Computers in Industry 116 (2020) 103190

Contents lists available at ScienceDirect

Computers in Industry

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

A Decision Support System for rapid ramp-up of industry 4.0 enabled production systems

Stefanos Doltsinis[a], Pedro Ferreira[b,*], Mohammed M. Mabkhot[b], Niels Lohse[b]

[a] Information and Technologies Institute, Centre for Research and Technology Hellas, 57001, Thessaloniki, Greece
[b] Intelligent Automation Centre, The Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough, UK

ARTICLE INFO

Article history:
Received 28 August 2019
Received in revised form
20 December 2019
Accepted 14 January 2020

Keywords:
Cyber-Physical System
Decision Support System
Industry 4.0
Intelligent manufacturing
Production ramp-up

ABSTRACT

Production ramp-up is a key phase during the introduction or changeover of a production system. Process calibration and tuning are inevitably required to make such a system fully operational and let it reach its maximum production yield. A complex decision-making process takes place in order to optimally tune the system and requires a long time for testing and experimenting that will determine the system behaviour. This work considers the sequential nature of ramp-up and proposes a Cyber-Physical Systems approach based on data capturing, learning mechanisms and knowledge extraction, leading to an Industry 4.0 compliant Decision Support System (DSS) for human operators. The proposed system is implemented as an online DSS and also supports offline learning using previously gathered knowledge. A number of experiments have been carried out on a micro scale assembly station, validating the expected benefits of the proposed DSS. Results show a reduction of over 40% in the number of ramp-up steps required when using the DSS.

1. Introduction

Nowadays, production systems (PSs) are subject to rapid technological changes, which led to very short product lifecycles and high variety of products, to meet the growing demand of product customization (Kalir and Rozen, 2018). In addition to that, PSs becomes more modular and able to change from one configuration to another (Colledani et al., 2018). The rapid introduction of new products, mostly complex, and changeover cause a prolonged and more frequent production ramp-up (Letmathe and Rößler, 2019). Ramp-up phase is very important for PSs because having high outputs with good quality and fast cycle time are the key factors for increasing profits, where at the beginning of product life-cycle product prices are the heights and competition is scarce (Kalir and Rozen, 2018).

Production ramp-up is a decision-making process where human experts decide on the best actions to fine-tune the process. It is a highly complex parameter tuning process with many interrelated factors leading to a well-defined goal. Studies on ramp-up report long lead times of up to a few months from the initial system built to full production (Terwiesch and Bohn, 2001). Since the product, PS, and the supply chain are new, uncertainty is very high, making the ramp-up process very difficult to manage (Hansen and Grunow, 2015). Also, ramp-up process is unstable, expressed in the unpredictability of the system at this stage, which makes planning proactive steps to avoid some problems ineffective for many ramp-up problems (Schuh et al., 2015).

In practice, during ramp-up, a PS is adjusted and changed until it becomes sufficiently stable (disturbances reduced to a minimum) and its production output reaches the desired level (Carrillo and Franza, 2006; Doltsinis et al., 2014; Vits et al., 2006). The current level of stability and performance is not clearly quantified during the process and different studies adopting range of metrics, including cost, quality, quantity and time related indicators (Surbier et al., 2014; Glock and Grosse, 2015). The required time is highly dependent on the system complexity and the ability of system integrators to make good decisions according to their experience. Ramp-up is essentially a human-machine fine-tuning process that varies in every instance according to the personnel's experience and the system's complexity, making the result highly unpredictable.

A substantial body of research highlights the important role of the operators and their gained knowledge during ramp-up which is usually not documented and not shared across an enterprise (Terwiesch and Bohn, 2001; Hansen and Grunow, 2015; Schmitt et al., 2018). Gaining knowledge through training and experimenta-

* Corresponding author.
E-mail addresses: doltsinis@iti.gr (S. Doltsinis), P.Ferreira@lboro.ac.uk (P. Ferreira), m.mabkhot@lboro.ac.uk (M.M. Mabkhot), n.lohse@lboro.ac.uk (N. Lohse).

https://doi.org/10.1016/j.compind.2020.103190
0166-3615/© 2020 Published by Elsevier B.V.

tion with the production system is time-consuming and expensive, considering the stochastic nature of the process and the individuality of every manufacturing system. Therefore, knowledge gathering and sharing have an essential role in current approaches in the literature focusing on supporting the operators with the appropriate information (Heine et al., 2016; Doltsinis et al., 2017; Zülch, 2006).

Decision Support Systems (DSS) are used to help decision makers cope with inherently complex problems or situations. The human perception and intelligence can often reach its limits when confronted with complex systems generating a large amount of data with many interdependencies. Computers, on the other hand, can process large amounts of data fast, reliably, and accurately. Recently, Industry 4.0 is introducing significant changes to systems, particularly on the availability and processing of data in CPS (Liu et al., 2017). This alone enables significant progress in knowledge representation models and learning algorithms enhancing the DSS capabilities and their use in commercial areas (Glock and Grosse, 2015).

To develop an efficient DSS for ramp-up, a lot of attention needs to be paid on the right learning algorithms and knowledge representation. The limited similarity between production systems poses a significant challenge for finding a generally applicable model for extracting rules and recurring patterns. Also, human operators are a significant element of ramp-up and more human-friendly designs can support better intelligent manufacturing systems (Pacaux-Lemoine et al., 2017). Previous works have supported this approach showing that machine learning (ML) approaches can achieve promising results when designed to enable human-machine cooperation. Particularly, Reinforcements Learning (RL) methods show a potential capability to tackle the specificity of ramp-up problems (Doltsinis et al., 2014).

In the literature, ramp-up problems are the focus of considerable works, but few of them developed a DSS or RL approaches to handle the ramp-up process. However, as it will be discussed in section II, there is no work exploits the aforementioned potentials offered by the CPS an RL to manage the ramp-up process. This work effectively combines different RL learning approaches and utilizes the extracted knowledge, supporting decision-making at the ramp-up process. A DSS for ramp-up is developed that captures knowledge and shares it across different cases based on clear performance criteria. The DSS manages the ramp-up process on the fly, creating an intelligent CPS and an effective Human Machine Interface (HMI). Several scenarios are tested with several human operators ramping-up an I 4.0 enabled production station. The next section reports related studies in ramp-up and positions the paper to these studies. Section IIIII presents the main contribution of this study, a DSS for ramp-up, followed by the experimental procedure and results in sectionIV. Section V discusses the result and section VI draws the conclusion and possible research directions.

2. Related works

Ramp-up is defined as the phase that starts after the development of a product, during which the production output is scaled up to reach a predefined maximum throughput. To achieve that, a detailed fine-tuning takes place and the system is adjusted accordingly. The process of fine-tuning a production system exhibits some characteristics that make it difficult and time-consuming. Previous studies highlight the stochastic and sequential nature of the process with little knowledge of system's dynamics and its uncertain behaviour (Haller et al., 2003; Ball et al., 2011). According to Surbier et al. (2014) ramp-up phase is characterized by a low level of initial knowledge, throughput, and production capacity; high in cycle time, demand, and disturbances; and lack of planning reliability.

2.1. Decision models in ramp-up

In industry and academia, a variety of models and strategies have been developed to support ramp-up, with some of the characteristics being common across those cases. In the literature, three interesting review articles investigate exiting works and analysed various aspects of production ramp-up (Colledani et al., 2018; Surbier et al., 2014; Glock and Grosse, 2015). Surbier et al. (2014) present an overview of the production ramp-up, classifying existing works based on industrial application. Colledani et al. (2018) address the quality improvement problem at ramp-up phase. They discuss two common strategies to deal with the problem; anticipating problems and quality performance improvement. In the first strategy, the ramp-up problems and its causes are anticipated at the product design phase. In contrast, the second strategy focuses on improving quality performance during the system ramp-up.

Glock and Grosse (2015) summarize existing decision support models in the ramp-up phase. In general, the focus of decision supports models in ramp-up is on:

- ***Capacity investment/expansion:*** Models aim at determining the capacity of equipment, production lines, or workforce at the ramp-up stages. Knowledge about the capacity of available resources is essential for planning the ramp-up process, where some disturbances can be avoided in advance (Hansen and Grunow, 2015; Couretas et al., 2001).
- ***Product design management:*** Models in this category identify and fulfil product functional requirements and plan a set of verification activities at the design stage. It supports stabilizing the ramp-up process by taking a proactive modification in product design (Kukulies and Schmitt, 2018).
- ***Inventory management/lot sizing/supplier selection:*** These models aid in the selection of demand rate (Manna and Chaudhuri, 2006), inventory level (Glock et al., 2012), lot size (Manna and Chiang, 2010; Pal et al., 2014), or suppliers (Meisel and Glock, 2018), which contribute to optimize the ramp-up phase.
- ***Worker assignment:*** Models here focus on allocating workforce to tasks/workstation, trying to exploit the experience and expertise of workforce to improve the ramp-up phase (Terwiesch and Bohn, 2001; Glock et al., 2012).
- ***Performance measurements/Monitoring/diagnostics***: These models measure the performance of the ramp-up process (Doltsinis et al., 2013), seeking to diagnose ramp-up problems (Ceglarek et al., 1994) or eliciting corrective actions.
- ***Workflow management***: Models in this category support controlling some parameters in the ramp-up process such as feed rate (Nembhard and Birge, 1998), production output and production yield (Terwiesch and Bohn, 2001), or other process specification (Terwiesch and Xu, 2004).

Although a few decision-support models have been developed for various aspects of the ramp-up process, an integrated decision support system targeting the fine-tuning of the whole process to reduce ramp-up time has not yet been reported in the literature.

2.2. Ramp-up in industry 4.0

Industry 4.0 provides great opportunities for ramp-up and, at the same time, creates new challenges. CPS promises the availability and exchange of data between all “things” (i.e. machines, workforce, products, building) that are components of PSs. The main idea of CPS is to access and analyse all data on the fly to integrate the flow of material and information (Dombrowski et al., 2018). As a result, information about the previous and current status of every “thing” can be interpreted and processed at any time by another human or machine. Such potential could facilitates mastering the ramp-up phase. In the other hand, ramp-up task repetitiveness (Schuh et al., 2015) and system resources heterogeneity are major challenges (Uhlemann et al., 2017). Data will be collected from heterogeneous system components, including human, which makes online processing of such a data unmanageable (Dalenogare et al., 2018).

Toward the realization of CPS, four review papers highlight number of potential research priorities in ramp-up (Colledani et al., 2018; Surbier et al., 2014; Glock and Grosse, 2015; Schmitt et al., 2018). Schmitt et al. (2018) point out that moving toward CPS induces revolutionary changes in the ramp-up managements. They state that existing traditional models are incapable of dealing with these changes and there is a need for new models for better management of the ramp-up phase. Surbier et al. (2014) highlight the need for more quantitative models where the dynamics of the system can be simulated using dynamic system modelling techniques, such as Markov chains. Similarly, Glock and Grosse (2015) conclude that developments should focus on better forecasting models that account for a wider range of ramp-up characteristics and manage ramp-up efficiently. Colledani et al. (2018) bring to an end that the new frontier research in the ramp-up management is to exploit learning and data-analytics capabilities.

DSS solutions can be seamlessly integrated into existing frameworks, which provides access to semantically harmonized temporal data (Leitão et al., 2016). Moreover, these systems will have existing HMI that can be automatically reconfigured to new DSS solutions (Segura et al., 2018). There is a clear potential for the development of new solutions that take advantages of these features. A DSS for ramp-up needs to combine the ability to learn from previous ramp-up cases and to be adjusted online to the specific behaviour portrayed by a system. As any DSS, it should be composed of three main elements, the database (data collection), a data processing model (inference, reasoning, learning etc.) and an HMI (decision-support) (Guo et al., 2009; Mok, 2009). However, as it will be discussed in section II.C, the focus in the literature is placed on the data processing mechanism that creates the required knowledge not on building an interactive DSS for the ramp-up phase. This work considers the development in the field and defines a new comprehensive DSS framework that builds ramp-up data and processes it, enabling HMI. This DSS is designed to learn from data, recommend possible solutions, and adapt to new situations imposed by the system.

2.3. Knowledge and learning in ramp-up

Learning and knowledge sharing throughout the ramp-up process are a potential solution for ramp-up phase problems. Majority of existing works on the ramp-up highlight the importance of such promising potential in managing ramp-up related problems. Many studies focus on gaining knowledge effectively during the ramp-up phase. Terwiesch and Bohn (2001) model the effect of knowledge and learning on the final quality of the production output and by how much it can reduce the ramp-up time. Fjällström et al. (2009) classify the different sources of knowledge and information highlighting their effect on the operator's performance. Hansen and Grunow (2015) investigate the significance of the experience gained during ramp-up and link it to the cumulative production volume.

One of the important challenges for improving the ramp-up process is to gather knowledge and experiences. However, knowledge requires experimentation, something that is not always encouraged during ramp-up. At the ramp-up phase, market pressures to quick products delivery and cost reduction; therefore, there is no time available for experimentation (Doltsinis et al., 2012). Besides, previous studies conclude that long term investment in experimentation and learning from experimentation suffers from short term cost improvement (Terwiesch and Bohn, 2001).

One such strategy aiming to minimize changes during ramp-up is the Copy-Exactly (CE) approach (McDonald, 1997). In CE, the process is followed during the development phase of the first system and copied without any changes to ramp-up subsequent systems. Even in the CE case, the behaviour of the system is not exactly the same and new characteristics behaviour will emerge in later copies of the same system not observed previously. CE case shows the advantage of knowledge management between similar ramp-up cases, and the need to more effectively cope with the inherent variations of each individual system.

Neumann and Medbo (2017) study the operator's learning process and its effect on ramp-up time. They model human learning within a ramp-up discrete event simulator, showing that learning leads to more efficient decisions and shorter ramp-up time. Similarly, in Hansen and Grunow (2015) they develop a mixed integer linear programming model used as a decision support tool in an industrial case study. Their novelty lies on defining a new ramp-up performance function based on the effective capacity instead of time ramp-up time. Letmathe and Rößler (2019) investigate the effect of tacit knowledge transfer on spillover learning in ramp-ups. In a laboratory experiment, they analyse how this knowledge is transferred from one ramp-up process to another and between people and found that capturing and organizing knowledge is superior to transfer knowledge through observations. Indeed, real-time data is a cornerstone in the ramp-up phase. It facilitates learning and offers better decision support mechanism and thus speeds up and improves the ramp-up process (Schmitt et al., 2018).

2.4. Reinforcement learning in ramp-up

In machine learning, Reinforcement learning (RL) is the area where a software agent can take a sequence of actions to maximize a reward received from its environment. This capability allow RL approaches to deal with problems, where supervised and unsupervised learning is not able to deal with (Yau et al., 2012). Based on the learning mechanism, RL can be classified into two broad types: Model-based and Model-free learning. Model-based RL relays on planning to build an internal model for the actions and intermediate rewards using experience (policies). On the other hand, model-free RL relay on learning directly (trail-and-error or deliberative planning) from actions and reward without building any model (Sutton et al., 2018).

Both model-based and model-free methods have advantages and limitations. While model-based requires sufficient data to build the model, model-free elicit learning gradually and can learn from partial data (not complete policies) (Renaudo et al., 2015). Model-based methods can efficiently learn a policy and adapt to new problems via planning, reasoning about the model uncertainty. on the other hand, model-free methods can learn faster while also compensate for nonstationary environments (Dayan and Niv, 2008).

The potential of RL methods has been identified as promising approaches to learn from ramp-up experience and bring out more successful decision-making policies. Doltsinis and Lohse (2012) formalised the ramp-up process in a reinforcement learning set up and solved through a Monte Carlo based algorithm, where each ramp-up case is considered an episode. They suggested a model-free RL for a CE test case. Similarly, in Doltsinis et al. (2012), the implementation of QL-algorithm revealed fast convergence and time reduction. In Doltsinis et al. (2014), the authors suggested a formal MDP model for the capture and analysis of ramp-up process. The framework facilitates the reasoning and extraction the cause-and-affect relationships between actions and their impacts. Using the framework, a model-based RL is developed and shows that finding an optimal policy by learning from a few ramp-up cases is achievable. Later on, Doltsinis et al. (2017) investigate the symbiotic human-machine interaction to learn optimal actions, aiming at building effective ramp-up policies. They showed that using a human agent for guiding the exploration strategy significantly improved the policies quality using less datasets compared to random and greedy exploration strategies.

However, combining both model-free and model-based RL systems in one integrated model appears as a promising potential to endow a DSS with decision autonomy and flexibility to deal with uncertainty and unpredictability of the ramp-up phase. To the best of the author's knowledge, this paper presents the first approach that combines the benefits of both model-based and model-free learning to tackle the management of the ramp-up process.

3. Ramp-up Decision Support System

The functionality of a DSS is defined by the field of application that is reflected in its structure. For ramp-up, a DSS needs to provide supportive information while also suggest the most promising action, or a list of actions, that can enhance the system's performance for every occurring ramp-up state. This should be based on a well-defined decision-making process and knowledge acquired from experience. For this purpose, ramp-up is considered as a sequential process where an operator applies a sequence of actions to tune the system (Doltsinis et al., 2017). A ramp-up process $P_R = \left\{e_1 \ldots e_i E\right\}$ is formally defined as a sequence of experiences e_i. The experience set E at step i is defined as $E_i = \left\{s_i, a_i, s_{i+1} S, A\right\}$, the combination of a current ramp-up state s_i, a chosen action a_i, and the following state s_{i+1}, where S and A are the state and the action lists, respectively.

The proposed DSS monitors ramp-up and formalizes captured data into experiences e_i. The experiences are stored and fed to a learning model, which in turn used by the decision support mechanism to provide the operator with performance information and a proposed action for every ramp-up state. The core functionalities of the proposed DSS are analysing and learning from experiences. This supportive functionalities are enabled through capturing experience; a learning model and the decision support mechanism, respectively (cf. Fig. 1). Their specific functions are defined in the following sections.

3.1. Experience capturing

During the experience capture, the aim is to formally capture the operator's actions during the ramp-up process. The experience structured as a set of states with the operator's actions, indicating the transitions and allowing formalize ramp-up process as a Markov Decision Process (MDP). In fact, MDP becomes a de facto standard formalism for learning such sequential decision making (Wiering and van Otterlo, 2012). Also, previous works prove the benefits of formal representing ramp-up experience as an MDP (cf. section IID). The MDP requires a defined state space, a finite list of actions and an objective function that determines a reward. A brief overview of the detailed MDP model for ramp-up defined in (Doltsinis et al., 2014) is presented below.

- ***Ramp-up state space:*** The state space is based on the necessary process information and is composed of three main characteristics. These are the process functionality, the output product quality and the process optimization.
- ***Action list:*** The list should include actions that can directly affect the state variables. The emphasis is placed on the description of the action and not on its application. They are qualitatively defined but not quantitatively to allow the operator to judge the implementation. In this way, the generated policy from the learning model will be independent of how operators apply actions.
- ***Ramp-up reward***: The proposed reward for ramp-up is based on the performance measures proposed in Doltsinis et al. (2013) and defined as $R_{PM} = \frac{1}{n} w \cdot f_{RU}$, a linear combination of the system functionality ($\mathrm{f_f}$), the output quality ($\mathrm{f_q}$) and the process optimization ($\mathrm{f_o}$). The $\mathrm{w} = \left[\mathrm{w_f}, \mathrm{w_q}, \mathrm{w_o}\right]$ is the weight vector and $\mathrm{f_{RU}} = \left[\mathrm{f_f}, \mathrm{f_q}, \mathrm{f_o}\right]$ is the performance metrics vector. The reward function varies from -1 to 0 where -1 indicates too extreme ramp-up state. Parameter n is the number of process runs and acts as a normalization factor across several consecutive runs to account for stochastic variation in the state information of the system.
- ***Policy***: The policy (adjustment mechanism) is a sequence of actions to maximize the rewards. As the process behaviour captured as a probabilistic design, the return (recommended actions) considers all the possible state transitions based on the transition probability (cf. section III.B1). Policies are recommended to the operator as qualitative actions during the ramp-up phase.

3.2. Ramp-up learning

This work proposes a reinforcement learning approach in order to extract knowledge from experiences and formalize it in a reusable structure. In RL, the learning agent interacts with the system and captures its responses. Based on this, an optimal policy can then be found.

In ramp-up, the learning agent is the human operator who applies actions according to intuition and then monitors their effect on the system's response. This process contains the exploration/exploitation strategy implemented by the human operator where actions are intentionally chooses based on previous experience (exploitation) or experimenting (exploration) in unseen states. In human exploration, actions are not random but are based on the operator's intuition and knowledge (Doltsinis et al., 2017).

The second functionality of the proposed DSS uses the gathered experiences to extract an optimal policy or a ramp-up model based on which a policy is determined. The proposed DSS can incorporate both **learning a ramp-up model (model-based) and directly learning a policy (model-free)** depending on the ramp-up characteristic and the data availability, which will be enabled in two DSS operating outlined in sectionIIID.

3.2.1. Model-based learning

In model-based learning, algorithms aim at learning a model which is then used to extract a policy. Having a learned model can be very advantageous since it gives the ability to plan and test different strategies with different prediction horizon or award types etc. This allows analyzing and comparing the different results or even scrutinizing the ramp-up behavior in depth. However, this is not always possible in practice to learn a representative model since this requires a large amount of data with an adequate exploration strategy of the state space. This requires computational resources and time that are not always available during ramp-up and in real-time decision making.

The model-based approach is defined below in an algorithmic form.

- Given a finite number of unique ramp-up states $s \in S, \left|S\right| < \infty$ for a certain equipment/process and a listed of finite number of actions $\mathrm{a} \in \mathrm{A}, \left|\mathrm{A}\right| < \infty$, a number of episodes are accumulated and linked to the respective reward in forming the experience set as follows:

$$E = (s_t, s_{t+1}, a_{s_t} R_{s_t}) \quad (1)$$

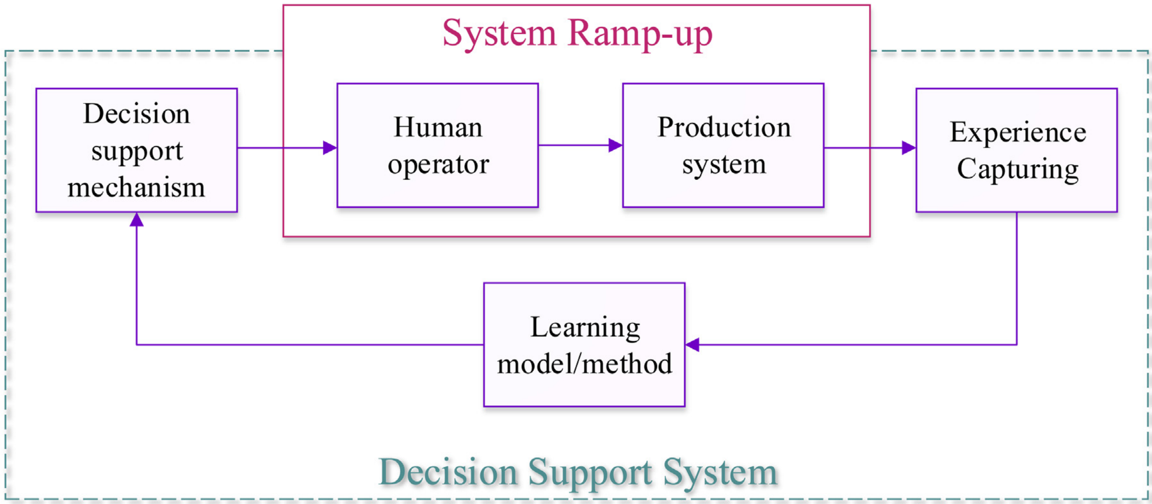

Fig. 1. Decision Support System overview.

- Given the state action pairs, the transition probabilities P are calculated to find a ramp-up model as follows:

$$P_{s_t a}\left(s^{'} s_t, a_{s_t}\right) \tag{2}$$

- Based on the extracted model and the state values $V(S)$ (cf. Eq. (3)), the state-action values $Q(S, A)$ are calculated for different planning horizons and set-ups. An optimal policy π is then extracted (cf. Eq. (4)) from choosing actions to accumulate the maximum rewards given the model transition probabilities P and state values $V^{\pi}(S)$.

$$V(S) = \max_{\pi} V^{\pi}(S) \tag{3}$$

$$\pi = \max_{a \in A} \sum_{s^{'} \in S} P_{sa}\left(s^{'}\right) V(s^{'}) \tag{4}$$

3.2.2. Model-free learning

Unlike model-based approaches, model-free learning aims at directly learning a policy. In this case, the learning agent finds an optimal policy based on the received rewards from every action, rather than learning the system's dynamics. This approach can provide results very fast for the visited states even if the policy is incomplete. In addition, the policy can be adjusted on the fly while new actions are applied and even if the system changes dynamically. This is a data-efficient approach, especially for online ramp-up approaches.

The description of policy learning is outlined below.

- Given a finite number of unique ramp-up states $s \in S$, $\left|S\right| < \infty$ for a certain equipment/process and a list of finite number of actions $a \in A$, $\left|A\right| < \infty$, a number of episodes are accumulated in the form of past experiences and linked with the respective reward (cf. Eq. (1)).
- Given that an exploration policy is followed. For every state action transition, the state action value $Q(s, a)$ can be calculated as follows.

$$Q(s, a) = E_{\pi}\left\{\sum_{k=0}^{T} \gamma^k r_{t+k+1} s_t = s\right\} \tag{5}$$

- Where $\max_{a^{'}} Q\left(s^{'}, a^{'}\right)$ is the maximum state action value in the following state $s^{'}$; α is the learning rate and γ is the discount factor, which will be further explained in the next part. The prediction horizon T updates the state-action values for every new state transition. The update rule is crucial for the quality of the learning results and differs between algorithms. It is essential to note that the update is based on the temporal difference between the experiences (cf. Eq. (6)).

$$Q^{'}(s, a) \longleftarrow Q(s, a) + \alpha\left[r + \gamma \max_{a^{'}} Q\left(s^{'}, a^{'}\right) - Q(s, a)\right] \tag{6}$$

- An action can be found that if applied, it will take the process to a better state. The policy is based on maximizing the accumulated reward that can vary from one step ahead up to all the steps until the end of the process.

$$\pi(s) = \underset{a}{argmax} Q(s, a) \tag{7}$$

The proposed DSS utilizes both the model-based and model-free approaches. The two different cases are realized in the two operating modes of the DSS.

3.3. Decision support mechanism

Both the experience capture and the learning model are used to eventually support the operator while carrying out the ramp-up process. First, the captured experience is used to provide information to the operator in the form of a ramp-up index and an action recommendation mechanism will use the extracted knowledge from the learning process that will be used as an action recommendation list. The learning model will not always be in a state to provide an action list with certainty. Hence, the operator should always be provided with all necessary information in order to take the best decision even without a recommendation. Therefore, the support mechanism projects a ramp-up index based on the performance of the process, showing a progress curve during ramp-up. Studies have shown the significant impact of progress quantification on the decision-making process and how such a measure can positively affect an operator's decision (Doltsinis et al., 2014; Neely, 2005), The ramp-up index works as instant feedback as well as tracking the long term progress of an operator.

The index is based on the performance measure (reward) presented in Eqs. (8)–(10). More specifically, four indices are presented, those of process **functionality**, process **quality** and process **optimization** based on (8), (9) and (10), respectively.

$$f_f(j) = -\sum_{j=1}^{n} k_j D_j \tag{8}$$

$$f_q(j) = -\sum_{j=1}^{n} \lambda_j Q_j \tag{9}$$

$$f_o(j) = -\sum_{j=1}^{n} \beta_j T_{o_j} \tag{10}$$

Parameters $\mathrm{k_j}$, λ_j, β_j are weights and $\mathrm{D_j}$, $\mathrm{Q_j}$, T_{o_j} are state parameters referring to the process functionality level, the product quality and the process optimization level, respectively (Doltsinis et al., 2014). The last metric of optimization is not always calculated since it only becomes meaningful once the system is fully functional. Therefore, it is calculated as follows.

$$\mathrm{f_o}(\mathrm{i,j}) = \begin{cases} \max \mathrm{f_0}, & \mathrm{f_f} < 0 \\ -\sum_{\mathrm{i}=1}^{\mathrm{m}} \frac{1}{s_t}\left(\sum_{\mathrm{j}=1}^{\mathrm{n}} \beta_\mathrm{j} \mathrm{T_j}\right)_\mathrm{i}, & \text{otherwise} \end{cases} \tag{11}$$

This reflects the significance of functionality and the correlations between the three metrics. Hence, the performance optimization metric is assumed to have the maximum negative value while the system is not yet functional.

3.4. DSS operation

Based on what was presented above, it is clear that ramp-up requires different decision support strategies that incorporate both model-based and model-free learning. These learning methods are activated through two operating modes, the offline and online. This is a result of the process data flow and depends on the systems ramp-up state and the previously acquired experience.

Fig. 2 presents the operational flow diagram of the proposed DSS, combining two modes of operation, namely online and offline. These are related to the requirement for decision support and the operation of the learning model. Ultimately, ramp-up is always an online process of changes and hence the offline acquired experience is combined with the online experience to generate a head start.

Once a ramp-up process is started, a process test-run is carried out to capture the state of the system and initialize the DSS. The ramp-up process then starts driven by an operator and the ramp-up status along with process information is captured. The information in the form of an experience set is passed for storage and model update. A query is sent to the process performance, which handles and presents the information in the form of an index. This process is repeated as many times as required by the operator in order to have a representative picture of the system's performance. With this information, the decision support mechanism is activated and the operator can take one action to support ramp-up. The learning mechanism is then activated and a different operation is followed according to the type of learning (online or offline). If the followed process is done online, the DSS activates the online learning mode that used to further support the process with a proposed action and predictions regarding its progress. If the process is done offline, the offline learning function is triggered and executed independently to ramp-up without any proposed actions. The operator can change from an online to an offline mode during the process. If the operator decides to make a query, the DSS activates the knowledge retrieval

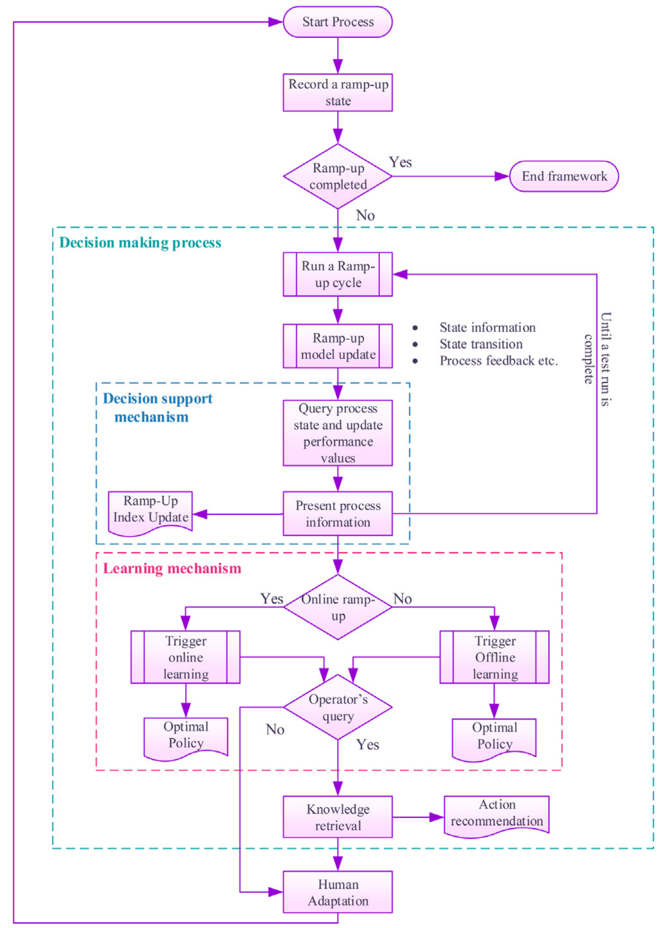

Fig. 2. Decision support system for ramp-up flow diagram.

mechanism and provides an action to support ramp-up. An action is then applied and the same process continues until the system is ramped-up.

The distinction between online and offline learning is mainly due to the different processing requirements of the algorithms and not to the process of ramp-up. Online learning algorithms aim to create results as soon as possible while offline algorithms have acquired a batch of data before processing them. The latter results will be more accurate. Online learning though is combining previous process information in order to efficiently initialize the model.

3.4.1. Offline operation

Fig. 2 shows how the DSS operates while ramp-up is carried out without presenting the detailed operation of the learning mechanism. The offline operation (cf. Fig. 3) of the DSS is focused on the data processing in an offline mode, meaning that processing time is not a major issue while the amount of data is also expected to be enough. In the offline mode, the system has access to a set of data without the ability to interact with the system. This is treated as a batch learning process. In batch learning, there is no interaction between the learner and the system. In other words, the data sets and the interactions are predefined during learning. That does not give the option of exploration to the learner. Consequently, the learning process is independent of the applied policy. Batch learning algorithms aim to find the best policy within the applied policies, rather than seeking for the optimal policy through interaction. New data sets are treated as an exploration phase. That gives a combination of online and offline learning practices, combined under the batch learning.

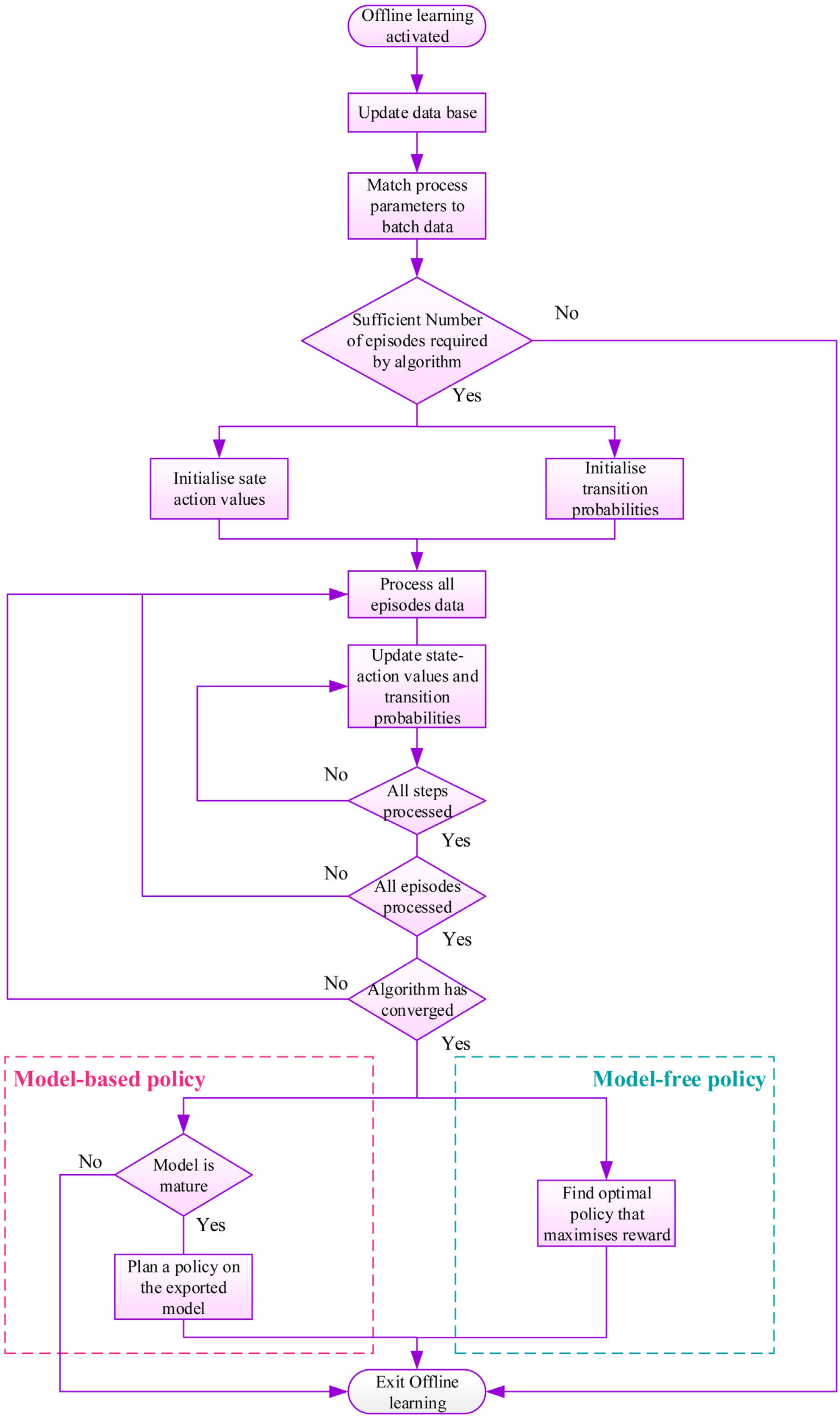

Fig. 3. Offline learning flow chart.

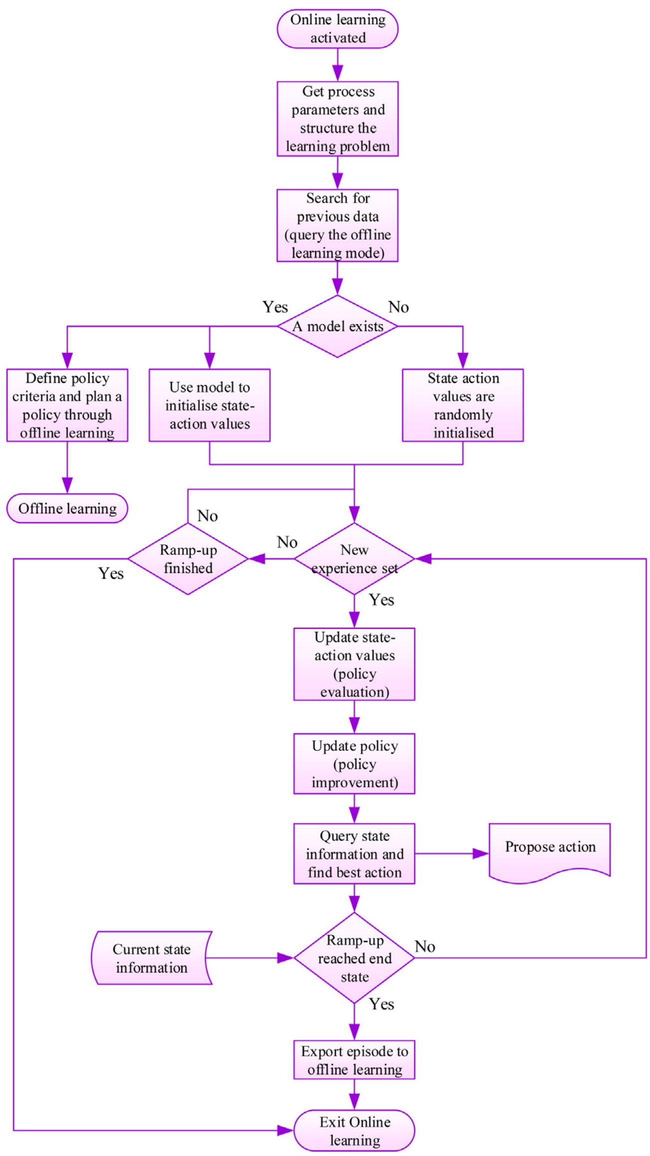

Fig. 4. Online learning flow chart.

3.4.2. *Online operation*

The second operating mode of the proposed DSS is focusing on processing the data online and step by step policy update, while also providing support during the process. The learning update rule changes in order to provide an evaluation of the currently applied actions. The offline knowledge will not be discarded but will be used for initialization of the algorithm. The system, in this case, can interact with the system (through the operator) and trigger exploration if necessary. The main difference between online and offline learning is the dynamic update of the policy based on the incoming new data. Additionally, due to the lack of data, the early learning process targets a direct learning approach. In Fig. 4, the flow chart of the online mode is presented.

4. Experimental procedure and results

In this part, the proposed DSS is implemented on an Industry 4.0 enabled assembly system to validate its functionality and ability to support the decision process. The DSS is tested under different scenarios to evaluate all operating modes. Following this section, the used equipment is presented, followed by implementation details and the experimental scenarios along with their results.

4.1. *Experimental equipment and DSS implementation*

For the experimental validation of the DSS, an assembly station of the microscale assembly system SMC HAS 200 is used. The assembly process is composed of three subroutines and delivers a small box filled with components (cf. Fig. 5). The station is fully automated composed of an electric motor, conveyor belt, pneumatic actuators and sensors. A Beckhoff industrial PC CX 1030 running soft PLCs for the control logic is used. The station is selected to implement the DSS for two main reasons; the ramp-up process complexity and the station capability. Ramp-up process is complex (cf. section II) and it will be very difficult to implement the DSS in a full-scale I4.0 application for the first time. It is more practical to start the implementation in such systems, taking lessons for the implementation in a full-industrial system. The second reason is the modularity of the station software and hardware components, which makes the implementation feasible.

The DSS is implemented on the above station as a JAVA software application and contains three software modules, the data collection process (experience collection), the data processing (learning mechanism), and the decision support mechanism through action recommendation and performance awareness. This effectively creates a Cyber-Physical System implementation of the station.

Data Collection: The data collection mechanism structures experience sets by combining a ramp-up state, the applied action by the operator and the new state. The information is labelled with the captured reward based on the ramp-up performance. The captured experience set is based on one or more running cycles according to that the operator's requirement to determine a state. This enables the temporal and semantically harmonized collection of all data as postulated by the Industrial Internet of Things.

Data processing: Data processing implements the learning process of the DSS. It receives experience sets and runs the learning algorithms according to the learning requirements (model-based/model-free; online/offline). The DSS interface allows the operator to change the ramp-up targets along with the weights of the performance measures (reward). That gives the option to reconfigure the process targets and the DSS can be applied to different systems or their variations. The learning element of the framework creates and refines a cyber-representation of the system.

Decision Support Mechanism: The mechanism has two functionalities to provide; performance awareness and action recommendations. Firstly, four performance measures (three performance metrics and the overall performance) are presented in graphs, while the process is carried out. The process state is also presented. A list of hierarchical actions is outlined once the learning mechanism can provide results. This feedback mechanism allows for operator-in-the-loop self-optimization and thereby creates a smart system.

An overview of the software architecture is shown in Fig. 6. The DSS connects the individual components of the process and provides an interface with the operator. The operator can import information into the DSS, such as the chosen action, any operational observations and control the tests. The DSS interacts with the PLC software to control the process and allow the exchange of information. Finally, the DSS interacts with Matlab where the learning element is implemented.

The interface of the DSS can be seen in Fig. 7. During a ramp-up cycle, an operator can trigger the DSS through the interface (start detection button), then the DSS triggers the production process and the data is collected automatically. After the end of the process, the operator observations are captured. An action is then

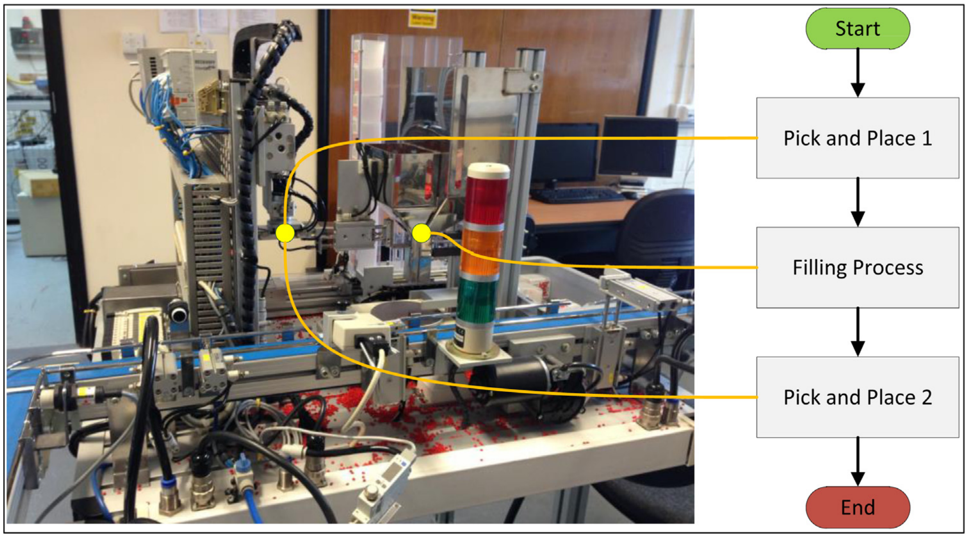

Fig. 5. SMC HAS 200 assembly station.

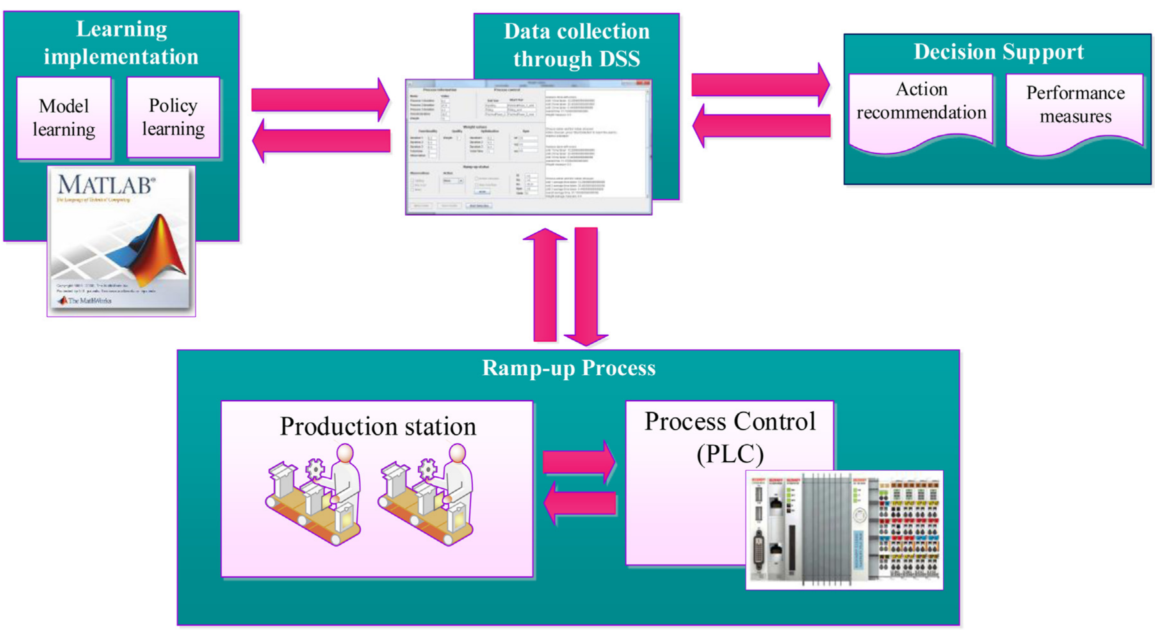

Fig. 6. CPS implementation overview.

recommended according to the ramp-up state. The operator has the option to follow the recommendation or choose a different action which is imported through the action list on the DSS interface. The operator has also the option to run another test to observe the process in more detail. Once the ramp-up process is on a satisfactory state the support process is stopped.

4.2. Experimental Results

For the DSS validation, four experimental scenarios are implemented and run 5 times from five different operators.

- ***Scenario 1***: Five ramp-ups are carried out by different operators without support. Results are used as a benchmark for the evaluation of the remaining scenarios. Only process information is presented; the cycle times and product quality.
- ***Scenario 2***: In the second scenario, the aim is to realize the effect of the performance measure support. Ramp-up is carried while the DSS projects the performance indices. Results are compared to those from scenario 1. Additionally, operators were asked to provide feedback on the functionality.
- ***Scenario 3***: This scenario assesses the functionality of the action recommendation and learning. This scenario considers model-free offline learning since it updates its policy at the end of every ramp-up process. Ramp-up is carried out by a different operator who has the option to follow the recommended action or divert. Online learning is assessed as if all ramp-ups constitute one big data set.
- ***Scenario 4***: The fourth scenario targets model-based learning, which is intended to support ramp-up in the long term. All previous ramp-ups are used to learn a model, which is evaluated.

For the reproducibility of the experiments, some of the operational aspects during the above scenarios are defined.

Operators: For scenario one and two, the same five operators are used in order to receive comparable results. Scenario three is

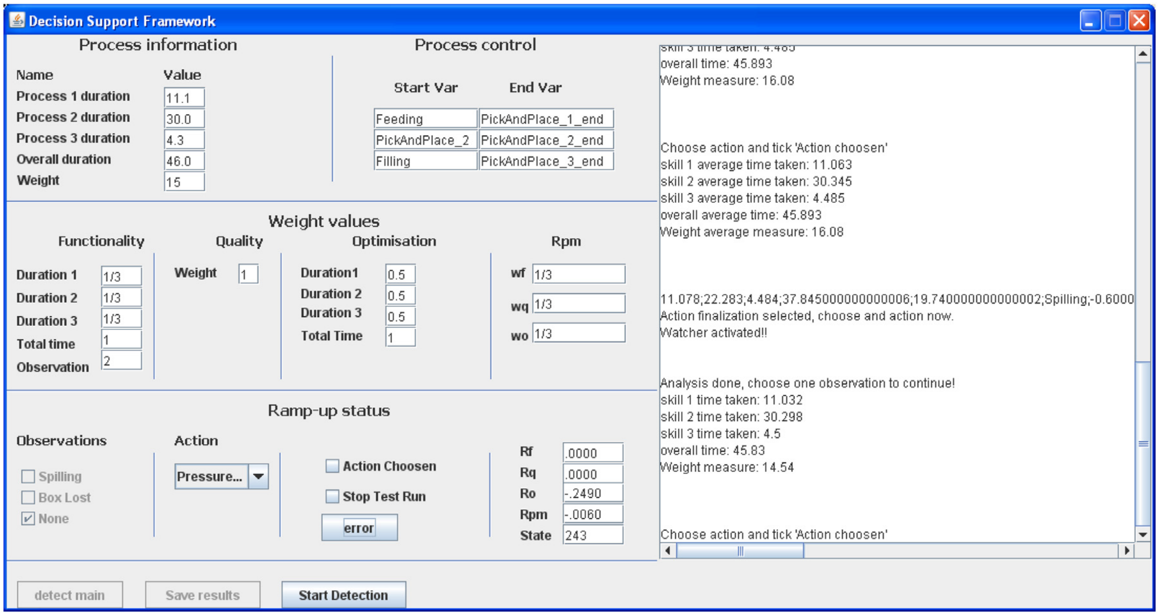

Fig. 7. Decision Support System interface.

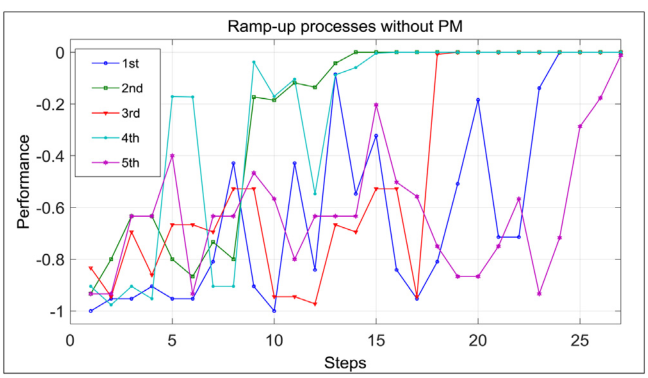

Fig. 8. Ramp-up processes without any support.

executed by different operators. The operators are engineers with little knowledge of the station's behavior provided just before the experiment. The different processes are explained in advance as well as the functionality of the individual components in terms of the stations' operational capabilities, its layout and its different elements.

Ramp-up process: The ramp-up process is crucial for the validation scenarios and it has to run under the same conditions in every case. First, the ramp-up state of the station is randomly initialized, by creating some critical error. The errors generate a number of disturbances during the ramp-up process which cause previously defined observations. For a ramp-up process to be considered complete, at least 4 out of 5 consecutive test runs should be successful.

Experimental procedure: In the experimental procedure, a list of actions is provided to the operators along with information regarding the status of ramp-up. According to the experimental scenario, the respective information is provided in every case through the decision support system. The operational requirements and the targets of the ramp-up process are provided at the beginning of the process and no further information is provided.

4.2.1. Scenario 1 – benchmarking

In this part, the results of the first scenario are presented in Fig. 8 where the performance graphs of the five ramp-up processes are presented.

The results show an average of 19.2 steps with a maximum of 27 and a minimum of 15 steps. The big difference between the minimum and maximum steps was expected and reflects the different understanding of the operators, the variability of strategies and applicability of actions. Results show that operators followed different strategies, focusing on different process targets. Some would prioritize product quality without considering other dysfunctional observations, while others would aim to achieve full functionality before they would resolve other issues.

An interesting observation appears for those cases where ramp-up is very long. In these cases, operators, who did not manage to solve the occurring problems, changed their strategy and different problems were targeted. This led to confusion, which did not allow them to understand the system's behavior resulting in longer ramp-ups. When the chosen actions had a positive effect, the targeted problems where fixed and the overall process appeared more straightforward. Finally, another interesting observation was the required time for the ramp-up process. Time varied between 1:30 to 2:30 h across all five scenarios. Some operators spent a lot of time identifying the best way of applying an action and its actual effect, while others experimented faster. The first managed to ramp-up the system faster although the initial actions were taking a lot more time. Their learning curve appeared to be steeper and received better results. The difficulty realizing the effect of actions during the initial steps was widely reported and the operators only managed to ramp-up the system they made the connection between the actions and their effect.

4.2.2. Scenario 2 – performance awareness

In scenario 2 the operators are provided with the performance measures that reflect the effect of a change to the system. Performance during ramp-ups is presented in Fig. 9.

Results show a minimum of 10 steps and a maximum of 16, creating an average of 12.8 steps. In comparison to scenario 1, this presents 6.4 steps (25%) average reduction. Comparing the ramp-up results of the same operators from both scenarios, all the ramp-ups

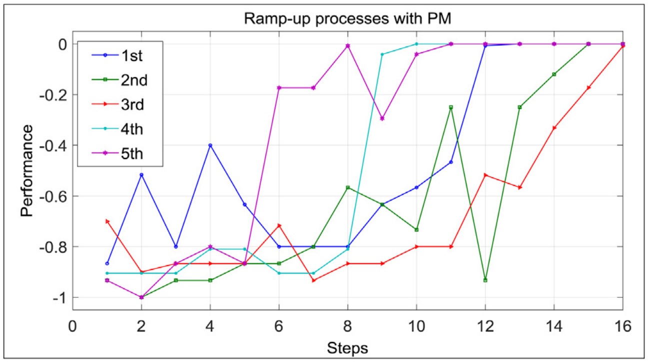

Fig. 9. Ramp-up processes with performance measures.

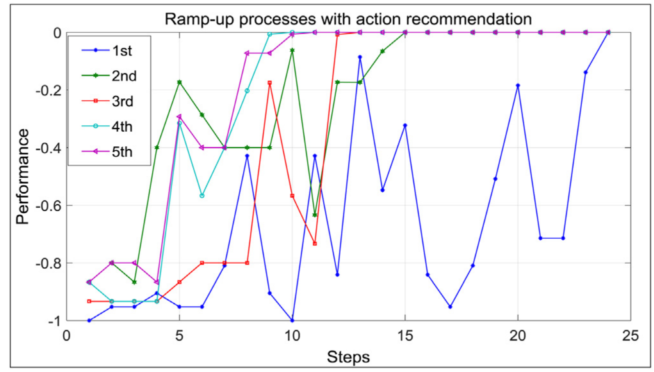

Fig. 10. Ramp-up processes with action recommendation.

appear reduced. The number of steps increases by just one in only one case. In terms of time, all ramp-ups were finished in less time with an average of 1:15' hour.

The operators appear to take corrective actions in those cases where an action had a negative effect. For instance, this is observable on the first, second and fifth curve, on the 3rd, 12th and 9th step, respectively. Operators also followed different strategies during the process. A gradual curve improvement shows that a step-by-step approach was followed compared to other cases where the big changes in performance reveal a bigger action effect.

Although it is assumed that the step reduction is a result of the DSS and the use of the performance measures, two points need to be made. Each ramp-up process is unique, starting from a random initial state; hence some processes might start from a better state than others and therefore require fewer actions. Second, since ramp-up is a stochastic problem, the same actions might result in different states and therefore applying the same policy for the same initial state might result in different process outcomes. Finally, it is important to note since the same operators from the first scenario were used, their experience and knowledge should be enhanced in parallel with the system. This might have resulted in better performance even without the performance feedback.

In order to get a better understanding of the DSS effect to the operators, a discussion was carried out with each one, after the process and some interesting points were made.

- All operators reported that the performance measures provided a clear picture of the ramp-up targets with a clear start and end state.
- Most of the operators claimed to gain an overall perception of the process progress through the performance measures. This was perceived to be clearer compared to just the raw state variables such as cycle times (as in Scenario 1).
- All operators reported that following the change of the performance measures after an action gave them the opportunity to reverse a bad action. That prevented the long ramp-up times resulting from extensive iterations without progress towards the final ramp-up state.

4.2.3. Scenario 3 – action recommendations

Results from the implementation of scenario 3 are presented in Fig. 10 and summarized in Table 1. Here the DSS provides a list of suggested ranked actions to an operator who has the option to accept the recommendation or apply another one.

The first process runs without any support since there is no initial experience in the system. Four more processes were carried out with the action recommendation functionality in place. Table 1 shows the number of steps and time required for ramp-up, the number of available actions from the recommendation system; how many times the operator accepted the recommendation and the number of times they accepted the second recommendation option.

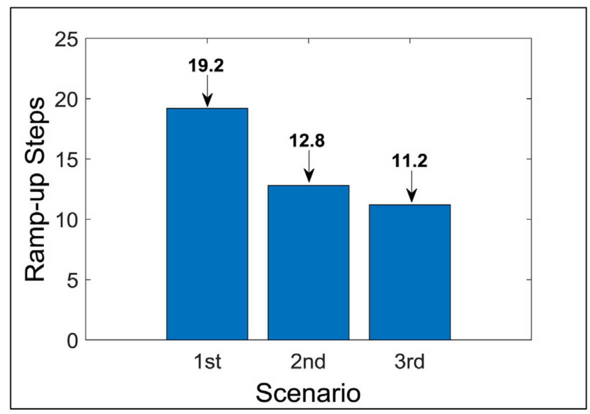

Fig. 11. Average ramp-up steps for the different scenarios.

Overall, the operators gave very positive feedback on the action recommendation mechanism and its effect at the beginning of the process where their knowledge was limited. After a few steps, the operators felt confident to divert from the recommendation. It is important to note that all the proposed actions guided the operator without the system getting stuck or oscillating between states. An interesting observation is that the operators rarely diverted from the proposed actions.

Finally, conclusions on overall DSS evaluation can be drawn by comparing the three scenarios (cf. Fig. 11). A significant improvement can be observed. Results improve gradually while the support is enhanced. The application of the DSS (scenario 3) in comparison to an unsupported ramp-up process (scenario 1) shows a ∼41% ramp-up time reduction.

4.2.4. Scenario 4 – model based learning

For this scenario, a model is generated based on the combined data sets captured from all the previous scenarios. The aim in this part is to assess the exported model and draw conclusions on its usability. There are 21 available data sets, comprising of 277 state transitions. Within these transitions, there are 29 unique states and the rest are reoccurring states coming from the exploration of the operator. Reviewing the model, the distribution of the experience across the state space is very different. Fig. 12 shows the number of visits for every occurred state. The number of visits varies between 1 and 46. Nine of the states have been visited 1 time, and 16 states less than five.

A closer look at state 142 where the most visits (state 142 occurred in 14 out of 19 episodes and 47 times) occurred, shows non-uniform distribution of action explorations. The actions cho-

Table 1
Results Summary with Action Recommendation.

Case	Steps/time(min) to completion	Recommendation availability	1st recommendation	2nd recommendation
1	14/50	14/14	8	2
2	12/39	8/12	6	2
3	9/40	8/9	5	2
4	10/42	9/10	7	2

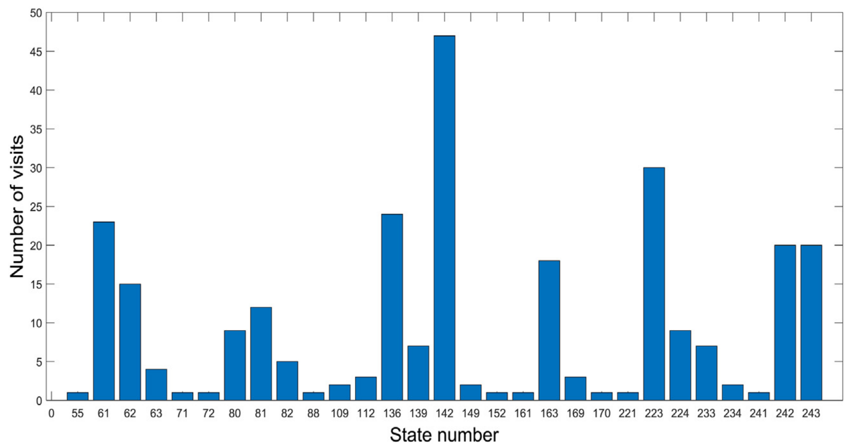

Fig. 12. Number of visits per state.

sen in state 142 are 1, 2, 3 and 5. They were chosen 19, 5, 10 and 13 times, respectively. This shows that even for the most visited state, the exploration of all actions is limited.

The exported model shows that only a part of the transition probabilities was learned. The model learning approach has strong assumptions, such as the requirement for a sufficient number of visits per state. These are opposite to the practical challenges faced during ramp-up, making it difficult to evaluate and use these results. Nevertheless, learning a complete and accurate model can generate an optimal policy that will benefit the ramp-up process. This is a matter of generating a significant amount of data outside the scope of this study. The functionality of learning a model is essential for the ramp-up DSS and the application of model-based techniques for partial models should be further investigated in future studies.

5. Discussion

The weights of the performance measures (w_f, w_q, w_o) influence the feedback and so actions of the next human agents. In the different stages of the ramp-up process, it is more practical to assign weights differently. At the first stage, a higher weight should be assigned to functionality, this is to encourage the human operator to focus on improving the functionality. The quality and optimization measures are at this stage not relevant as you can only improve a working system. Once the system is fully functional, the focus will be on quality and eventually on optimization. It is important to note that in this study, the human operator assigns the weights, which can be freely updated at each stage. However, at certain stages a higher weight may be given for a measure that should not be the focus for that state of the system. This might led to a prolonged ramp-up process. Therefore, it will be worth to investigate the design of a weight interval for each indicator at different ramp-up stages. The weight interval can vary from one production system to another, allowing more specificity of the system and enhance the genericity of the model.

In this work, RL is used with a reward that is calculated by evaluating the performance gain of changes in the system. In order to ensure a better representation of the ramp-up process, an accumulated reward for the policy is proposed. This enables the resilience of the accumulated results to minor variations in the performance. The results demonstrate a good performance of the designed reward function, however it is important to note that RL could use different reward function based on the transition of states (Wiering and van Otterlo, 2012). While this is worth to investigate, it was considered outside of the scope of this work. Considering such reward function may improve the fidelity of the result as both the state context and actions contribute to the possibility of selecting successful actions.

In this paper, actions are recommended qualitatively not quantitively to allow the operator to decide the implementation of actions. This implies that the learning process is independent of how actions are applied. A recommender system quantitative action is a technique that allows the operator to implement actions easily. Instead of including an increase or decrease a parameter value (i.e. pressure) hundreds of values will be in the list for each parameter. Therefore, this would explode the solution space and make it unmanageable to quantify in the ramp-up process. Another aspect to consider is the amount of options available for the operator, as this might lead to unmanageable amount of parameter options. Enhancements of CPS DSS, could tackle this by using multi-criteria techniques for ranking actions based on the expected future improvement on the system performance.

6. Conclusions

In this study, a Decision Support System was defined following the Industry 4.0 Cyber-Physical System design principles to learn the behaviour model of a production system during ramp-up. The DSS was evaluated in a CPS enabled microscale assembly station. Results indicate that supporting human operators during system ramp-up with performance measures can significantly reduce both the required time and the number of steps to ramp-up completion. Providing further support by recommending the most appropriate set of ramp-up actions further reduced both the time and number of steps required for ramp-up. Qualitative feedback from the operators regarding the individual elements of the DSS indicates a very positive utility of the support provided during ramp-up. Users reported that the DSS enhanced their understanding of the process's progress and its goals. Although learning a model can eventually provide an optimal policy, the amount of data required poses a significant challenge when compared to a feedback mechanism. The 41% ramp-up time reduction demonstrates the significant support of the proposed DSS during the ramp-up process. These are very promising results that should be further evaluated in larger scale production systems.

In addition to the points elaborated in the discussion section, there is additional potential for further investigation of the proposed DDS. One aspect is the transferability of model-based learning from one system to another (i.e. from one production line to other different types of the production lines). This would highlight the common and different characteristics between them and enable a wider generalization of the proposed solution. This could potentially reduce the required number of iterations, which would result in time gains for the ramp-up process. Another aspect to consider is the use of suggested DSS in more complex and diverse production environments. This is critical to ensure the generic applicability of the solution, but also to identify improvement opportunities.

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.

References

Ball, P.D., Roberts, S., Natalicchio, A., Scorzafave, C., 2011. Modelling production ramp-up of engineering products. Proc. Inst. Mech. Eng. Part B J. Eng. Manuf. 225, 959–971, http://dx.doi.org/10.1177/09544054JEM2071.

Carrillo, J.E., Franza, R.M., 2006. Investing in product development and production capabilities: the crucial linkage between time-to-market and ramp-up time. Eur. J. Oper. Res. 171, 536–556, http://dx.doi.org/10.1016/J.EJOR.2004.08.040.

Ceglarek, D., Shi, J., Wu, S.M., 1994. A knowledge-based diagnostic approach for the launch of the auto-body assembly process. J. Manuf. Sci. Eng. Trans. ASME 116, 491–499, http://dx.doi.org/10.1115/1.2902133.

Colledani, M., Tolio, T., Yemane, A., 2018. Production quality improvement during manufacturing systems ramp-up. CIRP J. Manuf. Sci. Technol. 23, 197–206, http://dx.doi.org/10.1016/j.cirpj.2018.07.001.

Couretas, J.M., Zeigler, B.P., Subramanian, I., Sarjoughian, H., 2001. Capacity analysis for mixed technology production: evaluating production ramp resource modifications via distributed simulation. Int. J. Prod. Res. 39, 163–184, http://dx.doi.org/10.1080/00207540010005051.

Dalenogare, L.S., Benitez, G.B., Ayala, N.F., Frank, A.G., 2018. The expected contribution of Industry 4.0 technologies for industrial performance. Int. J. Prod. Econ. 204, 383–394, http://dx.doi.org/10.1016/J.IJPE.2018.08.019.

Dayan, P., Niv, Y., 2008. Reinforcement learning: the good, the bad and the ugly. Curr. Opin. Neurobiol. 18, 185–196, http://dx.doi.org/10.1016/j.conb.2008.08.003.

Doltsinis, S.C., Lohse, N., 2012. a model-free reinforcement learning approach using Monte Carlo method for production Ramp-up policy improvement - a copy exactly test case. IFAC Proc. Vol., IFAC, 1628–1634, http://dx.doi.org/10.3182/20120523-3-RO-2023.00288.

Doltsinis, S., Ferreira, P., Lohse, N., 2014. An MDP model-based reinforcement learning approach for production station ramp-up optimization: Q-learning analysis, IEEE Trans. Syst. Man, Cybern. Syst. 44, 1125–1138, http://dx.doi.org/10.1109/TSMC.2013.2294155.

Doltsinis, S., Ferreira, P., Lohse, N., 2017. A symbiotic human-machine learning approach for production ramp-up. IEEE Trans. Human-Machine Syst. 48, 229–240, http://dx.doi.org/10.1109/THMS.2017.2717885.

Doltsinis, S.C., Ratchev, S., Lohse, N., 2013. A framework for performance measurement during production ramp-up of assembly stations. Eur. J. Oper. Res. 229, 85–94, http://dx.doi.org/10.1016/J.EJOR.2013.02.051.

Doltsinis, S., Ferreira, P., Lohse, N., 2012. Reinforcement learning for production ramp-up: a Q-batch learning approach. Proc. - 2012 11th Int. Conf. Mach. Learn. Appl. ICMLA 2012 1, 610–615, http://dx.doi.org/10.1109/ICMLA.2012.113.

Dombrowski, U., Wullbrandt, J., Krenkel, P., 2018. Industrie 4.0 in production ramp-up management". Procedia Manuf. 17, 1015–1022, http://dx.doi.org/10.1016/j.promfg.2018.10.085.

Fjällström, S., Säfsten, K., Harlin, U., Stahre, J., 2009. Information enabling production ramp-up. J. Manuf. Technol. Manage. 20, 178–196, http://dx.doi.org/10.1108/17410380910929619.

Glock, C.H., Grosse, E.H., 2015. Decision support models for production ramp-up: a systematic literature review. Int. J. Prod. Res. 53, 6637–6651, http://dx.doi.org/10.1080/00207543.2015.1064185.

Glock, C.H., Jaber, M.Y., Zolfaghari, S., 2012. Production planning for a ramp-up process with learning in production and growth in demand. Int. J. Prod. Res. 50, 5707–5718, http://dx.doi.org/10.1080/00207543.2011.616549.

Guo, Z.X., Wong, W.K., Leung, S.Y.S., Fan, J.T., 2009. Intelligent production control decision support system for flexible assembly lines. Expert Syst. Appl. 36, 4268–4277, http://dx.doi.org/10.1016/J.ESWA.2008.03.023.

Haller, M., Peikert, A., Thoma, J., 2003. Cycle time management during production ramp-up. Robot. Comput. Manuf. 19, 183–188, http://dx.doi.org/10.1016/S0736-5845(02)00078-9.

Hansen, K.R.N., Grunow, M., 2015. Modelling ramp-up curves to reflect learning: improving capacity planning in secondary pharmaceutical production. Int. J. Prod. Res. 53, 5399–5417, http://dx.doi.org/10.1080/00207543.2014.998788.

Heine, I., Beaujean, P., Schmitt, R., 2016. Improving production ramp-up through human resource methods. Qual. Manage. J. 23, 7–19, http://dx.doi.org/10.1080/10686967.2016.11918459.

Kalir, A.A., Rozen, K., 2018. Optimizing starts for capacity, velocity, and output during the ramp-up period of a semiconductor fab. Proc. - Winter Simul. Conf. 2018-Decem, 3494–3501, http://dx.doi.org/10.1109/WSC.2018.8632471.

Kukulies, J., Schmitt, R., 2018. Stabilizing production ramp-up by modeling uncertainty for product design verification using Dempster–Shafer theory. CIRP J. Manuf. Sci. Technol. 23, 187–196, http://dx.doi.org/10.1016/j.cirpj.2017.09.008.

Leitão, P., Colombo, A.W., Karnouskos, S., 2016. Industrial automation based on cyber-physical systems technologies: prototype implementations and challenges. Comput. Ind. 81, 11–25, http://dx.doi.org/10.1016/J.COMPIND.2015.08.004.

Letmathe, P., Rößler, M., 2019. Tacit knowledge transfer and spillover learning in ramp-ups. Int. J. Oper. Prod. Manage. 39, 1099–1121, http://dx.doi.org/10.1108/IJOPM-08-2018-0508.

Liu, Y., Peng, Y., Wang, B., Yao, S., Liu, Z., 2017. Review on cyber-physical systems. IEEE/CAA J. Autom. Sin. 4, 27–40, http://dx.doi.org/10.1109/JAS.2017.7510349.

Manna, S.K., Chaudhuri, K.S., 2006. An EOQ model with ramp type demand rate, time dependent deterioration rate, unit production cost and shortages. Eur. J. Oper. Res. 171, 557–566, http://dx.doi.org/10.1016/J.EJOR.2004.08.041.

Manna, S.K., Chiang, C., 2010. Economic production quantity models for deteriorating items with ramp type demand. Int. J. Oper. Res. 7.

McDonald, C.J., 1997. Copy EXACTLY! A paradigm shift in technology transfer method. IEEE/SEMI Adv. Semicond. Manuf. Conf. Work., 414–417, http://dx.doi.org/10.1109/asmc.1997.630771.

Meisel, F., Glock, C.H., 2018. Self-induced learning vs. Project-based supplier development for production ramp-up with two supply options. Int. J. Prod. Econ. 198, 60–69, http://dx.doi.org/10.1016/j.ijpe.2018.01.029.

Mok, P.Y., 2009. A decision support system for the production control of a semiconductor packaging assembly line. Expert Syst. Appl. 36, 4423–4430, http://dx.doi.org/10.1016/J.ESWA.2008.05.021.

Neely, A., 2005. The evolution of performance measurement research: developments in the last decade and a research agenda for the next. Int. J. Oper. Prod. Manage. 25, 1264–1277, http://dx.doi.org/10.1108/01443570510633648.

Nembhard, H.B., Birge, J.R., 1998. A startup procedure for process industries using a multiple objective nonlinear program. IIE Trans. (Institute Ind. Eng.) 30, 291–300, http://dx.doi.org/10.1080/07408179808966470.

Neumann, W.P., Medbo, P., 2017. Simulating operator learning during production ramp-up in parallel vs. serial flow production. Int. J. Prod. Res. 55, 845–857, http://dx.doi.org/10.1080/00207543.2016.1217362.

Pacaux-Lemoine, M.-P., Trentesaux, D., Zambrano Rey, G., Millot, P., 2017. Designing intelligent manufacturing systems through Human-Machine Cooperation principles: a human-centered approach. Comput. Ind. Eng. 111, 581–595, http://dx.doi.org/10.1016/J.CIE.2017.05.014.

Pal, S., Mahapatra, G.S., Samanta, G.P., 2014. An EPQ model of ramp type demand with Weibull deterioration under inflation and finite horizon in crisp and fuzzy environment. Int. J. Prod. Econ. 156, 159–166, http://dx.doi.org/10.1016/J.IJPE.2014.05.007.

Renaudo, E., Chatila, R., Khamassi, M., 2015. Respective advantages and disadvantages of model-based and model-free reinforcement learning in a robotics neuro-inspired cognitive architecture. Procedia Comput. Sci. 71, 178–184, http://dx.doi.org/10.1016/j.procs.2015.12.194.

Schmitt, R., Heine, I., Jiang, R., Giedziella, F., Basse, F., Voet, H., Lu, S., 2018. On the future of ramp-up management. CIRP J. Manuf. Sci. Technol. 23, 217–225, http://dx.doi.org/10.1016/j.cirpj.2018.03.001.

Schuh, G., Gartzen, T., Wagner, J., 2015. Complexity-oriented ramp-up of assembly systems. CIRP J. Manuf. Sci. Technol. 10, 1–15, http://dx.doi.org/10.1016/j.cirpj.2015.05.007.

Segura, Á., Diez, H.V., Barandiaran, I., Arbelaiz, A., Álvarez, H., Simões, B., Posada, J., García-Alonso, A., Ugarte, R., 2018. Visual computing technologies to support the Operator 4.0. Comput. Ind. Eng., 105550, http://dx.doi.org/10.1016/J.CIE.2018.11.060.

Surbier, L., Alpan, G., Blanco, E., 2014. A comparative study on production ramp-up: state-of-the-art and new challenges. Prod. Plan. Control. 25, 1264–1286, http://dx.doi.org/10.1080/09537287.2013.817624.

Sutton, R.S., Barto, A.G., Bach, F., 2018. Reinforcement Learning: An Introduction, second. The MIT press, Cambrideg Massachuestts, London, UK.

Terwiesch, C., Bohn, R.E., 2001. Learning and process improvement during production ramp-up. Int. J. Prod. Econ. 70, 1–19, http://dx.doi.org/10.1016/S0925-5273(00)00045-1.

Terwiesch, C., Xu, Y., 2004. The copy-exactly ramp-up strategy: trading-off learning with process change. IEEE Trans. Eng. Manage. 51, 70–84, http://dx.doi.org/10.1109/TEM.2003.822465.

Uhlemann, T.H.J., Lehmann, C., Steinhilper, R., 2017. The digital twin: realizing the cyber-physical production system for industry 4.0. Procedia CIRP 61, 335–340, http://dx.doi.org/10.1016/j.procir.2016.11.152.

Vits, J., Gelders, L., Pintelon, L., 2006. Production process changes: a dynamic programming approach to manage effective capacity and experience. Int. J. Prod. Econ. 104, 473–481, http://dx.doi.org/10.1016/J.IJPE.2004.08.007.

Wiering, M., van Otterlo, M., 2012. Reinforcement Learning State-of-the-art. Springer, London, UK, http://dx.doi.org/10.1007/978-3-642-27645-3.

Yau, K.-L.A., Komisarczuk, P., Teal, P.D., 2012. Reinforcement learning for context awareness and intelligence in wireless networks: review, new features and open issues. J. Netw. Comput. Appl. 35, 253–267, http://dx.doi.org/10.1016/J.JNCA.2011.08.007.

Zülch, G., 2006. Modelling and simulation of human decision-making in manufacturing systems. Proc. Winter Simul. Conf., 947–953, http://dx.doi.org/10.1109/WSC.2006.323180.