committed due date for an order (i, l) and $T\mathscr{P}_{i,l}$ the planned completion time based on the current schedule. The slack $\mathscr{S}_{i,l}$ for order (i, l) can be defined as:

$$\mathscr{S}_{i,l} = T\mathscr{D}_{i,l} - T\mathscr{P}_{i,l} \tag{5}$$

To ensure that no new order insertion causes delays to previously planned orders, the PPC checks if adding a new order at its proposed delivery time reduces the slack of any existing order below zero. If inserting a new order into a station's queue would result in

$$\mathscr{S}_{i,l} < T_{safety} \tag{6}$$

for any already scheduled order, then the insertion at that position is deemed infeasible. T_{safety} is a parameter which covers the uncertainty coming from $pt_{(i,l),v,ps,j}$. By applying this slack-based rule, the PPC maintains delivery reliability, ensuring that adjusting delivery-time-price combinations remains a practical strategy. This careful control of the production schedule ultimately supports the manufacturer's profit optimization by minimizing tardiness, reducing unnecessary earliness, and maintaining the feasibility of promised delivery times. For the design of an optimal delivery-time-price function the manufacturer needs to predict the outcome of the production scheduling and the resulting capacity usage, ensuring that the proposed delivery times and prices remain both feasible and profitable.

Based on the introduced variables in this section and a new variant, processing step and station specific cost variable $ct_{v,ps,j}$ we can now define an equation to calculate the cost c_l:

$$c_l = \sum_{v,ps,j} ct_{v,ps,j} \cdot y_{i,l,v,ps,j} \quad \forall (l, \quad i) \tag{7}$$

The solution presented here is applicable to both single-stage (where the product requires only one processing step) and multi-stage (where the product undergoes multiple consecutive processing steps) models. The main difference in applying the solution to single-stage versus multi-stage models lies in the determination of the expected total processing time which gives as the planned completion time $T\mathscr{P}_{i,l}$. It is calculated by the sum of processing times $pt_{(i,l),v,ps,j}$. In single-stage models, this can be simply calculated using the expected processing time of the one relevant station. In contrast, in multi-stage models, due to mutual interactions and potentially dynamic material flows influenced by reallocation based on actual processing times, determining the planned completion time becomes a complex challenge. Various solution approaches could be developed, but they are beyond the scope of this work.

3.1.2. Customer

Customers $I = \{i_1, i_2, \ldots\}$ exhibit a range of demands that are specific to each product. These demands can be characterized by their distributions, which detail the frequency and quantity of product requests. As per established practices [11], the frequency of product requests is assumed to follow an exponential distribution, while the quantity is assumed to follow a Poisson distribution. Additionally, each customer is equipped with a utility function to model their behaviour, as discussed earlier. For the utility function in this work, the following assumptions are made:

- The overall utility of an agreement is determined by two external quantities: agreed delivery time $x_{i,l}$ for a specific product variant, price $p_{i,l}$ per unit for a specific product variant. Any payments from penalties are not considered – the customer bases the decision under the assumption that the order is fulfilled as agreed.
- The customer has an individually optimal delivery time for the order $b_{i,l}$ and a willingness to pay $d_{i,l}$ as given properties (parameters).
- The right side of the utility function exhibits a steeper slope than the left side, indicating that deviation from the optimum utility value towards delay has a more negative impact than deviation towards earliness. The slope in this context represents a trade-off between price and deviation from optimal delivery time which is a customer individual property controlled by parameter $a_{i,l}$.
- The utility function has exactly one maximum, which is always positive if the price is set to zero.
- The utility function is continuous and differentiable.

Based on these assumptions, the following utility function is developed:

$$u_i(p_{i,l}, x_{i,l}) = -a_{i,l} \cdot e^{(x_{i,l}-b_{i,l})} + a_{i,l}(x_{i,l} - b_{i,l}) + d_{i,l} - p_{i,l} \tag{8}$$

This function can also be seen exemplarily plotted in Fig. 3. It is worth noting that the utility function works with relative values here. The base is the base price P. All prices $p_{i,l}$ are implicitly relative changes to this base price. This way, decreasing marginal utility can be taken into account easily which is relevant as shown by Kahneman and Tversky (1979) originally [26]. In addition, these relative changes improve the ability to generalize during the learning of the agent later. The following table explains the parameters, the reasonable value range, and their influence on the function.

It's crucial to recognize that the customer model presented here doesn't mirror reality precisely. Customer behaviour often includes elements of randomness, akin to statistical noise. However, the model is a realistic-enough representation suitable for training and evaluating an agent in a simulated environment. The discrepancies between this model and real-world scenarios will be narrowed as the model is further trained during real-world application.

3.1.3. Environment model

The environment model is implemented using the problem descriptions in the previous subsections as an executable simulation model. The later introduced RL agent can interact and learn within this simulation model. The modelling of the environment in this study includes both the customers and the manufacturer, including the production system, as these elements are integral to the agent's pricing decisions. In the environment model the parameters - especially those listed in Table 1, Table 4, and Table 6 - can be adjusted in order to mirror different use cases. The parameters also limit the space of some variables, e.g. the possible delivery time.

3.2. Optimization approach based on actor-critic RL

As shown in the problem definition in Section 3.1 the manufacturer needs to find a way to offer a delivery-time-price menu maximising the profit. For this goal, the manufacturer needs to assume the utility function of the customer to include solutions in the menu which satisfy the customer. At the same time the capability of the production system needs to be considered so that the offered menu can be fulfilled. This can be classified as and MDP (cf. Section 2).

In this work, an actor-critic agent is chosen to solve the MDP due to its ability to combine the advantages of policy-based and value-based methods, as well as the capability to handle continuous state and action spaces [27]. The continuous state and action spaces allow the determination of a continuous delivery-time-price function based on continuous inputs which allows to implicitly and dynamically segregate customers by type. This approach has already been successfully implemented in the energy industry (cf. Section 2). In addition, it can handle continuous time input from the production system as a state.

To implement an actor-critic RL, the actor's policy and the critic's value function need to be approximated. One possibility to do so is the usage of neural networks as these are able to represent any function in theory [8,9]. Once the fundamentals of the actor-critic approach are understood, the key elements of the MDP, namely the reward function R,

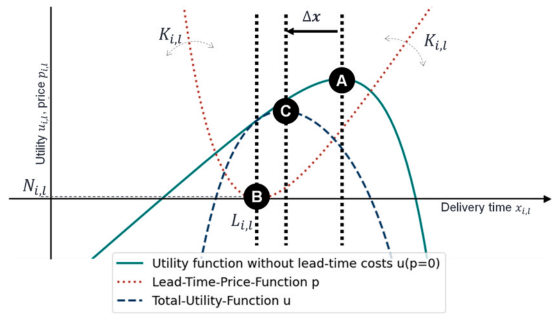

Fig. 3. Manufacturer price function (and shift of customer's utility).

Table 1
Definitions of parameters for profit function.

Symbol	Description	Value Range
c_l	Cost parameter for customer order l	[0, ∞)
c_{Cap}	Capital costs, e.g. possession of machines	[0, ∞)
c_{Op}	Operational costs, e.g. maintenance of machines	[0, ∞)
$i \in I$	Customer i as an element of all Customers I	N/A

Table 2
Definitions of variables for profit function.

Symbol	Description	Value Range
g	Profit of manufacturer	(-∞, ∞)
$p_{i,l}$	Price offered to customer i and order (with specific delivery time) l	[0, ∞)
$m_{i,l}$	Amount ordered from customer i within the order l	[0, ∞)
$c_{Tardiness,i,l}$	Tardiness cost for customer i and order l	[0, ∞)
$c_{Earliness,i,l}$	Earliness cost for customer i and order l	[0, ∞)
$l \in L$	Order l as an element of all orders L	N/A

Table 3
Definitions of parameters within the PPC.

Symbol	Description	Value Range
$v \in V$	Product variant as an element of the set of all product variants	N/A
$ps \in PS$	Process step as an element of the set of all process steps	N/A
$\mathscr{M}$	Set of machines available for processing orders.	N/A
$R(v, ps)$	Route sheet defining the capable machines for process step ps of variant v	N/A
$pt_{(i,l),v,ps,j}$	Processing time of processing step ps of variant v of order (i, l) on machine j.	[0, ∞)
$T_{cap,j}$	Available capacity of machine.	[0, ∞)
T_{safety}	Safety threshold for slack to account for uncertainties in processing time.	[0, ∞)
$ct_{v,ps,j}$	Cost of processing step ps of variant v on machine j.	[0, ∞)

Table 4
Definitions of variables within the PPC.

Symbol	Description	Value Range
$\mathscr{O}_t$	Set of agreed orders at runtime t. Each element represents a unique agreed order with an agreed delivery time.	N/A
$y_{i,l,v,ps,j}$	Binary assignment variable: 1 if order (i,l), product variant v, processing step ps is assigned to machine j, else 0.	{0,1}
$\mathscr{S}_{i,l}$	Slack for order (i, l)	(-∞, ∞)
$T\mathscr{D}_{i,l}$	Committed due date for order (i, l)	[0, ∞)
$T\mathscr{P}_{i,l}$	Planned completion time for order (i, l) based on the current schedule.	[0, ∞)
$h_{i,l}$	Binary variable that is 1, if customer i orders an order l (agreed order) (variable)	{0,1}

Table 5
Definition of utility function variables.

Symbol	Description	Value Range
u_i	Utility value of customer i	(-∞, ∞)
$x_{i,l}$	Agreed delivery time picked by the customer from the delivery-time-price function	[1, 30]
$p_{i,l}$	Resulting price from picked delivery time (cf.Sections 3.1 and 3.2.1 for more details)	[0, ∞)

Table 6
Definition of utility function parameters.

Symbol	Description	Value Range
$a_{i,l}$	Steepness factor	[0.1, 0.9]
$b_{i,l}$	Individually optimal delivery time for the specific customer	[1, 30]
e	Exponential constant	-
$d_{i,l}$	Willingness to pay	[0, 1.5]

the state space S and the action space A, will be presented in detail. Also, the composition of the neural network for the actor's policy and the critic's value function will be explained. Finally, the training of the actor's policy as well as the critic's value function will be explained in-depth.

3.2.1. *Action space and customer decision*

The action space is derived by looking at the decision that the agent has to take, in order to set dynamic prices for an individual customer. This decision results from a price function of the manufacturer, which is dependent on the delivery time. The mentioned price function for different delivery times that has been developed for the model presented in this paper, is shown in Eq. (9) and is characterised by the different variable values $K_{i,l}$, $L_{i,l}$ and $N_{i,l}$. An exemplary graph of the price function is presented in Fig. 3 (red dotted line). From a mathematically point of view, the variables of the price function work analogue to the parameters of the utility function. $K_{i,l}$ is analogue to $a_{i,l}$ and adjusts the steepness. This variable needs to be negative as deviation from optimal delivery time increase the price in contrast to the utility function where a deviation leads to a decrease in utility. $L_{i,l}$ is analogue to $b_{i,l}$ and defines the optimal delivery time form the perspective of the manufacturer. $N_{i,l}$ is analogue to $d_{i,l}$ and defines the minimum margin. As the variables are analogue to their parameter counterpart in the utility function the range is the same.

$$p_{i,l}\left(x_{i,l},\quad K_{i,l}, L_{i,l}, N_{i,l}\right) = K_{i,l}e^{(L_{i,l}-x_{i,l})} - K_{i,l}\left(L_{i,l} - x_{i,l}\right) - K_{i,l} + N_{i,l} \tag{9}$$

The function assumes that there is one optimal delivery time for the manufacturer. Orders that imply a shorter delivery time introduce overhead for replanning of existing orders and may cause delays. Orders with a longer delivery time may cause suboptimal capacity usage or imply storing the order. By setting the variables differently for every customer, its utility changes and the manufacturer in principle is able to move a customer's preferred due date (see point A, Fig. 3) in the direction of the manufacturer's optimal due date (see Point B, Fig. 3) based on the price. The customer's optimal delivery time is moved by Δx to Point C (cf. Fig. 3).

To implement the decision process of the customer after a price function was delivered by the actor, it is necessary to combine the price function of the actor (9) and the utility function of the customer (8) to get the total utility function (see blue dashed line in Fig. 3). This way, the new optimum of the customer's utility can be determined (Point C in Fig. 3). For this purpose, calculating the derivative of the total utility function, setting it to zero and checking also the second derivative for being negative is necessary. The customer's new optimum by calculating the first and the second derivative can be seen in the following equations:

$$x_{i,l,1/2}^{opt} = \ln\left(\frac{\left(a_{i,l} - K_{i,l}\right) \pm \sqrt{\left(a_{i,l} - K_{i,l}\right)^2 + 4\cdot a_{i,l}\cdot e^{-b_{i,l}}\cdot K_{i,l}\cdot e^{L_{i,l}}}}{2a_{i,l}\cdot e^{-b_{i,l}}}\right) \tag{10}$$

$$\frac{d^2u_i}{d^2x_{i,l}} = u_i'' = -a_{i,l}\cdot e^{\left(x_{i,l,1/2}^{opt}-b_{i,l}\right)} - K_{i,l}\cdot e^{\left(L_{i,l}-x_{i,l,1/2}^{opt}\right)} < 0 \tag{11}$$

Here, $x_{i,l,1/2}^{opt}$ represent the first and second possible solution of the derived and transformed function equated to zero to find the maximum of the function. If Eq. (11) holds true for the given solution, a maximum is found.

3.2.2. *Reward*

The design of the reward is critical for the performance of the agent. The RL agent gets a reward for each action taken. As an action is taken after each customer request, the reward is created after each order that the RL agent has been setting the price for. This reward is aligned with the profit maximisation goal of a manufacturer. Here, it is examined if the negotiation between the customer and the manufacturer has been successful in the sense that an order is arranged. In case the RL agent has set the variables of the price function in a way that it leads to a utility below zero for the customer, the customer will not order, and the agents receive a negative reward. This penalty could be interpreted as opportunity costs of a rejected order. It's crucial to incorporate penalties for unsuccessful negotiations. Without such penalties, there's a risk that the dominant strategy for agents might become setting excessively high prices. This tactic could artificially reduce the number of orders, thereby helping to avoid potential future penalties that arise from delays. Penalizing unsuccessful negotiations ensures a more balanced and realistic approach to pricing. When a successful order negotiation occurs, the difference between the planned production completion time and the actual due date is calculated. If the order is scheduled in such a way that it will be completed later than the agreed upon due date, a penalty is applied. This approach ensures adherence to the delivery schedule and encourages efficient production planning. This tardiness penalty is proportional to common contractual penalties. If no contractual penalties exist, the manufacturer needs to adjust the penalty in a way that the profit and delays are balanced from the manufacturer's perspective. If an order has been negotiated such that it will be expected to be early, meaning that the production end is planned before the due date, an earliness penalty is imposed. The earliness penalty is proportional to e.g., storage costs. Nevertheless, it's important to note that the penalty for earliness should be significantly lower than that for tardiness. This is because contractual penalties for late deliveries are typically more substantial and have greater financial implications. By penalising the deviation of the planned production end from the due date, the RL agent gets feedback on whether its decision has contributed to a punctual completion of the order. Finally, the negotiated price must be rewarded.

3.2.3. *States*

The state information critical for the agent to effectively determine the delivery-time-price function encompasses details about the customer requesting a product as well as the current status of the production system. These are collected from the environment model which is implemented as a simulation.

Customers are modelled with individual and heterogeneous preferences. The manufacturer does not know about a customer's preferences, but by interacting, the customer reveals them. By telling the agent which customer is ordering, it can learn the preferences implicitly. Therefore, the first partial state consists of the ordering customer that is one hot encoded (compare [28] for one hot encoding). The total number n of customers is known as it is a parameter of the system and can be specified beforehand. The dimension of this partial state therefore matches the number of customers n.

For the agent it is essential to receive information on the capacity usage of the production system and more specifically on the possible machines to produce the current order. Therefore, an estimator with this information is constructed. It consists of the shortest possible completion time of the considered order. Thereby, for each possible machine, the shortest possible completion time is calculated. Finally, the minimum of all shortest completion times is identified. The completion time as an input is normalised between 0 and 1 by assuming that there is a maximum realistic completion time. For this work, 30 days are assumed.

This value can be adjusted depending on the use case.

Having the shortest possible completion time as a partial state allows the agent to have an idea about the optimal delivery time of the manufacturer. The agent can combine this knowledge with the information on the current customer and thus set the price parameters in a way that the optimal due date of the customer moves towards the optimal due date of the manufacturer. As the agent can learn the customer preferences over time, it can try to decide on the price parameters so that the utility of the customer at the optimal point stays non-negative. If for example a highly delivery-time-sensitive customer is present and the optimal due date for the manufacturer is far in the future, then the agent has no choice but to drag the customer's utility into the negative area, which is the equivalent of rejecting a request. The state information together with the expected rewards allow the agent to weight the trade-off between earliness/tardiness and accepting/not accepting an order.

3.2.4. *Training of the actor and the critic*

Training the RL agent aims to improve the agent's policy to approximate the optimal policy for the given problem. It is based on the state of the art presented by Sutton and Barto (2018) [8] and Plaat (2020) [7]. Therefore, only approach-specific insights will be given. For the actor, an implementation of the Bellman equation and for the critic a classic mean squared error loss function are used. For the latter, the ADAM algorithm is used as a training algorithm in accordance with the state of the art [28].

To improve the learning of the agent, a common method is the learning rate decay. The learning rate α is reduced over time to enable big improvements in the beginning and small steps towards the optimum in the end. In this work, the learning rate decay is implemented using Eq. (12) based on Aggarwal (2018) [10].

$$\alpha_t = \quad \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) * e^{-\lambda * t_{le}} \tag{12}$$

3.3. *Key performance indicators for evaluation*

To quantitatively evaluate the performance of the dynamic pricing model, it is essential to use quantitative indicators. The core hypothesis posits that the dynamic pricing model will yield higher profits compared to current practices. As such, the manufacturer's profit, denoted as g (see Eq. (1) in Section 3.1), is established as the primary optimization objective of the algorithm. However, focusing solely on profit may overlook potential issues and decrease result comprehensibility. Hence, additional indicators are introduced to provide a comprehensive assessment in this context: The service level sg (Eq. (13)), the number of delays n_v (Eq. (14)), the number of lost orders n_{aa} (Eq. (16)), the sum of deviation times from delivery dates t_{la} (Eq. (17)). In addition to these commonly used key performance indicators, we suggest the sum of unused capacities n_m (Eq. (20)) as a proxy for saved capacities are identified as additional key performance indicators. This way, the agent has the incentive to interact in such a way that some machines are not required anymore. This can be beneficial if the cost of using a machine is greater than the value created. For example, if a product A is very valuable and uses 9 out of 10 machines while a product B having a low value is requested always on short notice using machines 8, 9 and 10 it could be beneficial to not use machine 10 anymore.

The key performance indicators are calculated using the following equations. The respective indices, parameters and variables are explained in Table 1 and Table 9.

$$sg = \frac{\sum_{i,l} h_{i,l} for \Delta x_{i,l} \leq 0}{\sum_{i,l} h_{i,l}} \tag{13}$$

$$n_v = \sum_{i,l} h_{i,l} for \Delta x_{i,l} > 0 \tag{14}$$

Table 7
Description of price function symbols.

Symbol	Description	Value Range
$K_{i,l}$	Steepness factor (variable)	[0.1, 0.9]
$L_{i,l}$	Optimal delivery time of manufacturer (variable)	[1, 30]
e	Exponential constant	-
$N_{i,l}$	Minimum margin (variable)	[0, 1.5]

Table 8
Description of learning parameters (hyperparameters).

Parameter	Description	Value Range
α_t	Learning rate at time t	[0, ∞)
$\alpha_{\min}$	Minimum learning rate	[0, ∞)
$\alpha_{\max}$	Maximum learning rate	[0, ∞)
λ	Exponential factor	[0, ∞)
t_{le}	Learning time	[0, ∞)

Table 9
Definitions of variables used for and as key performance indicators.

Symbol	Description	Value Range
sg	Servie level	[0, 1]
$h_{i,l}$	Binary variable that is 1, if customer i orders an order l (agreed order)	{0, 1}
$x_{i,l}$	(Negotiated, agreed) delivery time for customer i and order l	[0, ∞)
n_v	Number of delays	[0, ∞)
n_{aa}	Number of lost orders	[0, ∞)
$x_{real,i,l}$	Real delivery time time for customer i and order l	[0, ∞)
t_{la}	Sum of absolute deviations from the optimal lead time, e.g. sum of absolute earliness and absolute tardiness time	[0, ∞)
t_{early}	Sum of absolute earliness time over all customers and orders	[0, ∞)
t_{tardy}	Sum of absolute tardiness time over all customers and orders	[0, ∞)
n_m	Number of unused resources as a proxy for saved capacity	[0, ∞)
$y_{i,l,j}$	Binary assignment variable: 1 if order (i, l) is assigned to machine j, else 0.	{0, 1}
$\Delta x_{i,l}$	Deviation of lead time that was realized vs. lead time that has been committed to the customer	(-∞, ∞)

$$\Delta x_{i,l} = x_{real,i,l} - x_{i,l} \tag{15}$$

$$n_{aa} = \sum_{i,\quad l} (1 - h_{i,l}) \tag{16}$$

$$t_{la} = t_{early} + t_{tardy} \tag{17}$$

$$t_{early} = \sum_{i,l} |\Delta x_{i,l}| for \Delta x_{i,l} < 0 \tag{18}$$

$$t_{tardy} = \sum_{i,l} |\Delta x_{i,l}| for \Delta x_{i,l} > 0 \tag{19}$$

$$n_m = \quad \sum_{i,l} Min_j \left(1 - y_{i,l,j}\right) \tag{20}$$

4. Simulation experiments

The implementation of the dynamic pricing system including the environment is based on open source software developed mostly implemented by the authors themselves [29]. For the implementation of the RL agents the established library *tensorflow* is used as the core. A crucial aspect for the validity of every simulation is the handling of randomness and probability distributions. The probability distributions of the system, e.g. the ones mentioned in Section 3.2, have been implemented based on the mathematical state of the art using the inverse transformation method. The randomness is defined by a randomness source on which the stream of randomness is defined [11]. The source of randomness is also called the seed and typically has the form of a random number. Therefore, it is crucial to make sure to have a truly random seed as it defines the statistical reliability of the simulation. The seed for the randomness stream of our simulation is won from the free service *random.org*. With the developed software, a real-life application has been performed. First, a brief introduction of the use case is given, then a hyperparameter optimisation is presented to find good configuration parameters for the RL agent and, lastly, the central hypothesis of this work, improvement of profit, is investigated. Besides the improvement of profit, the already mentioned KPIs are examined.

4.1. Introduction of the use case

The developed model is being applied by taking the example of an industrial manufacturing company that is a world leader in automation technology. The use case data of this manufacturer is presented in table and will be explained in the following. With about 33.000 products in over 1.000 variants, the company can be classified as a variant manufacturer. Besides other products, the examined manufacturer produces pneumatic cylinders for handling solutions. Such a cylinder will be the basis for the real-life application. The considered variants of this product sum up to 87 different configurations with different processing times. The cylinder is sold and produced mainly in the USA, China, and Germany. Therefore, three different manufacturing locations in these countries are included in the environment model. Further, the production systems at the different locations are equipped with CNC machines. Here, a specific production step of milling is focused which is the bottleneck of the production. In total, 7 machine types are modelled. China and the USA each have 2 machine types, while the German site has 3 machine types Each machine has a specific amount of time that is required to produce a particular variant of a product. This time is in between 1.56 and 15.06 min per unit The modelling of the customers is based on interviews and data of the industrial use case partner. The analysis revealed the typical delivery time per customer and the price sensitivity. Values like the cost for delays have been determined through discussions with the industry partner.

4.2. Training and evaluation method

The evaluation of the agent is only reasonable after it was configured and trained sufficiently. The configuration is determined by a hyperparameter optimization (cf. Section 4.3). The training is carried out on the data described in Section 4.1. Now, one experiment run is structured as follows: After a settling phase filling the production system with orders the training of the agent begins. After the training is completed, the simulation is reset. Then again, a settling phase is carried out before the evaluation phase takes place. This structure is depicted in Fig. 4.

The goal of the evaluation phase is to determine the performance of the dynamic pricing model. A possibility is to compare it with another system in use. As the dynamic pricing model requires the handling of continuous state and action space, the meaningful comparison with existing algorithms is a huge challenge. Instead, in this work, the dynamic pricing model is compared to the current practice in the use case. This is called the status quo. Two possibilities of a status quo system are used reflecting observed practice in industry.

The first method is to provide a static delivery-time-price pair (static status quo). This is a *take it or leave* it offer for the customer without any negotiation space. In practice, this method can be used for commodity products which come close to mass products. Especially, if the manufacturer uses a webshop to distribute the products, prices for mid to low volume orders can be fixed based on simple price lists calculated in advance. In addition, the manufacturer sets a fixed delivery time which is used statically for a time frame like one year. The calculation is based on predicted volume and specific indicators like the lead time. In this case, the customer needs to judge if the offer is suitable. In terms of our model, this means inserting the offer into the utility function and checking for a positive utility value.

The second method is to accept basically any delivery time request of the customer (variable status quo) if it is within a given time frame. The time frame is currently assumed to be 1 to 30 days (cf. Section 3.2) but can be adapted to the use case. Prices are again set based on price lists calculated beforehand. This approach was observed for products with low total volume, smaller markets, and higher dynamics. In terms of our model, the optimal delivery time of the customer is used in combination with a price of the price list. Again, it is checked if the utility for the given price and optimal delivery time is positive.

The comparison between our dynamic pricing model and the status quo takes place in three scenarios to show the adaptability of the method when customer preferences change. To do so, the three scenarios simulate three relevant market developments. Scenario 1 reflects a situation with preference stable customers represented by the 8 types. Scenario 2 shows a drift towards shorter delivery times, imitating e.g., a market with shorter product lifecycles. Scenario 3 implements a drift towards price sensitivity, thereby representing e.g., a market with very mature products becoming commodity. To show significance of the result, the comparison is based on hypothesis tests. The hypothesis of higher profits is tested using a paired *t*-test at a 5 % significance level. Paired seeds are used for the simulation runs to allow the application of the *t*-test. 30 experiment repetitions are carried out per scenario and per status quo system to fulfil the central limit theorem. This totals in 180 (= 30 repetitions • 3 scenarios • 2 system comparisons) experiment runs. A simulation time frame of 2 (virtual) years is selected for each run.

4.3. Hyperparameter optimisation

It is beneficial to do a hyperparameter optimisation to adapt the RL agents to the use case specific environment. Hyperparameters are all parameters of the agent, in this case the neural networks, that can be controlled in the beginning to influence the learning process [10]. This includes the number of neurons, the number of hidden layers, the learning rate, and the learning rate decay. As the time of the hyperparameter optimisation grows exponentially with the number of possible parameters and their value range, the most relevant parameters must be selected, and their value limited to a reasonable space. The table provides an overview of empirically determined value ranges for the hyperparameters considered.

According to literature, good results and advantages in execution can be achieved with random search optimisation [28]. For this reason, a random search is used in this work as well. To gain more robustness, each configuration is run twice. The configurations are compared using the achieved reward. An overview of the search results is given by the

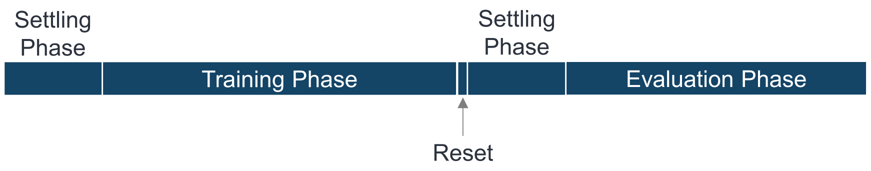

Fig. 4. Overview of experiment run structure.

following Fig. 5.

The best configuration, as indicated in table, is selected among all the model runs for the subsequent hypothesis test.

5. Results and discussion

As stated in Section 4.2 the simulation time of 2 years is run for each repetition. The variables of the *t*-test according to which the scenarios were evaluated are defined in table.

The test representing for all three scenarios is carried out in detail in the following subsection. At the end, a summary is given.

5.1. Scenario 1

The results for scenario 1 can be found in table.

The paired *t*-test shows that the presented dynamic pricing model is significantly better than the actual system in terms of profit as the null hypotheses of all three scenarios could be rejected. Considering the other KPIs, the following results (see table) were achieved:

5.2. Scenario 2

The results for scenario 2 can be found in the following tables.

The agent achieves statistically significant higher profit than the compared approach in scenario 2.

The agent achieves better values than the status quo approach regarding the service level and number of delays. However, this is not the case for lost orders and the deviation of optimal lead time. No machines have been unused in either approach.

5.3. Scenario 3

The results for scenario 3 can be found in the following tables.

In scenario 3, the agent outperforms the compared approach significantly.

The results of the agent indicate a better performance than the status quo approach regarding the service level and number of delays. Again, this is not the case for lost orders and the deviation of optimal lead time. No machines were completely idle except for a few runs in the flexible status quo comparison.

5.4. Summary of results

Summarising all three hypothesis tests, the dynamic pricing model is significantly better in all three scenarios (cf. Table 20). In the table the green check marks summarise the results from before that the agent performed better than the compared algorithm in all cases. However, considering the other KPIs this is not always true as can be seen in the previous tables.

The results of the study indicate that the developed model successfully incorporates dynamic pricing by considering the state of the production system and individual customers. One significant finding is that the dynamic pricing model achieved higher profits compared to the static status quo, which serves as a central performance indicator. However, it is important to approach the evaluation with a critical, nuanced perspective rather than a binary one. The results indicate that the agent was able to reduce delays and improve the service level. However, in scenario 2, the agent did not outperform the static status quo in terms of lost orders. In fact, compared to the variable status quo,

Table 10
Overview of the use case data.

Data description	Value
Product	Pneumatic cylinder
Variants of product	87
Current revenue	$\sim 8.7 Million€$
Reference machine	DMG Mori CTX
Calculated mean cycle time	$\sim$ 4 min
Machine invest (mean)	$207,000€$
Number of machine types (location USA)	2
Number of machine types (location China)	2
Number of machine types (location Germany)	3
Delay cost per day in percent of order price	25%
Delay cost (max) in percent of order price	200%
Earliness cost per day (inventory holding) in percent of order cost	0.041%

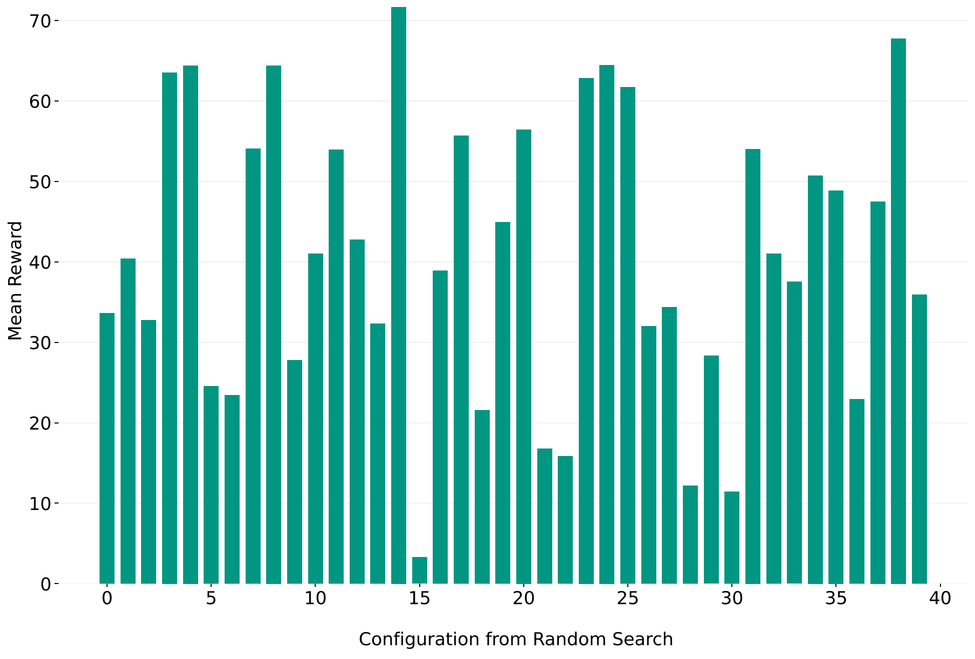

Fig. 5. Mean reward for different configurations from random search.

Table 11
Search space for different parameters of the model.

Parameter	Search Space
Number of hidden layers (actor)	{1, 2, 3}
Number of neurons per hidden layer (actor)	{1, 3, 5, …, 15}
Minimum learning rate (actor)	[0.1, 0.0001]
Maximum learning rate (actor)	[0.1, 0.0001]
λ for learning rate update (actor)	[0.001, 0.00001]
Number of hidden layers (critic)	{1, 2, 3}
Number of neurons per hidden layer (critic)	{1, 3, 5, …, 15}
Minimum learning rate (critic)	[0.1, 0.0001]
Maximum learning rate (critic)	[0.1, 0.0001]
λ for learning rate update (critic)	[0.001, 0.00001]

Table 12
Best configuration identified from random search.

Parameter	Value
Number of hidden layers (actor)	1
Number of neurons per hidden layer (actor)	9
Minimum learning rate (actor)	0.00008310
Maximum learning rate (actor)	0.00083101
λ for learning rate update (actor)	0.00018730
Number of hidden layers (critic)	2
Number of neurons per hidden layer (critic)	5
Minimum learning rate (critic)	0.00098259
Maximum learning rate (critic)	0.00982585
λ for learning rate update (critic)	0.00049037

Table 13
Variable definition for scenario evalutation.

Variable	Description
μ_g	Mean value of g
σ_g	Standard deviation of g
$\min_g$	Minimum of g
$\max_g$	Maximum of g
$\overline{\Delta G}$	Difference of the mean values of g
$\widehat{\sigma}_{\Delta G}$	Standard deviation difference of the mean values of g
n_f	Degree of freedom
T^*	calculated t-statistic
$t_{29,95\%}$	t-statistic value from the t-distribution with 29 degrees of freedom and 95 % confidence level
H_0	Null hypothesis

Table 14
Results of the hypothesis test for scenario 1.

	Agent	Static Status Quo	Agent	Variable Status Quo
μ_g	8, 009, 801 €	5, 359, 357 €	10, 069, 250 €	8, 545, 226 €
σ_g	460, 586 €	236, 056 €	474, 039 €	186, 196 €
$\min_g$	7, 206, 097 €	4, 858, 570 €	8, 440, 648 €	8, 016, 935 €
$\max_g$	8, 917, 600€	5, 725, 493 €	10, 734, 360 €	8, 817, 330 €
$\overline{\Delta G}$	2, 650, 444 €		1, 524, 029 €	
$\widehat{\sigma}_{\Delta G}$	442, 155 €		450, 371 €	
n_f	29		29	
T^*	32.83		18.53	
$t_{29,95\%}$	1.699		1.699	
H_0	*rejected*		*rejected*	
Confidence interval	[2, 485, 340; 2, 815, 548]		[1, 355, 857; 1, 692, 201]	

the agent consistently experienced higher losses in terms of orders. This outcome suggests that the dominant strategy was to focus on specific profitable customers and maximize their utility to generate higher profits.

Table 15
Overview of selected KPIs of scenario 1.

		Agent	Static	Agent	Variable
sg	μ	0.99	0.94	1.00	0.99
	σ	0.01	0.10	0.01	0.03
	min	0.94	0.63	0.97	0.86
	max	1.00	1.00	1.00	1.00
n_v	μ	351.77	1, 593.57	108.5	403.83
	σ	374.89	2, 619.45	181.03	839.21
	min	0	0	0	0
	max	1, 692	10, 355	649	3, 952
n_{aa}	μ	14, 392.47	16, 026.57	1, 379.87	0
	σ	1, 265.65	463.74	451.63	0
	min	12, 013	15, 136	825	0
	max	17, 318	16, 870	2, 954	0
t_{la}	μ	32, 043, 140	9, 061, 766	24, 114, 540	11, 168, 420
	σ	14, 770, 220	2, 021, 833	34, 007, 620	8, 427, 309
	min	16, 452, 100	7, 270, 063	8, 800, 213	8, 494, 451
	max	93, 210, 790	15, 925, 790	163, 826, 900	55, 505, 530
n_m	μ	0	0	0	0
	σ	0	0	0	0
	min	0	0	0	0
	max	0	0	0	0

Table 16
Results of the hypothesis test for scenario 2.

	Agent	Static Status Quo	Agent	Variable Status Quo
μ_g	7138,630 €	5354,359 €	8904,947 €	8527,393 €
σ_g	734,846 €	246,064 €	828,374 €	148,080 €
$\min_g$	5448,941 €	4859,762 €	6242,138 €	8233,124 €
$\max_g$	8173,410 €	5813,095 €	9999,909 €	8779,301 €
$\overline{\Delta G}$	1784,271 €		377,554 €	
$\widehat{\sigma}_{\Delta G}$	693,672 €		790,799 €	
n_f	30		30	
T^*	14.08		2.61	
$t_{29,95\%}$	1.7		1.7	
H_0	rejected		rejected	
Confidence interval	[1525,250; 2043,292]		[82,264; 672,843]	

Table 17
Overview of selected KPIs of scenario 2.

		Agent	Static	Agent	Variable
sg	μ	0.99	0.94	0.99	0.98
	σ	0.01	0.08	0.01	0.02
	min	0.93	0.69	0.98	0.93
	max	1	0.99	1	1
n_v	μ	167.3	1532.1	85.2	347.7
	σ	429.33	2051.2	120.37	501.52
	min	0	2	0	0
	max	2077	8523	431	2023
n_{aa}	μ	17,139	16,042.3	3814.17	0
	σ	2534.12	436.2	1654.37	0
	min	11,192	14,871	1295	0
	max	23,050	16,856	9130	0
t_{la}	μ	68,467,970	8408,414	78,150,650	12,222,360
	σ	41,180,890	660,757	37,901,610	9949,829
	min	22,498,440	7418,803	10,086,980	8080,062
	max	182,575,700	10,404,940	162,334,000	61,969,100
n_m	μ	0	0	0	0
	σ	0	0	0	0
	min	0	0	0	0
	max	0	0	0	0

Additionally, when using the RL agent, the sum of delivery deviations was consistently higher compared to the status quo. While the number of delayed orders was lower, indicating the agent's effective order management, the agent tended to produce with a higher deviation time. The agent demonstrated a cautious approach in accepting orders. In terms of capacity utilization, neither the agent nor the status quo

Table 18
Results of the hypothesis test for scenario 3.

	Agent	Static Status Quo	Agent	Variable Status Quo
μ_g	7229,551 €	4629,224 €	9515,279 €	8258,437 €
σ_g	589,362 €	237,467 €	869,50 €	1808,213 €
$\min_g$	5245,905 €	4195,372 €	6265,226 €	1293,336 €
$\max_g$	8024,972 €	5286,056 €	10,678,590 €	8877,499 €
$\overline{\Delta G}$	2600,327 €		1256,842 €	
$\hat{\sigma}_{\Delta G}$	650,753 €		1722,144 €	
n_f	30		30	
T^*	21.88		3.99	
$t_{29,95\%}$	1.7		1.7	
H_0	rejected		rejected	
Confidence interval	[2357,332; 2843,322]		[613,783; 1899,901]	

Table 19
Overview of selected KPIs of scenario 3.

		Agent	Static	Agent	Variable
sg	***μ***	0.99	0.99	0.99.43	0.96.70
	σ	0.02	0.02	0.0	0.15
	min	0.89	0.88	0.96	0.18
	max	1	1	1	1
n_v	***μ***	303,23	315,13	140,43	984,97
	σ	715	565,32	256,3	4.362
	min	0	0	0	0
	max	3.165	2.655	1.064	24.034
n_{aa}	***μ***	15,670,83	19,406,37	2397,2	0
	σ	1828,13	519	1855	0
	min	12,322	18,324	990	0
	max	20,599	20,502	9111	0
t_{la}	***μ***	61,230,880	40,975,310	58,143,230	24,957,580
	σ	53,372,570	1944,294	56,321,490	49,286,040
	min	19,896,810	38,365,060	8789,931	8843,152
	max	271,525,700	48,986,170	182,705,000	216,454,200
n_m	***μ***	0	0	0.17	0.03
	σ	0	0	0.38	0.18
	min	0	0	0	0
	max	0	0	1	1

spared any machines, suggesting that maintaining flexibility to meet customer demand was the prevailing strategy. Notably, a detailed analysis of the optimal configurations obtained from hyperparameter optimization showed that while some configurations succeeded in conserving machines, they did not yield profits as high as the most effective configuration.

6. Conclusion and future research

To address the challenges and increasing competition in variant manufacturing, aligning the production system's capabilities with customer demands is crucial. In this context, dynamic pricing strategies can be an effective alternative to solely relying on capital-intensive flexibility as a buffer against market disruptions.

The goal of this study was to develop a dynamic pricing model, utilizing reinforcement learning (RL) techniques, tailored to the unique needs of this sector. The model was designed to accommodate individual customer modelling, integrate a continuous delivery-time-price function, and handle multiple products and machines while considering the costs associated with delays and tardiness. Our investigation revealed that, although these aspects are critical, they have not been comprehensively addressed in existing literature. This gap underscores the innovative nature of our approach in optimizing dynamic pricing strategies for variant manufacturers.

Within the proposed dynamic pricing model, a Key Performance Indicator (KPI) system was established to evaluate the environment, the agent, and the overall dynamic pricing model, following the general RL model. The RL agent utilized an actor-critic approach. The developed dynamic pricing model was tested in a real-life application within the automation technology industry. To demonstrate its potential, the dynamic pricing model was compared against two status quo systems in three different scenarios. The status quo systems comprised a static price-delivery time pair and a customer-requested delivery time with a static price pair.

The results revealed statistically significant higher profits achieved by the dynamic pricing model in all three scenarios. However, a critical examination of other KPIs highlighted that the dynamic pricing model was not universally dominant. The increased profits were attained by focusing on a smaller customer base compared to the status quo while maximizing the service level for those customers. The initial claim that the dynamic pricing model would improve levelling of capacity usage was valid only within certain limitations, as no method could entirely save capacity. In this particular use case, it was hypothesized that preserving existing flexibility and prioritizing customer benefits were more effective in generating higher profits. However, this may not hold true for every application, and the balance between delay cost, lost order cost, and profit needs to be individually determined. This can have a huge impact on the results. In this study, the balancing was achieved through discussions with an industry partner, suggesting the consideration of production or company strategies in determining the appropriate balance.

The dynamic pricing model shows promise, yet there are areas for enhancement. A notable limitation is the model's requirement for a predetermined, fixed number of customers, which doesn't reflect the fluid nature of real-world market dynamics where new customers may appear over time. Continuously retraining the agent to accommodate new customers is a workaround, but it's computationally inefficient and not practical. To better align with real-world scenarios, future iterations of the model should be designed to manage a fluctuating customer base more efficiently. Additionally, while the model is intended for complex production systems, its effectiveness needs further validation in a broader range of complex use cases to fully establish its versatility and applicability. As part of this validation, a comparison with common approaches based on discretisation such as Random Forrest, Artificial Neural Networks or Support Vector Machines is necessary to show whether this approach can outperform them in practice.

A pivotal area for future exploration in the dynamic pricing model is its inherent multi-objective optimization problem. The diverse objectives – such as maximizing profit, minimizing delivery deviations and lost orders, and optimizing capacity usage – present a complex challenge as they do not integrate as straightforwardly as cost variables. In this study, the balance among these objectives was established through

Table 20
Overview of the RL agent's performance in comparison to two status quo systems in terms of profit.

	Scenario 1		Scenario 2		Scenario 3	
	Static	Variable	Static	Variable	Static	Variable
g	✓	✓	✓	✓	✓	✓

discussions with an industry partner, which had a significant impact on the outcomes. Future research should consider investigating different use cases to examine how varying the balance of these objectives might influence the results. This could provide a more comprehensive understanding of the model's applicability and effectiveness across a wider range of scenarios.

Lastly, considering a real-world integration a long list of additional points becomes relevant. Here, only three examples will be given. First, customers may get frustrated by varying prices. A quick solution can be to limit the price variation by adjusting the variable range of the price function and to frame them as discounts. Second, there may be scenarios with long-term orders which do not work well with the presented approach. Here, the pricing model should be combined with other models handling e.g. long-term orders differently. Third, customers could try to misuse the system, e.g. by ordering in such a way that delays occur and delay payments are given. It is necessary to examine under what conditions this behaviour may occur.

In conclusion, this research has provided valuable insights and laid a solid foundation for the necessary future investigations outlined in the preceding paragraphs, paving the way for a deeper understanding of dynamic pricing and its implications.

CRediT authorship contribution statement

Matthias Henzi: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization. **Florian Stamer:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Gisela Lanza:** Supervision, Project administration, Investigation, Funding acquisition, Conceptualization.

Declaration of Competing Interest

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

Acknowledgement

This publication is based on the research and development project 'ReKoNeT' which is/was funded by the German Federal Ministry of Education and Research (BMBF) within the 'Innovations for Tomorrow's Production, Services, and Work' Program and implemented by the Project Management Agency Karlsruhe (PTKA). The authors are responsible for the content of this publication.

References

[1] Kruse A, Butzer S, Drews T, Steinhilper R. A simulation-based framework for improving the ecological and economic transparency in multi-variant production. Procedia CIRP 2015;26:179–84. https://doi.org/10.1016/j.procir.2014.07.101.
[2] Dolgui A, Ivanov D. Ripple effect and supply chain disruption management: new trends and research directions. Int J Prod Res 2021;59(1):102–9. https://doi.org/10.1080/00207543.2021.1840148.
[3] Queiroz MM, Ivanov D, Dolgui A, Fosso Wamba S. Impacts of epidemic outbreaks on supply chains: mapping a research agenda amid the COVID-19 pandemic through a structured literature review. Ann Oper Res 2020:1–38. https://doi.org/10.1007/s10479-020-03685-7.
[4] Fogliatto FS, da Silveira GJ, Borenstein D. The mass customization decade: an updated review of the literature. Int J Prod Econ 2012;138(1):14–25. https://doi.org/10.1016/j.ijpe.2012.03.002.
[5] Ghadge A, Wurtmann H, Seuring S. Managing climate change risks in global supply chains: a review and research agenda. Int J Prod Res 2020;58(1):44–64. https://doi.org/10.1080/00207543.2019.1629670.
[6] Ivanov D, Tsipoulanidis A, Schönberger J. Global Supply Chain and Operations Management. Cham: Springer International Publishing,; 2019. p. 593. https://doi.org/10.1007/978-3-319-94313-8.
[7] Plaat A. Learning to Play. Cham: Springer International Publishing,; 2020. p. 335. https://doi.org/10.1007/978-3-030-59238-7.
[8] Sutton RS, Barto AG. Reinforcement learning: An introduction. Cambridge Mass: MIT Press,; 2018. p. 526 (xviii).
[9] Calin O. Deep Learning Architectures. Cham: Springer International Publishing,; 2020. p. 768. https://doi.org/10.1007/978-3-030-36721-3.
[10] Aggarwal CC. Neural Networks and Deep Learning. Cham: Springer International Publishing,; 2018. https://doi.org/10.1007/978-3-319-94463-0.
[11] Banks, J., 2013. Discrete-event system simulation, 5. ed., new internat. ed. ed. Pearson, Harlow, 559 pp.
[12] Zatta D. Revenue Management in Manufacturing. Cham: Springer International Publishing,; 2016. https://doi.org/10.1007/978-3-319-30240-9.
[13] Wang X, Liu S, Yang T. Dynamic pricing and inventory control of online retail of fresh agricultural products with forward purchase behavior. Econ Res-Èkon Istraživanja 2023;36(1). https://doi.org/10.1080/1331677X.2023.2180410.
[14] Chen X, Pang Z, Pan L. Coordinating inventory control and pricing strategies for perishable products. Oper Res 2014;62(2):284–300. https://doi.org/10.1287/opre.2014.1261.
[15] San-José LA, Sicilia J, García-Laguna J. Analysis of an EOQ inventory model with partial backordering and non-linear unit holding cost. Omega 2015;54:147–57. https://doi.org/10.1016/j.omega.2015.01.007.
[16] Lu R, Hong SH, Zhang X. A Dynamic pricing demand response algorithm for smart grid: reinforcement learning approach. Appl Energy 2018;220:220–30. https://doi.org/10.1016/j.apenergy.2018.03.072.
[17] Bahrami S, Chen YC, Wong VWS. Deep reinforcement learning for demand response in distribution networks. IEEE Trans Smart Grid 2021;12(2):1496–506. https://doi.org/10.1109/TSG.2020.3037066.
[18] David ME, Bearden WO, Haws KL. Priced just for me: the role of interpersonal attachment style on consumer responses to customized pricing. J Consum Behav 2017;16(6). https://doi.org/10.1002/cb.1651.
[19] Krämer A, Friesen M, Shelton T. Are airline passengers ready for personalized dynamic pricing? a study of German consumers. J Revenue Pricing Manag 2018;17(2):115–20. https://doi.org/10.1057/s41272-017-0122-0.
[20] Lastner MM, Fennell P, Folse JAG, Rice DH, Porter M. I guess that is fair: How the efforts of other customers influence buyer price fairness perceptions. Psychol Mark 2019;36(7):700–15. https://doi.org/10.1002/mar.21206.
[21] Ata B, Olsen TL. Congestion-based leadtime quotation and pricing for revenue maximization with heterogeneous customers. Queueing Syst 2013;73(1):35–78. https://doi.org/10.1007/s11134-012-9288-8.
[22] Garmdare HS, Lotfi MM, Honarvar M. Integrated model for pricing, delivery time setting, and scheduling in make-to-order environments. J Ind Eng Int 2018;14(1):55–64. https://doi.org/10.1007/s40092-017-0205-y.
[23] Zhao X, Stecke KE, Prasad A. Lead time and price quotation mode selection: uniform or differentiated? Prod Oper Manag 2012;21(1):177–93. https://doi.org/10.1111/j.1937-5956.2011.01248.x.
[24] Baykasoğlu A, Subulan K, Güçdemir H, Dudaklı N, Eren Akyol D. Revenue management for make-to-order manufacturing systems with a real-life application. Eng Econ 2020;65(1):27–65. https://doi.org/10.1080/0013791X.2019.1571145.
[25] Stamer F, Lanza G. Dynamic pricing of product and delivery time in multi-variant production using an actor critic reinforcement learning. CIRP Ann 2023;72(1):405–8. https://doi.org/10.1016/j.cirp.2023.04.019.
[26] Kahneman D, Tversky A. Prospect theory: An analysis of decision under risk. Économ: J Econom Soc, Intern Soc Adv Econ Theory Relat Stat Math 1979;47(2):263–91.
[27] Scharf F, Helfenstein F, Jäger J. Actor vs Critic: Learning the Policy or Learning the Value. In: Belousov B, Abdulsamad H, Klink P, Parisi S, Peters J, editors. Reinforcement Learning Algorithms: Analysis and Applications. Cham: Springer International Publishing; 2021. p. 123–34.
[28] Agrawal T. Hyperparameter Optimization in Machine Learning. Berkeley, CA: Apress,; 2021. p. 177. https://doi.org/10.1007/978-1-4842-6579-6.
[29] Stamer, F., 2022. Dynamic Pricing in Production Networks: An Implementation with Reinforcement Learning in Typescript. doi:10.5281/zenodo.5791970.