CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

ORCA – Online Research @ Cardiff

This is an Open Access document downloaded from ORCA, Cardiff University's institutional repository:https://orca.cardiff.ac.uk/id/eprint/132913/

This is the author's version of a work that was submitted to / accepted for publication.

Citation for final published version:

Oliff, Harley , Liu, Ying , Kumar, Maneesh , Williams, Michael and Ryan, Michael 2020. Reinforcement learning for facilitating human-robot-interaction in manufacturing. Journal of Manufacturing Systems 56 , pp. 326-340. 10.1016/j.jmsy.2020.06.018

Publishers page: http://dx.doi.org/10.1016/j.jmsy.2020.06.018

Please note:
Changes made as a result of publishing processes such as copy-editing, formatting and page numbers may not be reflected in this version. For the definitive version of this publication, please refer to the published source. You are advised to consult the publisher's version if you wish to cite this paper.

This version is being made available in accordance with publisher policies. See http://orca.cf.ac.uk/policies.html for usage policies. Copyright and moral rights for publications made available in ORCA are retained by the copyright holders.

Reinforcement Learning for Facilitating Human-Robot-Interaction in Manufacturing

Harley Oliff[a], Ying Liu[a*], Maneesh Kumar[b], Michael Williams[c] and Michael Ryan[a]

[a]*Institute of Mechanical and Manufacturing Engineering, School of Engineering, Cardiff University, Cardiff, CF24 3AA, UK*

[b]*Cardiff Business School, Cardiff University, Cardiff, CF10 3EU, UK*

[c]*Olympus Surgical Technologies Europe, OSTE, Cardiff, UK*

[*]*Corresponding author. E-mail: LiuY81@Cardiff.ac.uk*

Abstract

For many contemporary manufacturing processes, autonomous robotic operators have become ubiquitous. Despite this, the number of human operators within these processes remains high, and as a consequence, the number of interactions between humans and robots has increased in this context. This is a problem, as human beings introduce a source of disturbance and unpredictability into these processes in the form of performance variation. Despite the natural human aptitude for flexibility, their presence remains a source of disturbance within the system and make modelling and optimization of these systems considerably more challenging, and in many cases impossible. Improving the ability of robotic operators to adapt their behaviour to variations in human task performance is, therefore, a significant challenge to be overcome to enable many ideas in the larger intelligent manufacturing paradigm to be realised. This work presents the development of a methodology to effectively model these systems and a reinforcement learning agent capable of autonomous decision-making. This decision-making provides the robotic operators with greater adaptability, by enabling its behaviour to change based on observed information, both of its environment and human colleagues. The work extends theoretical knowledge on how learning methods can be implemented for robotic control, and how the capabilities that they enable may be leveraged to improve the interaction between robots and their human counterparts. The work further presents a novel methodology for the implementation of a reinforcement learning-based intelligent agent which enables a change in behavioural policy in robotic operators in response to performance variation in their human colleagues. The development and evaluation are supported by a generalized simulation model, which is parameterized to enable appropriate variation in human performance. The evaluation demonstrates that the reinforcement agent can effectively learn to make adjustments to its behaviour based on the knowledge extracted from observed information, and balance the task demands to optimise these adjustments.

Keywords: Intelligent Manufacturing; Reinforcement Learning; Human-Robot Interaction; Human Factors; Adaptability;

1. Introduction

The adoption of automation within manufacturing processes has experienced accelerated growth over the past decade, which shows no signs of imminent abatement. Due in part to governmental and industry initiatives, like the German developed Industry 4.0 [1], the increased adoption of automated systems is perhaps the inevitable conclusion of concurrent developments in several relevant areas [2]. These advancements, particularly within computer science and control, have enabled autonomous systems to leverage intelligent processing of data to generate process knowledge, which may be used to inform behaviour [3-5].

Despite the rapid adoption of automation technologies within manufacturing processes, the transition from manual to robotic processes is gradual, and many operations require a level of dexterity, flexibility, or adaptability which remains unachievable by contemporary robotic systems. The result of which is an increasing number of interactions between humans and their machine counterparts. Whilst such a combined approach has numerous advantages (combining the strengths of both disciplines), it also introduces significant complexities to the system and necessitates the study and understanding of how best to facilitate collaborative behaviour within these new robotic elements [4, 6].

The presence of human operators within automated systems presents such a problem, as repeatability and accuracy are arguably the hallmarks of robotics, and conversely the main traits which prevent human beings from remaining competitive. Reconciliation of these behaviours is a challenge that remains to be overcome, for manufacturing systems to fully realize the potential that robotics and automation can provide.

The variation in human task performance exists both between individuals and between iterations of the same task, temporally dependent on a wide range of other human factors, most notably manifesting

thorough fatigue [7-10]. This variation introduces disturbance through a disparity in performance that is dynamic and prevents robotic systems and their behaviours from being optimized at the process level.

Improving the capacity for adaptability and flexibility of robotic systems within manufacturing is seen as both a key indicator of the level of a system's intelligence [11] and provides a solution for overcoming the performance disparity within human-machine interaction, mitigating the effect on the system. Reducing this performance disparity can also be said to improve the fluency of the interactions themselves [12], and enable manufacturing processes to move towards a one-piece-flow, in line with the modern application of lean-manufacturing methods [13, 14].

Enabling an agent, such as a robotic operator, to intelligently process observations of their environment, to make decisions about their actions and operating parameters, has already been demonstrated to improve the ability of numerous control and autonomous decision-making systems. The most effective approaches utilising reinforcement-based machine learning, typically employing neural networks to make predictions for the value of an action based on an observed state. The reinforcement approach has enabled similar agents to be trained without supervision to achieve additional capabilities and human level task performance in many domains [15-17]. However, there remains a lack of research on how best to apply these capabilities in the in terms of adjusting behaviours to improve adaptability, and even less on how such agents and the abilities may be leveraged to reduce disparity in performance and its resultant influence on the system as a whole, in scenarios with complex system dynamics, where such approaches have proven successful.

This work presents a study of applying the principles of reinforcement learning to robotic control, under the context of HRI within manufacturing processes. A methodology is presented based on the presented literature review, to define the necessary capabilities of a reinforcement learning agent for robotic process control, given sufficient knowledge of the process and the performance and behaviours of the human counterparts.

2. Literature Review

The following section presents a review of relevant literature and is divided into three key areas of relevance. Further study on the current state of intelligent manufacturing and the applications of learning and intelligence are considered, the enabling machine learning technologies are discussed in the second section, and the third section presents current work in the application of these technologies and methods to Human-Robot-interaction.

2.1. Intelligent Manufacturing

Intelligent Manufacturing is the term given to research and work covering the wide variety of disciplines and applications necessary to integrate intelligence within modern manufacturing systems [18, 19]. Advances in the capabilities of computational systems have enabled numerous aspects of manufacturing to be improved through the use of observation, and appropriate processing of data generated by these processes [20]. Undoubtedly, it is the increasing rate of data generation, and the ease and widespread practice of data collection within the manufacturing industry, which is responsible for the emergence of intelligent manufacturing [21, 22]. Processing of this data enables the Digitalization of manufacturing systems, as the processes and events can be captured, processed, and analysed to build digital representations of these systems [23-25] for modelling, prediction and optimization. The world that exists today is one of continuous and widespread data collection. The moral concerns over the appropriate and moral usage of such data, especially where such data is human-generated are beyond the scope of this work but represent an important area for discussion.

The digitalization of these systems is crucial for realizing Cyber-Physical-Systems (CPS)[11, 26, 27], combinations of hardware and software components which are characterized by their capacity for intelligent behaviours. Advanced systems are capable of adaptation and autonomous decision making, based on their observations of the environment [28] facilitating flexibility and optimization in

manufacturing processes [29-31]. Digitalization of manufacturing systems has in itself given rise to focused work on how digital representations can be used for system control [24]; although this has identified a key challenge that is presented by advanced digitalization methods. As the representations of production processes become increasingly detailed and complex, contemporary hierarchal structures present some problems and challenges when faced with enabling intelligent systems to behave in a manner which is autonomous and adaptable [32, 33].

To overcome this, systems based on the distribution of control aim to divide and distribute the control problem into multiple tasks with their own computational demands, which are distributed to a number of agents representing the elements in the system. These agents may be entirely virtual, or a combination of hardware and software which forms a logical unit within the system referred to as a *holon* [34, 35]; with these being the common constituents of CPS's. The software-based nature of these agents additionally facilitates a simulation-based approach, as the necessary structure closely resembles that of Object-Oriented programming languages. Individual agents can be represented within a process simulation as instances of an object; each with protected and distinct internal and external structures and functions to facilitate their behaviour. These agents then coordinate their actions to achieve the given goal, with the result of decreasing the overall system complexity. Importantly, these agents have individual autonomy and are governed by their own cumulative experiences, enabling agents with an identical structure to learn optimal behaviours based on the systems they are presented with [36]. The use and capabilities of multi-agent systems have continued to increase, with a number of demonstrable applications in the manufacturing sector [37, 38]. The variation in behaviour that distributed control enables places simulation as a critical tool in developing distributed control systems, as the use of simulation is a powerful tool enabling the design, evaluation and subsequent optimization of intelligent agent performance on representative and varied tasks in a repeatable environment [39]. A detailed overview of intelligent agents can be found in [40, 41].

To enable intelligent processing of sensory data, and coordination of actions with those of others, significant consideration must be given to the agent’s control structure, and how it is implemented to

enable an adaptable and appropriate response. This requires consideration of collaborative behaviours, and the internal structure of learning and data analysis. Whilst of course teamwork is not unique to Homo Sapiens, collaboration is commonly considered a very human trait, many studies on collaboration have sought to understand cognitive processes as they occur in humans, and to replicate these *cognitive* processes. This field of work has existed for many decades as a branch of neuropsychology, and numerous *cognitive architectures* [42], which define the structure of control systems, enable intelligent behaviour to develop for various purposes. A common feature among many is a modularized structure, with multiple interacting separate elements responsible for different aspects of cognition. The modules contained within each architecture are often structured around a central module which acts as a controller for internal processes and the sharing of information between modules. This module is then extended by other *modules* to facilitate necessary behaviours; such as Perception, Learning, Decision-Making, and Memory. This serves to further reduce the complexity of control and facilitates the integration of low-level perceptual and motor control systems with higher-level knowledge extraction and decision-making processes [43]. The isolation of these elements enables established control techniques for image capture and robotic motion planning to be used for control alongside higher-level processing without interference. This separation is analogous to the distinction between two types of cognitive processing found in human beings; *Type 1* and *Type 2*. The former fast and intuitive; the latter slower and analytical [44]. Both types of processing are necessary, dependent on the situation at hand. *Type 1* processing is typical in familiar situations where rapid response time is required, and where a large number of points of observation exist simultaneously. Conversely, situations where *Type 2* processing is dominant, are those where response time is non-critical and focus on the specific relationships between a relatively small number of observations. These situations are typically unfamiliar, and analytical reasoning is used to identify relationships, to form appropriate behaviours [45, 46].

Many architectures exist to best achieve a level of *cognition*. Notable examples of these architectures include ACT [47], SOAR [48], and C4 [49]. These architectures, whilst conceptualized before the current capabilities of computational systems were fully understood, retain several insights with regards

to structure and interaction, which are still valid today. There is, however, a wide variance in their application and capabilities; and many problems must still be resolved for effective integration and use of such methodologies in addition to the questions of agent behaviour and design, including the standardization and issues relating to connectivity, and security[50, 51].

Recent applications of *intelligence* to these autonomous agents have made use of machine learning (typically a neural network-based approach) to automate each agent's control and analytical processes, enhance the cumulative experience of each agent, and to control the decision-making processes [52-55].

2.2. *Recent Machine Learning advances in Manufacturing*

With increasingly high-profile success stories in recent years, the emphasis on the utilization of machine learning for intelligent data processing continues to grow, and consequently its adoption in intelligent manufacturing methodologies [56]. Continual developments in computer science, of both hardware and software, have enabled machine learning techniques to become ever more powerful, and increasingly feasible to implement on accessible hardware [57, 58]. Consequently, the application of machine learning for prediction and decision making has emerged as a field of research in its own right, enabling highly accurate prediction of values concerning large numbers environmental and process parameters enables the autonomous control and optimization of complex systems [16, 21, 59, 60].

In the manufacturing domain, machine learning has been leveraged in numerous ways in recent years, and as discussed, is a key enabler of realising autonomous CPS's. The *intelligent* analysis of data that supervised machine learning techniques have demonstrated have allowed for the development of advanced predictive systems, utilized to great effect with respect to maintenance problems *[61]*, and productivity and logistics optimisation through detailed knowledge extraction from data and more accurate and dynamic simulated models, environments, and scenarios [23, 55]. More recently,

advances in deep learning and reinforcement learning have further improved on these predictive models, computer vision capabilities, and furthered work towards realising systems capable of autonomous decision making and control [62, 63]. A thorough discussion of the application of machine learning in the manufacturing domain can be found in *[64]*.

Within the field of machine learning techniques, artificial neural networks have emerged as a capable and versatile learning methodology [65]. These algorithms are, with minimal apriori knowledge, able to approximate the complex functions which govern the relationships between multidimensional datasets. Numerous network structures have been developed to successfully utilise numerous data sources: Convolutional networks, have been shown to improve perceptive ability by reducing high dimensional data to extract knowledge, successful applications in image recognition [66]; Recurrent neural networks can accurately track temporal changes and improved pattern recognition over time from time-series data, with demonstrated success in speech and written word comprehension and generation [67, 68].

In line with the identified applications of adaptability and flexibility, neural networks are also able to iteratively learn from experience (as opposed to an existing dataset) in cases of Reinforcement Learning where multidimensional input data captures the environment, and the network approximates the effects of actions taken and the resultant influence on the environment. This ability has proven to be the critical factor in enabling a wide number of recent applications to control and decision-making problems; as the capacity for self-improvement ultimately enables the networks to experience a wider variety of situations, and to genuinely learn interactions as opposed to patterns in the dataset. Networks able to explore and iteratively improve have demonstrated great success when applied to many tasks, including those with high dimensional input data and for advanced manipulation using sensory data [69].

Reinforcement learning, from a mathematical perspective, is often implemented in the form of a Markov-Decision-Process or MDP, which models a decision-making process based on a set of states, actions and rewards. Actions are selected based on a policy, π, which maximises the cumulative sum

of rewards. Deep Q-learning Network's (DQN's), makes use of a multi-layered neural network to approximate the policy in an MDP, by calculating an optimum value function *Q(s, a)*, or 'Q-Score', representative of the quality of an action in a specific state [70, 71]. Such an approach is well suited to enabling adaptability in robotic systems, based on observations of the environment, as it makes use of state observations and observed feedback to determine performance and drive decision-making. This approach termed Q-Learning has proven exceptionally competent at learning complex and non-deterministic systems, due to the ability of neural networks to act as function approximators and to evaluate their performance based on received feedback, which minimizes the need for labelled data, and the need to fully define the environment and its possible states [66, 72, 73]. Other, more involved reinforcement learning algorithms and approaches exist, such as policy gradient algorithms, which stochastically model the probabilities of positive and negative actions as opposed to assigning a direct value. This enables them to act in a continuous action space, and to determine appropriate action policy in more complex environments, at the cost of necessitating on-policy training, and increased training time [74]. Other approaches, such as actor-critic models, seek to combine the benefits of both a value and policy-based approach to reinforcement learning, but again with the increased complexity of computation and the associated disadvantages [75, 76].

Multiple techniques exist to improve DQN performance, including more advanced action-selection policy methods, such as epsilon decay, where the epsilon parameter is used to choose between random and selection action and is decreased over the training duration to encourage initial exploration of the environment. More recently, Google's Deepmind developed a network that outperformed existing examples on several ATARI 2600 games via two key insights. Firstly, the use of training methodologies such as Experience Replay, where a buffer of recent transitions is collected and sampled for training, serving as a form of memory, and secondly, using a separate target network to make predictions for training, enabling the algorithm to converge to a stationary target. This Target network is then updated using the decision-making network after a certain number of training steps [77]. This work by deep mind was also one of the first to consider additional layer architectures, by utilising the strength of convolutional nets to extract abstract representations from images, or in the case of the Atari Work,

from the generated game screen pixels. This ability to create simple representations from high dimensional inputs further highlights the strength of neural networks to effectively learn from complex state information and has since been applied across varied domains, including manufacturing [78]. More recent work has looked to combine aspects of Recurrent neural networks with DQN's, due to their strength in tasks with temporal dependencies [79]. This has particular applicability to manufacturing tasks as many operations are constrained to a sequential task structure, i.e. many operations are detrimental if other actions have not been performed first. Consequently, developing network models capable of understanding these temporal links is crucial to the effective application of DQN Agent control to many manufacturing systems.

The applicability of neural networks to manufacturing control is necessary to consider, as the strengths of the approach lend themselves well to manufacturing control tasks. Such tasks are commonly composed of sets and subsets of instructions, with a temporal order additionally, the identification of sub-tasks through abstraction can improve generalization of a developed approach, as many sub-tasks are likely to be similar across manufacturing operations. There exist already, a significant number of approaches that aim to utilise reinforcement learning, often DQN's, to improve a variety of tasks across the manufacturing domain [80, 81].

2.3. Applications of Human-Robot-Interaction in Manufacturing

As discussed, the extension of decentralized control into the intelligent manufacturing field necessitates the consideration of the interactions of intelligent agents with their human counterparts [82]. In the application focused on in this work, these interactions occur between robots and their human colleagues, either directly or as sequential members of a production process. At the system level, it also necessitates consideration of more traditional elements of *collaborative* robotics, and how multiple agents (human or robotic) may observe and adapt to one another to achieve a common goal. Work in this area is particularly applicable to dealing with the challenge of Human-Robot-Interaction, as humans possess

their own agency, as well as a great deal of variation in their task performance due to a wide variety of external *Human Factors*, most notably fatigue [7, 8, 83], requiring a degree of adaptability to overcome.

Some of the more directly collaborative (that is, featuring humans and robots performing a single combined action, as opposed to coordinating their actions) human-robot tasks have already benefited from the introduction of intelligent control systems; particularly those involving direct, physical collaboration in a shared workspace. Robotic operators are frequently used to improve strength in handling tasks, and the use of learning techniques for improving precision and coordinating motion in such tasks enables large components to be manipulated safely and with relative ease [84]. Similar learning methods have also been used in these tasks to improve *Direct-Teaching* of robotic operators, and enable some complex manufacturing processes such as composite layup and welding fabrication (which often require a level of flexibility to be automated and to achieve a similar finish and quality to human professionals), to be automated [85-88].

Other work in the field has focused on more passive forms of collaboration, such as how knowledge of others in terms of ability and expertise, combined with context, can be sampled from the environment, or observed directly, and used to inform behaviour to achieve or optimise a current shared goal [89-91]. Such techniques have proven useful in techniques such as gesture control which is increasingly used for robotic control in human-machine-interactions [92]. Expanding the application of intelligent manufacturing and learning techniques to the field of human-robot-interaction, by considering the impact of human factors on performance, may enable adaptable behaviour to be expressed in response to changes in human performance.

Despite the successful application of neural network-based learning to improve the autonomy and adaptability of computational systems, and the demonstrated benefits of enabling adaptability to robotic systems which interact and collaborate with human counterparts; there remains limited work in applying reinforcement learning techniques to robotic control. Within the field of intelligent manufacturing, some implementations exist which look at machine learning from the system level. Such applications

inherit several challenges due to the system complexity, whereas the benefits of a decentralized approach are well recognized as able to overcome such issues. The following section presents a methodology for implementing a reinforcement learning algorithm within a cellular robotic operator, to enable it to align its actions with humans which exist within the same process.

3. Methodology

As discussed in the previous section, the utilization of agents with a DQN used for decision-making within a discrete action space is well understood in many domains, and learning Agents continue to be developed and applied to various simulated and virtual tasks. Despite this, little work exists to implement their ability for robotic control, and virtually none on leveraging these abilities to understand and account for human behaviours.

The authors hypothesise that a learning agent equipped with a DQN may be used to provide adaptability to robotic operators via a control algorithm which generates instructions to adjust the operational parameters of the robot. A DQN based approach to this reinforcement learning problem is chosen as the action space of robotic operators is discretized in the form of routines executed by the robotic controller. As such, the advantages of a stochastic approach do not apply in this case, without a different approach to affecting the decided behaviours with the hardware. The interaction dynamic present in most manufacturing operations is typically modelled in terms of speed, via the cycle time of the operator. As discussed, human operators inherently introduce variation and consequent disturbance into many processes leading to an increased idle time of operators and components and consequent decreases in productivity.

The following section presents the authors novel methodology, which may be used to implement a Learning Agent within a simulated manufacturing environment, with the intent of using the agent's capability for improving the interaction dynamics between human and robotic elements. The

methodology takes inspiration from many different approaches and existing frameworks and outlines the necessary considerations and process for appropriately discretizing a manufacturing process and providing the contextual information necessary for the agent to make informed decisions.

The methodology also defines a learning system, which enables an intelligent agent to effectively adjust its operating parameters in response to changes from its environment and collaborators, intending to reduce this performance disparity. This improved adaptability can be leveraged to reduce the influence of disturbances on the process dynamics. The methodology is outlined in terms of information flow, in Figure.1 below:

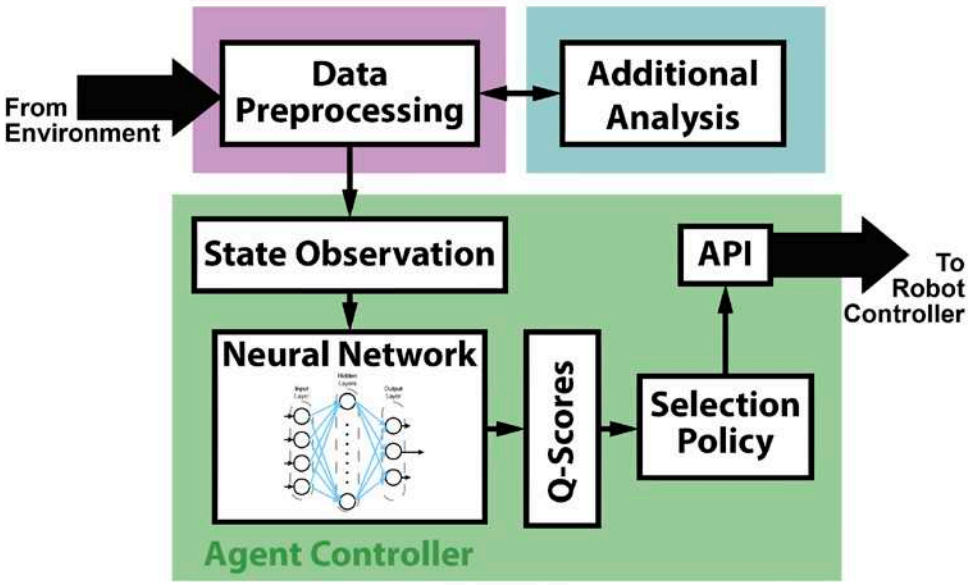

Figure 1. Architecture illustrating information flow through the Agent, and how it interacts with the simulation environment.

Furthermore, the agent must be designed to facilitate effective collaboration and adaptability with other operators, both robotic and human, whilst also maintaining a balance with operational demands. The actions selected by the agent act as calls to subroutines via traditional robotic controllers, which enables

a large variety of different commands to be represented dependent on the application and process. The methodology outlines 5 distinct steps that should be addressed to provide the desired functionality. Consideration must be given to the input observations and the output actions, and how these effectively model the environment and enable the desired control to be realised; The way rewards are defined and presented to the reinforcement agent, to enable effective learning of the interaction dynamic and the effect of its actions; How the agent interacts with the simulation environment to effect changes; how the simulation environment is parameterized, to appropriately model various interaction dynamics, and important to improve validity and generalizability of the approach; And finally, how such an agent may be evaluated concerning its ability, and its application to a real-world problem.

3.1. Defining the Observation & Action Spaces

Within the system, there exist a potentially unlimited number of generated data instances. As discussed in the previous section, observations may take many forms, they may be low-dimensional array-based sources or full pixel images. The data necessary to capture the desired knowledge will vary between a wide number of applications. Consequently, most intelligent control solutions are bespoke, and it is likely this will (to a certain degree of customization and fit) remain the case. As such, a cohesive set of rules for defining which observations to make is crucial to enabling systems capable of learning and adapting based on the influence on performance. These key capabilities can be defined as follows:

- The ability to conceptualize time and temporal patterns, enabled by observing some time-keeping data source. This enables identification of changes from one event to the next, critical for the analysis of changes over time.
- An awareness of the agents own performance, which is necessary to quantify this value to enable a value comparison with the respective rewards, and how it varies from target performance.
- Conversely, awareness of other agents' performance, and how this changes in response to the actions taken.

- Observation of any additional factors that are necessary to model the required dynamics of the system. i.e. anything that may change and influence the system that it should be able to account for.

These abilities allow the intelligent agent to build a sufficiently detailed representation of the manufacturing process, concerning the desired control factors. To reduce performance disparity, the following data points are defined to encompass the necessary information. Combined, these form the observation space: [CT, TCT, CT, IT, CIT, SM, BC]:

- The Agent Cycle Time (CT), defined as the time between completion of each action.
- The Target Cycle Time (TCT), given by:
- Remaining No. Products to Target/Shift duration remaining
- The Collaborator Cycle Time (TCT), multiple of these may be defined for multiple collaborators.
- The Idle Time (IT), which is the consequence of performance disparity and is indicative of the task performance
- The Collaborator Idle Time (CIT), an additional consequence of performance disparity
- The Speed Modifier (SM), which represents the current state of the control parameter.
- The Buffer Contents (BC), the number of products in the transition space, necessary to infer the impact of actions on the collaborator.

Similarly, to the Observation Space of the agent, it's Action Space must also be defined, which describes the available actions that the agent can make to influence its environment. As with the Observation space, the application domain is almost limitless, and there exist no standard frameworks for developing and defining these actions. To develop an agent able to control and adjust operating parameters, such as movement speed, force application, etc. there are several considerations. These actions provide a

large range of functionality and enable a robotic operator to achieve a wide range of tasks; and importantly, the final action, to adjust the operating parameters, enables dynamic adaptability of the control system. In the presented application, this takes the form of movement speed. To enable effective integration with a reinforcement learning model, the range is discretized to the necessary degree of fidelity (i.e. smaller steps enable finer control), and the action space can be defined as:

[Increase Parameter, No Change, Decrease Parameter]

This identified set of actions will enable the agent to control the sequence of operations of the robotic operator and adjust how it performs these operations concerning its parameters. To further facilitate effective learning of the DQN, it is important to provide a measure of maximum and minimum values for each observation during the learning process to enable progressive normalization of the observations over a range 0 to 1. As these values are not known in advance, it is necessary to update the network with a re-normalized memory each time a new minimum or maximum value is encountered during training. This will likely lead to initial instability in training, but ultimately a more effective learning process.

3.2. Defining Reward Structure

Within any reinforcement learning process, defining the reward system for the intelligent agent must be given careful consideration. Attention must be paid to not only the value of the rewards, and the factors that require positive or negative feedback, but also how these rewards are presented to the agent.

A manufacturing scenario is comprised of several discrete states and actions which are governed by certain rules. For example, certain actions must be performed in the order of operation for the assembly or manufacturing task to be a success, whereas certain other actions may not have a clearly defined necessary order. Nonsensical operations, such as attempting to put something down when there is no held object should also be avoided by the design of the reward schema. A degree of reward shaping is applied through the assignment of values of similar magnitude for both positive and negative actions,

avoiding large changes at single steps in learning. There are often cases where a non-ideal action must be taken to reach a state where a much more valuable action can then be performed. These issues are compounded by how these rewards should be presented, in what form, to what scale, and at which intervals. These factors influence agent behaviour by altering the feedback it receives from the environment based on its actions. Current work does not have a consensus or best practise established, and much of how rewards are generated and presented must be developed heuristically alongside the agent itself.

The design of a reward structure in an application such as this, where the agent aims to optimise its performance against a dynamic variable, presents additional difficulties. The agent must achieve a balance between several dynamic values, which also are influenced by the agent's performance. In a typical Reinforcement learning application, agents are provided with fixed rewards which are selected to result in desired behaviours. In this application, the agent may be provided with a fixed reward, R, for each advancement of the manufacturing sequence, which is reduced by a Reward Penalty in proportion to the magnitude of the error between its completed Cycle Time and its operational targets. It may also be influenced by other values, both positively and negatively. In the case considered, the reduction of idle time is a key aim, and this value may be used to penalise the agent for choosing actions which lead to these states. This is illustrated in Equation 1. Furthermore, there exists the potential to prioritize the agent if necessary, by weighting the value of these errors proportional to the impact this has on the reward received, represented by w_1 and w_2 in Equation 2.

$$Reward\ Penalty = (|TCT\text{-}CT|+|CCT\text{-}CT|)\text{-}Idle\ Time \tag{1}$$

$$Reward = R\text{-}(w_1\ |TCT\text{-}CT|+w_2\ |CCT\text{-}CT|)\text{-}Idle\ Time \tag{2}$$

Alternatively, rewards may be scaled for the agent, based on the combined error between the Robotic cycle time, the target performance cycle time, and the cycle time of the collaborator(s), the reward factor is calculated using Equation.3 below:

$$Reward = R/(((|TCT\text{-}CT|+|CCT\text{-}CT|)\text{-}Idle\ Time)) \tag{3}$$

This equation will scale rewards based on how close the agent is towards a balance of its operational targets and amplify the smaller differences in performance when close to the idealised point of operation. These observations which equations 1-3 depend on are made for state, *s'*, and so represent the Cycle Time which the operator has just completed, the calculated Target Cycle Time for all products remaining in the shift, and the predicted value of the Collaborators next Cycle Time. Furthermore, these values are dynamic, presenting issues for effective learning, as optimization towards a moving target has been a significant challenge for reinforcement learning systems, as evidenced by the performance gains Deepmind achieved using their target-fixing approach, which will also be leveraged here.

It was mentioned in the previous section that to effectively deliver rewards to the agent, each parameter update precedes an iteration of the control loop. At each decision-making point, the agent can choose to increase or decrease the value of the controlled parameters, or effectively take no adjustment action and just advance the manufacturing sequence. As the rewards are influenced by the effects on the environment, i.e. by the balance of performance and target/collaborator performance, the manufacturing sequence must be advanced with the parameter selection to avoid a negligible state change and no meaningful reward generation. To be effective, rewards must be delivered in response to actions on either side of an update to the environment.

3.3. Simulation Integration

With the capabilities of the intelligent agent defined to enable the reinforcement network to appropriately observe, respond, and affect the environment, the simulation model can be designed appropriately. Using the Anylogic simulation package [93], a system-dynamics based simulation can be combined with agent-based control to explore agent performance. The package is Java-based, enabling easy integration with the Deep-Learning-4-Java (DL4J library)[58]. The library facilitates the development of a Q-Learning network, that can be integrated effectively with the simulated

environment to provide control decisions.

3.3.1. Agent Interaction

The simulation model contains elements to represent several different operations, which are suitable generalized. Within a manufacturing operation, we can define again several key capabilities that are required to enable the necessary behaviours: Pick up and put down products; Move products from one location to another; Capacity to scrap a product in the event of error; and the ability to manipulate products by following pre-set motions/routines.

The elements within the simulation are consequently chosen to reflect the modelling of these abilities. Queue elements are used to represent individual or groups of products at a given position, which may be inserted or removed as necessary; Delay elements are used to represent operations with a fixed (or potentially variable) time associated with them (i.e. a fixing/glueing operation), and hold the product object for a defined period; additionally, Queue element with a capacity of one is used to represent the robotic gripper, which can move product objects around the simulation positions through variable output gates. These elements can be combined to represent any number of manufacturing operations, and the relevant observation and action spaces defined to enable a corresponding Q-learning model to be built.

Within the simulation model, the robotic operator is controlled by a statechart, which makes the appropriate function calls to the neural network and enacts the result action which the neural network decides upon. This enables the control structure to be designed in response to the available states and actions of the operator for each application and isolates the control structure from the rest of the simulated environment, whilst triggering actions which influence the environment. The statechart takes the generic form illustrated in Figure.2.

When in the Decision-Making state, function calls pass an observation of the environment to the neural network, which produces and returns a set of Q scores for the available actions, the highest-scoring action is selected (Although any policy can be used to make an action selection based on these scores epsilon greedy, Boltzmann etc.). This action is returned to the statechart, which triggers the action within the simulated environment, advances the simulation by the next operation sequence, generates a reward, and returns the full transition to the neural network where it is stored in its memory. The network is then fitted using a randomly sampled batch from its memory, using the target network to make forward predictions. The target network is then updated from the main network every tau training steps. As the received rewards are generated based on the influence of these factors the agent learns over time, to adjust these parameters to adjust its performance, and minimise the disparity between itself, and its collaborators.

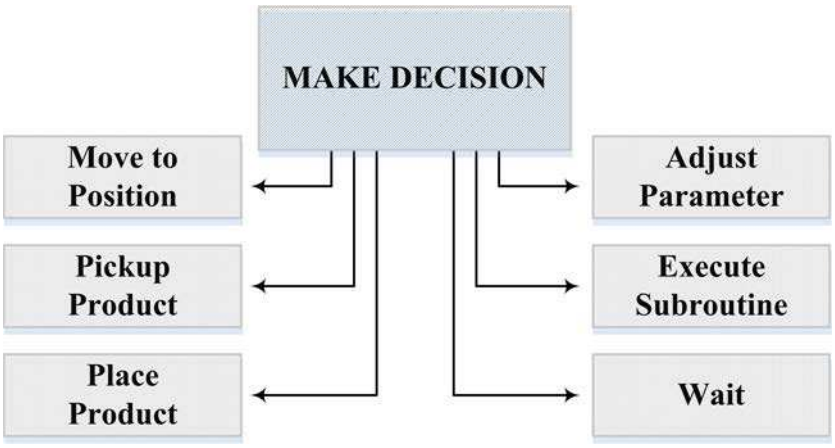

Figure 2. Generic behavioural Statechart format to discretize robotic processes within manufacturing simulation. Multiple similar states may be defined, and each state corresponds to a chosen action.

3.3.2. Environment Parameterization

To effectively develop a learning model capable of minimising variation between robotic and human operators, a representative degree of variation must be represented in the simulation model. This is done by defining objects representing human operators and parameterising their task-performance based on

several identifiable human factors. As discussed previously. there exist many manifestations of fatigue, particularly in cases involving dexterous manipulation of products. The task-duration, time-of-day, and day of the week all influence human performance, in ways which are dependent on the individual. To effectively parameters the environment, profiles are defined for three operators who will form the human element of the process, based on nominal cycle time CT. After each cycle, their performance is evaluated using a sequence of parameters: A shift modifier (SM), which adjusts performance based on the time of day (AM shift, midday shift, PM shift); a Weekday Modifier (WM), similar adjusting performance based on the day of the week; and a Fatigue Modifier (FM), which represents the susceptibility to time-on-task fatigue, and which scaled over the shift duration representing decreasing performance.

Different profiles are defined to provide a wide range of conditions and combinations of susceptibility. Operator 1 was parameterized as an experienced operator, with low initial cycle time, but a susceptibility to time-on-task fatigue; Operator 2 was designed to represent an average case, with a nominal base CT equal to that of the design takt time of the system, and no fatigue influence was included.; and operator 3 was designed to represent a more novice worker with slower cycle time, but a pace which reduces the influence of fatiguing. No fatigue influence was included. The combination of the modifiers enables the calculated cycle time for each human operator at a specific point in time (Monday, AM, in the illustrated example.) to be obtained via equation (4), where n is the specific operator, and the values for each modifier for each defined operator is illustrated in Table.1.

$$Calculated\ CT_{Ann} = CT_n \cdot WM_{an} \cdot SM_n \cdot FM_n \tag{4}$$

Table 1. Breakdown of the values used to modify the performance of each operator.

Operator Number	**1**	**2**	**3**
Base Cycle Time	CT_1	CT_2	CT_3
Fatigue Modifier (End of shift)	FM_1	FM_2	FM_3
Shift Modifier AM	SM_{AM1}	SM_{AM2}	SM_{AM1}
Shift Modifier Midday	SM_{MD1}	SM_{MD2}	SM_{MD1}

Shift Modifier PM	SM_{PM1}	SM_{PM2}	SM_{PM1}
Weekday Modifier Monday	WM_{a1}	WM_{a2}	
Weekday Modifier Tuesday	WM_{b1}	WM_{b2}	
Weekday Modifier Wednesday	WM_{c1}	WM_{c2}	
Weekday Modifier Thursday	WM_{d1}	WM_{d2}	
Weekday Modifier Friday	WM_{e1}	WM_{e2}	

Using these modifiers and the outlined equations, the simulation can generate models of HO performance, to induce different interaction dynamics and evaluate the DQN Agents versatility and ability to generalize over varied conditions. This is done by iterating the day of the week (Monday-Friday), representing each week of production, and then by varying the shift order of the operators each week, to vary the Time of Day each is working (These orders are: 123, 231, 312).

3.3.3 Evaluating Hyperparameters

Q-Learning Agents have demonstrated a great deal of success across several applications in recent years, as discussed in section 2. The key issue with implementing and developing learning algorithms is hyperparameter selection, as these values are often crucial to the learning efficacy and are difficult to determine apriori. The initial stages are to evaluate the performance of these algorithms given different learning parameters to identify potentially viable configurations. This is achieved using a grid search and a pre-defined training and test set.

Neural Networks are dependent on their internal structure, defined by the number of hidden layers and the number of hidden nodes in each of these layers. There are additionally dependent on a learning rate, and the choice of updater function, which describes how errors are propagated through the network.

To evaluate the effect of these hyperparameters on learning, two datasets are constructed containing a set of state, action, reward tuples by selecting random actions over several simulation runs and

observing the received rewards. These datasets form a training and a test set respectively (with a ratio of 10:1), and) and are used to evaluate each hyperparameter configuration. A DQN built and trained using the test set for each of the parameter configurations, which is then evaluated using the isolated test set producing an average error score for each configuration. The *Root-Mean-Squared-Error,* (RMSE) is used as the error score in this case, and for N samples, is given by:

$$RMSE = \sqrt{\frac{\sum(y_{pred} - y_{actual})^2}{N}} \tag{5}$$

This error is indicative of the networks predictive accuracy in terms of assigning the correct Q-Value to each action, and this method enables fairly high confidence in the hyperparameters to be achieved in advance of implementation.

In addition to the parameters associated directly with building an effective neural network implementation, there are other parameters which influence the integration and learning over time. In a Q-learning approach, consideration must be given to gamma, the discount factor, which influences the effect of future rewards; and Epsilon, which is used to implement Epsilon Annealing; enabling an agent to fully explore the state space. Defined through an initial maximum value which tends towards a minimum based on a specified decay rate, annealing to fast or to slowly can influence training performance. Both of these parameters should be evaluated where possible although typical values are gamma = 0.7 and epsilon decay = 0.99.

The hypothesis of this work, based on the reviewed literature, and the application of similar learning techniques to similarly discretizable problems, is that an agent equipped to learn to utilise some form of Q-Learning Agent will be able to achieve an optimal solution based on its performance constraints; which may be utilised for robotic control through the selection of appropriate action. The simulation is intended to be suitably generalized, although as the design of such systems is application-specific, such a generalization is not entirely possible. Furthermore, once the simulation environment is defined, the approach provides a training platform for the agents, which is non-invasive of the real-world process;

alongside the capacity to integrate agents within customizable and definable processes. These are both strengths of the approach, enabling variations and improvements to be evaluated and an optimal solution developed with ease and with minimal disruption.

3.4. Evaluating Methodology Performance

Evaluation of the proposed methodology includes several considerations, as multiple elements must be verified. Most of the constituent elements are validated as part of their development, but a method for establishing benchmark performance and enabling comparisons to be drawn must also be defined. As mentioned in section 3.2, the key metric to validate the approach is performance disparity. Minimising the performance disparity is the key goal of the agent for several reasons: It reduces the impact of human variation on the process by minimising the development of operator idle-time, and the build-up of product between processes. This can be said to improve the interactions in terms of fluency, and has real benefits for manufacturing systems, by moving closer to an idealised one-piece-flow.

This must be done, however, whilst maintaining process aims and constraints, including appropriate productivity. Balancing these contrasting demands is a key capability of the agent and must also be evaluated. The key metrics can, therefore, be said to be the cumulative idle time observed in the system (the product of performance disparity), and the number of products produced.

To evaluate the efficacy of the approach, a static case will be defined, utilising a separate agent which cannot adjust its parameters and executes the operation sequence as a traditional robotic operator, and the resultant idle times and productivity used to benchmark the existing dynamics. The defined intelligent agent will then be implemented in the static agents' place, and the simulation re-run to provide a comparison. The intelligent agent should be able to reduce the observed cumulative idle time, whilst maintaining an acceptable level of production. To further determine the generalizability and validity of the approach, several different simulation environments will be explored, and the process

repeated to ensure that the agent can achieve an optimised level of performance in multiple cases. This is achieved partially by adjusting the simulation parameters as discussed to represent different conditions and interacting behavioural changes in the operators and will partially be addressed by varying the task and production demands. Furthermore, the simulation environment explored in this work is based on, and representative of, an existing real-world problem and process; with improvement over a static control case lending further validity to the approach.

4. Case Study

This section presents an initial case-study to demonstrate the simulation and learning methodologies and explore how they can be practically applied to a manufacturing process simulation. Within the observed process, performance disparity between robotic and human operations in terms of the duration of each operation is a source of disturbance to the overall process. This disparity is the result of changing behaviours in human individuals and is an inherent part of such interactions. The capacity for robotic operators to adapt their behavioural policy as a result of these changes can reduce this disparity with benefits at the interaction and process level. Consequently, it is hypothesized that the agent responsible for controlling the operation sequence will be able to, via reinforcement learning, be able to identify ideal actions for each of the states of the process and converge to an optimal policy capable of matching an identified target speed.

4.1. Manufacturing Process Modelling

Following the methodology presented in the previous section, a case-study scenario was developed using the simulation package to demonstrate the feasibility of the approach, which is based on the semi-automated production line of an industrial partner. The production line is responsible for the manufacture of single-use medical suction devices and features numerous automated and manual operations. The interaction between the first and second stages of the process provides an ideal case

study, The initial manufacturing cell 1, is a robotic process which contains a combination assembly and glueing processes, and a curing process, before the product is passed to the second cell, which contains a manual assembly step. Furthermore, this manual assembly step is a good example of both physical and cognitive loading, as it involves both dextrous manipulation of objects and visual verifications and decision-making. The two cells are separated by a discrete buffer that holds 10 products, which, when either empty or full, leads to observable idle time in each operator. The manufacturing process is illustrated in Figure.3.

The manufacturing process was monitored, and timings made and averaged to develop nominal times for each operation in the process, and the overall nominal cycle time; alongside human performance in the simulation model. The parameters for human performance were defined as in Table.2. based on a combination of nominal process observations used to establish a baseline cycle time, and work examined in the existing literature regarding the impact of the discussed human factors, these values range approximately ±10s from the nominal 50s of cell1 operation.

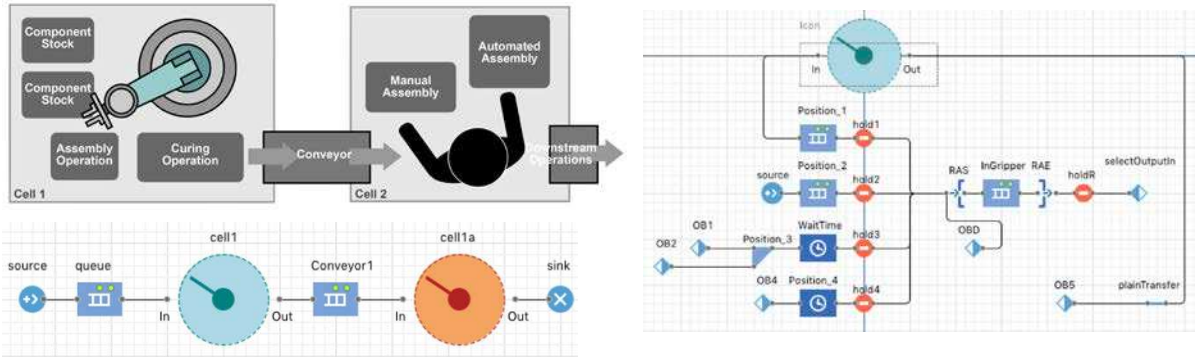

Figure 3. Tripolar production line at OSTE, illustrating the interaction between the robotic cell and human-operated cell, and the simulation model for the robotic operator developed in Anylogic.

The operations performed by cell 1 in the process may be broken down into four key operations, two instances of retrieving product stock (one from an upstream process, one from a stockpile of components), a task that joins these two components, and another task with an associated processing

time or time delay. These are represented by the elements down the left side and the block to the right acts as the robot's manipulator in Figure.3.

Table 2. Values used to profile performance for each of the three operators based on observed case-study

Operator Number	**1**	**2**	**3**
Base Cycle Time	40	45	50
Fatigue Modifier (End of shift)	1.2	1	1.1
Shift Modifier AM	1.1	1	1.1
Shift Modifier Midday	1	1	1
Shift Modifier PM	0.9	1	0.9
Weekday Modifier Monday	1	1.1	
Weekday Modifier Tuesday	1	1.05	
Weekday Modifier Wednesday	1	1	
Weekday Modifier Thursday	1	0.95	
Weekday Modifier Friday	1	1	

The action space of the agent is defined as detailed in the methodology, with actions identified to adjust the identified parameters, in this instance, the speed of operation (within acceptable limits) used to correct the overall cycle time to reduce the discrepancy between upstream or downstream operations. The resulting control logic of the agent is managed using the statechart illustrated in Figure.4, which makes the appropriate function calls depending on the current state of the operator. Again, following the presented methodology, the observation space of the agent is defined for the simulation environment. The state observation is provided as a 1D array of vector values representing the data points or input values, that are necessary to provide the policy with appropriate decision-making

information. Such information will vary between applications, but in the presented case study contains 6 main variables: the Robotic Operator's previous Cycle Time (CT); The calculated Target Cycle Time to reach the production target (TCT); The cycle time(s) of the current collaborators (upstream or downstream operators, each will have their own input variable) Collaborative Cycle Time (CCT); the current parameter modifier, the Speed Factor (SF) in this application; the current buffer contents (BC); and the Idle Time (IT) for the last cycle.

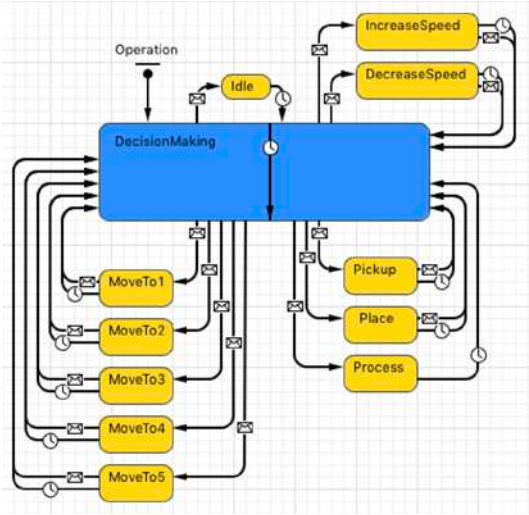

Figure 4. Statechart used within Anylogic to determine robotic operator behaviour, each yellow state is an action

4.2. Agent Training

Initial hyperparameters were identified following the search approach detailed in the methodology. Figure.5 below presents the grid search results for the DQN. Both approaches were evaluated on different dataset sizes, of 1000 and 10,000 instances. The networks were constructed using RELU activation functions and Adam updater. The networks were trained using stochastic-gradient-descent. Anomalous results were identified and not included in the colour-mapping, to provide a clearer indication of performance variation.

From the graphs illustrated, ideal hyperparameters can be selected based on the combinations which result in the lowest overall RMSE. The concentration of blue value at the top of the graphs in Figure.5 suggests that the most accurate estimations can be achieved using two hidden layers and 10 hidden nodes per layer, achieving the highest and stable accuracy for multiple learning rates. However, Figure.5 also suggests that a very simple, single hidden layer 3-node network is also highly accurate in terms of predicting these values, however, this configuration is likely to suffer from overfitting of the model, as a result of the low number of hidden nodes in relation to the number of input variables. As the heatmap shows that minimal gains come from significantly increasing the number of hidden nodes, a 5-hidden-node network as also selected to evaluate a simple model's performance, as this is less susceptible to overfitting in relation to the number of input variables. The influence of the learning rate on predictive accuracy is present, but negligible in comparison to the influence of the network structure. A Learning rate of 0.0025 will be used as lower rates tend to offer the best performance overall.

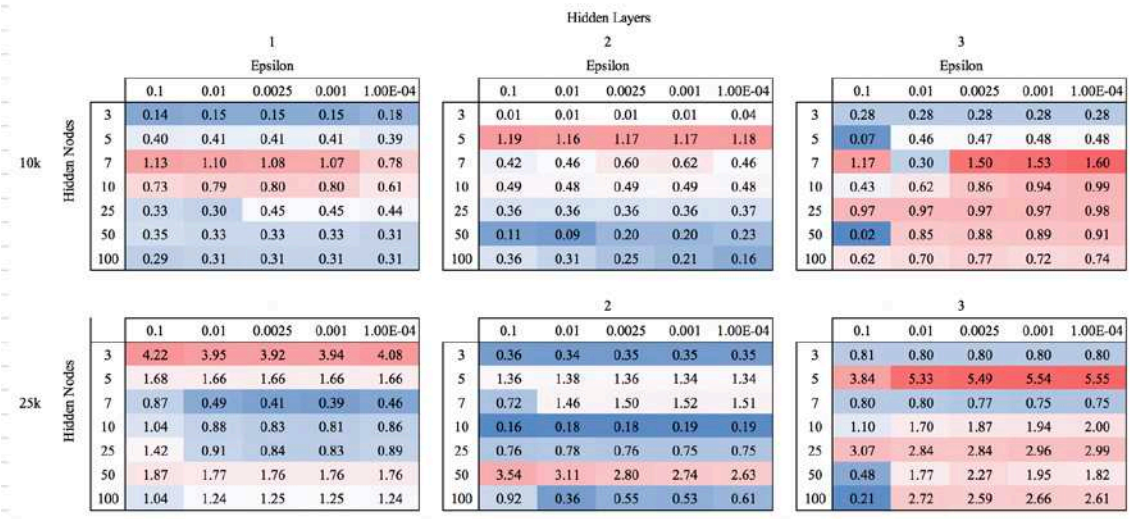

Figure 5. Hyperparameter grid search results for DQN Agent using: a)10k training instances, and b)25k training instances. RMSE scores represented by the colour (blue lowest).

Both datasets generate results of the same order of magnitude in term of accuracy for both training set sizes, with the 25k instance training set being slightly worse. This is potentially due to the greater

variance within the examples. This indicates that smaller training batches may be advantageous in terms of training time with limited impact on predictive accuracy.

Training episodes were set to represent a full day of real-time operation, to enable the agent to explore as much of the state space as it was likely to encounter. These days further represented each weekday, and three shift patterns were used to provide examples of operator performance at different times of the day. Epsilon annealing was utilised as discussed in the methodology. The following sub-sections present the results of the heuristic development process for the agent implementation, and the series of simulations run representing the changing temporal conditions for the specified number of training iterations.

5. Simulation Results

The following section presents the results of the heuristic development process and outlines the application of the methodology to a generalized manufacturing interaction. The agents were generated and trained as outlined in the methodology, and a series of simulations run representing the changing temporal conditions for the specified number of training iterations. The first section discusses the training performance of the DQN Agents alongside the static behaviour case and the relevant metrics of evaluation. The second section compares the effect of the gamma parameter determining the weights of future rewards, alongside different reward structures and their effect on Agent performance.

5.1. Evaluation of Algorithm Performance

This section explores the impact of the DQN approach to generate an action policy and compares the effect of this to the static control case. As discussed in the methodology, the simulation environment is parameterised to represent three different human operators and to investigate the variation of performance of several factors over time. As such, the static policy performs better under some

conditions than others, as a result of the natural variation in how these factors influence the system in combination. As discussed at various points throughout this work, the key aim is to reduce performance disparity between robotic and human operators based on parameter selection. Consequently, to evaluate the efficacy of these approaches in terms of providing benefit to the system, the cumulative Idle Time and the number of products produced are the best indicators of performance. These values are recorded for the Static case and used as a benchmark for DQN Agent performance.

Using the combination of hyperparameters identified from Figure.6 as providing the best predictive accuracy, two DQN agents were trained in the simulation environment over 500 iterations. The agent's scores, productivity, and total idle time were recorded for each iteration, and are illustrated in Figure.6. These DQN Agents were then evaluated over another full simulation run of 15 iterations without further training, to evaluate performance in terms of the system. The values of idle time, DQN score and productivity can be seen, alongside those for the benchmark case in Figure.7. These figures show that both configurations converge to a solution, and that performance after training and compared to the static behaviour case is improved in terms of decrease idle time and maintained levels of production.

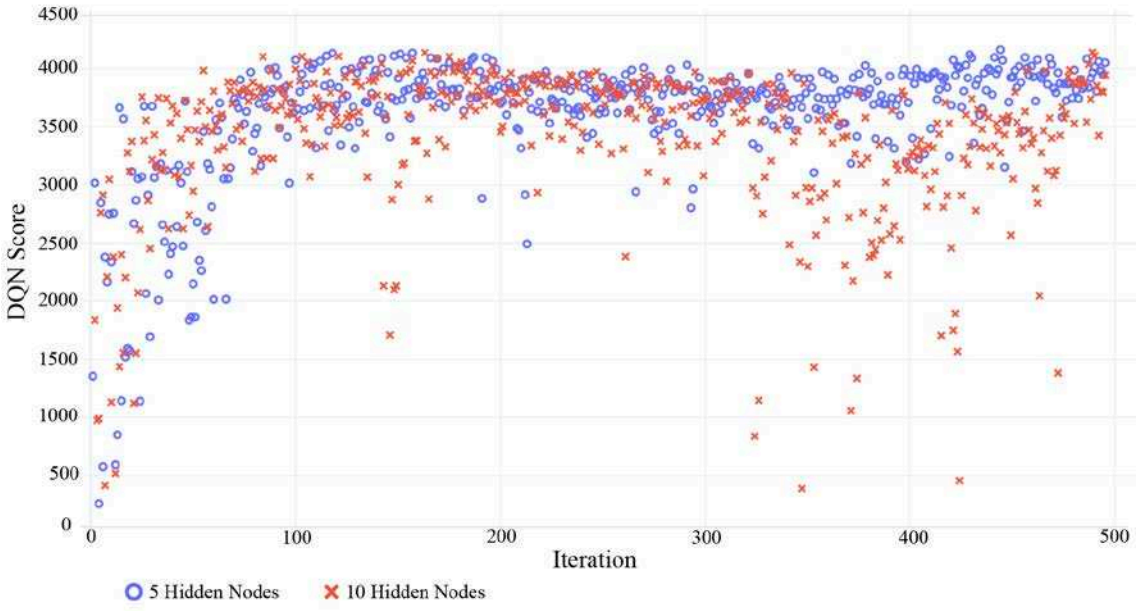

Figure 6. Results of DQN Agent training over 500 iterations for both the 5 Hidden Node and 10 Hidden node configurations.

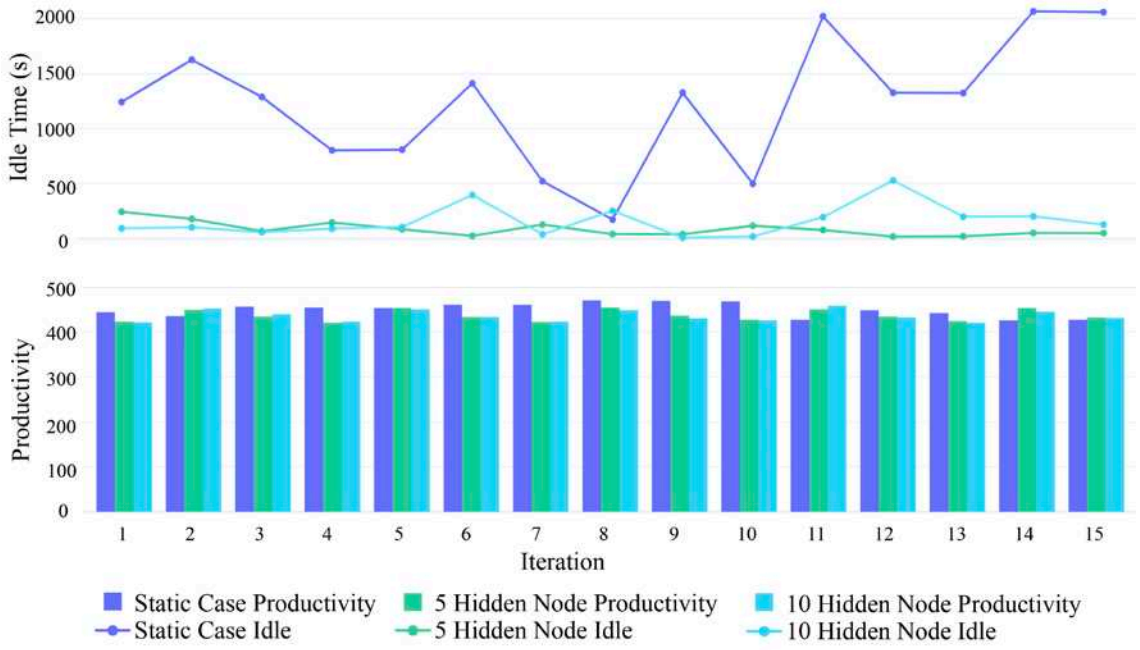

Figure 7. Comparison of DQN Agents to benchmark case in terms of Idle Time, DQN score and Product Count.

5.2. Comparison of Reward Structures

After evaluating the initial parameter selections, the influence of both the selected reward structure and effect of the gamma parameter is evaluated. The same methodology as before is repeated using the single-layer 5 hidden node configuration (selected for its simplicity and overall training stability), for

each of the reward structures outlined in the methodology. A base value of 10 is used for R. Each of these evaluations is repeated for gamma values of 0.1, 0.7, and 0.9. Further to this, both the penalty (Equation (1)) and the scaled reward structure (2) are evaluated for similar agents to explore how this may influence learning ability and policy generation, as described in Section 3.3). Figure. 8 illustrates the training of these agents for different gamma values for both reward structures in terms of the score achieved by the agent.

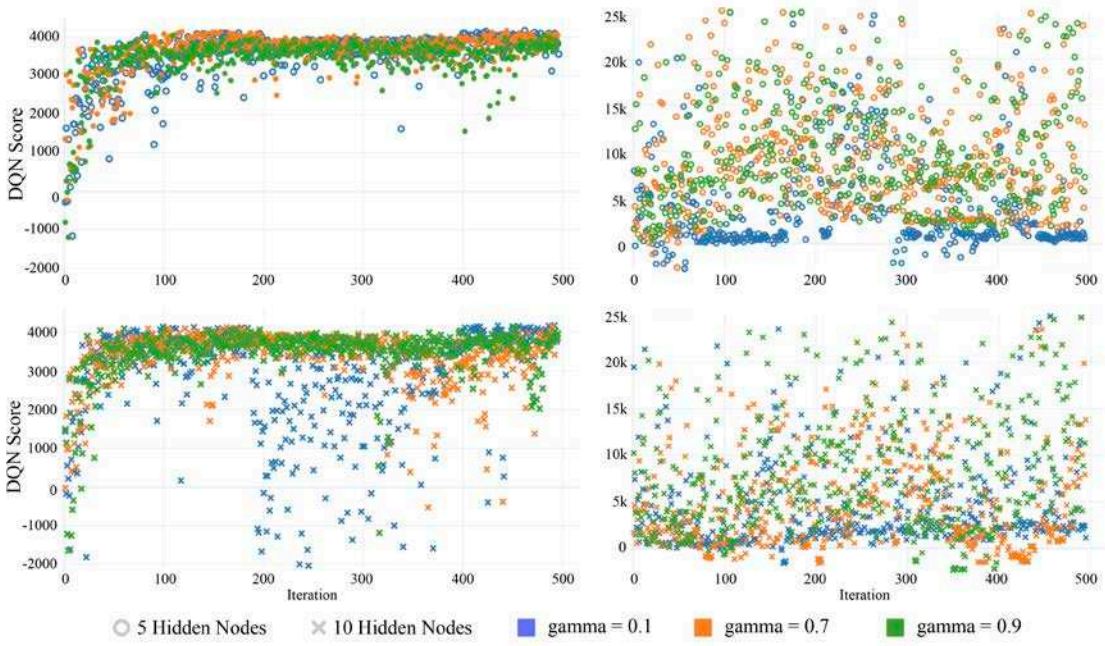

Figure 8. Comparison of DQN structures i)5 Hidden Nodes ii) 10 Hidden Nodes, and training in terms of DQN score for different gamma values and reward structures: a) Penalty rewards, and b) Scaled rewards.

Figure.8. shows that the gamma parameter has a negligible effect on learning or training performance for this application using the penalty reward policy. Conversely, the scaled reward policy can be seen to be much more sensitive to the gamma value, with only the 0.1 network (prioritising immediate reward and usually associated with poor generalizability) showing any indication of convergence to a consistent policy. Moving forward, the single-Layer, 5 Hidden Node configuration will be used, following the

penalty rewards policy, and using a gamma value of 0.7 will be chosen to balance the demands of current and future rewards.

5.3. Evaluating Performance in multiple scenarios

With the optimal hyperparameters and reward structure established, the fully developed agent can be comprehensively evaluated over multiple scenarios to determine its generalized knowledge and performance. As discussed, in the methodology, human task performance is dependent on multiple human factors which affect fatigue and introduce variation. The simulation is parameterized to represent the influence of these factors, and the intelligent agent is evaluated across the range of different scenarios. The benchmark static-case agent is run in the simulation initially and used to create a comparison case. Figure.9 illustrates both agents' performance with regards to the differing variation introduced by the human colleagues, in terms of both the time of day (shift order) and day of the week respectively and compares this performance to the static behaviour case.

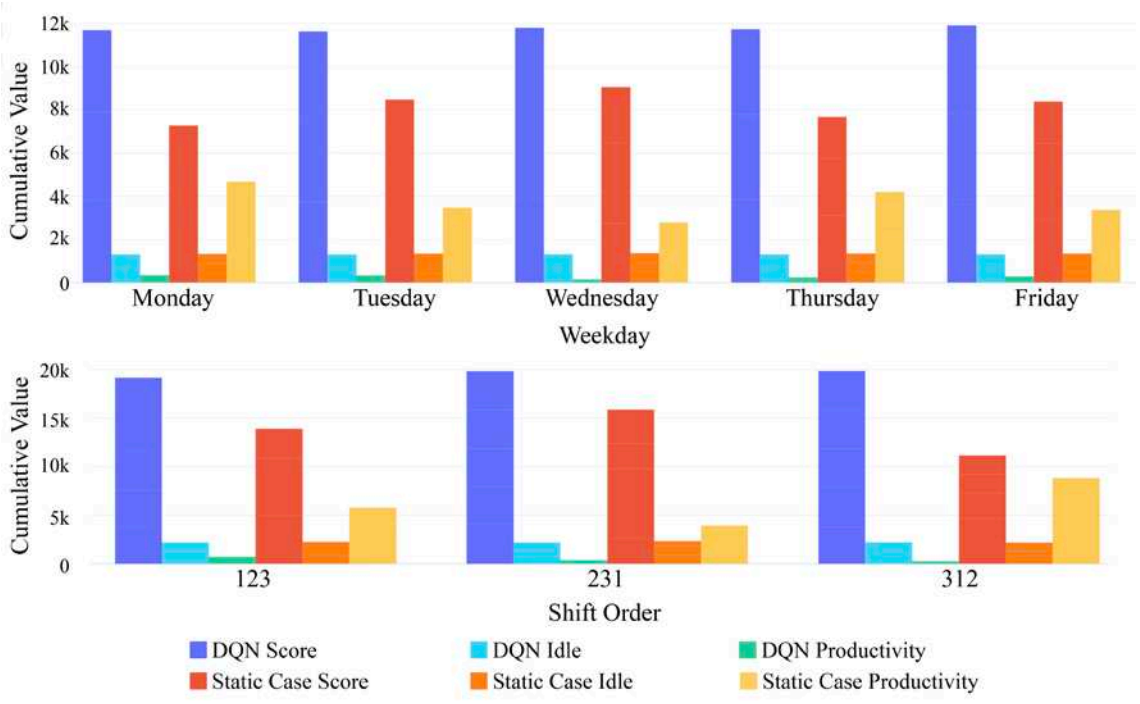

Figure 9. Breakdown of DQN performance over a 3-week simulation run incorporating different shift orders.

The agent's performance is also evaluated in terms of multiple constraints, as already evidenced in the previous sections, where the agent was shown to find an optimal solution prioritising minimising idle time, Figure.10 illustrates the agent's performance when tasked with an increased workload, the target products is increased to 80 per hour or a nominal cycle time of 45s. This is important, as it demonstrates that the agent is capable of generalizing its policy dependent on varied demands; providing a policy that balances both demands without one becoming a priority, and consequently improving the reliability of the approach in different applications and scenarios.

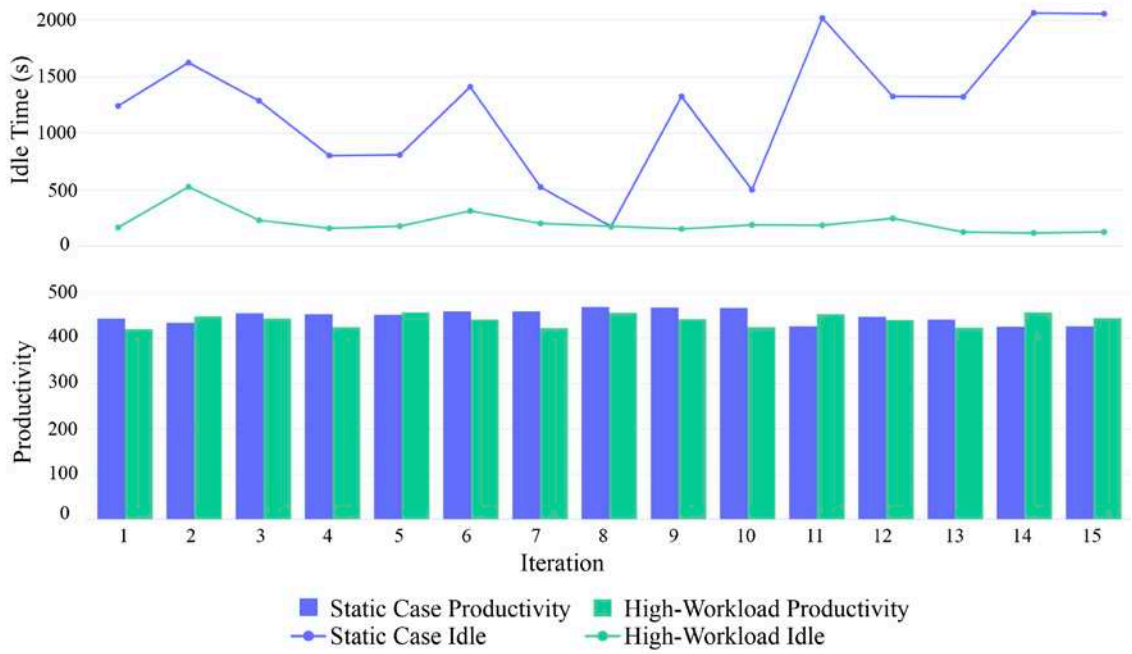

Figure 10. Comparison of DQN to benchmark for increased production demands.

6. Discussion

The results presented in the preceding section validate the author's hypothesis that a software agent made intelligent by the use of machine learning techniques, may be effectively used to improve robotic adaptability to minimise the impact of human performance variation. The presented results support the author's proposed methodology and demonstrate that the outlined approach may be used to effectively reduce performance disparity between a robotic operator and its human colleagues, over a varied set of scenarios.

The results support the observations from the literature suggesting that a reinforcement learning approach has an application in terms of improving robotic adaptability and that such an application may be further exploited to improve interaction dynamics between robots and their human counterparts. In contrast to the existing documented cases, where the focus is on directly aiding or learning from human motion, the authors' approach proposes a novel application of adaptability that focuses on accounting for performance variation in terms of factors which influence human repeatability; ultimately improving the scope for the use of *Intelligence*, to provide benefits to manufacturing systems. Adaptability to change behaviours and actions to different demands and contexts is a key identifier in the CPS architecture and has significant implications for improving the manufacturing process as a whole.

The results support the author's hypothesis that such adaptability, leads to reduced idle times whilst maintaining production, which can be said to improve productivity and bring production in line with idealised one-piece-flow objectives set by the case company. Furthermore, the application of the approach to a real-world case-study outlines its validity and demonstrates how tangible benefits may be realised.

Figure 6 illustrates that both DQN Agent configurations can effectively learn a policy to maximise their scores within the simulated process and that surprisingly, the more stable learning performance can be seen in the less complex network. The scores are initially random due to the initially high epsilon value, but the scores can be seen to converge over time to a consistent value; although this could be further refined through training for additional iterations, as there is still some variance in the DQN scores.

Additionally, as with the static case, the difference in variation between different conditions may also prevent the Agent from achieving a perfectly repeatable policy; i.e. in some conditions, the optimal policy may result in lower performance, simply due to the specific combination of factors.

Figure.7. shows that the DQN Agents are capable of learning and applying an appropriate action policy, to match the level of production and generate a decrease in observed idle time compared to the static case. This is the result of relatively few training iterations and demonstrates the effectiveness of DQN's being employed for robotic control in such a manner. What is not clear from the illustrated figures, is that in the static case, the observed idle time is largely that of the robotic operator itself, with the associated costs. When adaptable control is implemented, the cumulative value of observed idle time is moderately decreased, but it is the human collaborator that is idle, providing cost savings alongside softer benefits such as lower stress levels and a better working environment.

The evaluation of different reward structures illustrated in Figure.8 shows a disparity in performance across the two evaluated reward structures, most likely as a result of the structure design. Rewards for most states will be similar in terms of value, as there is limited potential for one state to lead to a much more valuable future state under the penalty policy, whereas the opposite is true for the scaled policy, where the closer performance is to optimal, the greater these potential differences become. The stability of rewards is also likely the reason why the gamma parameter has a lesser influence on the penalty reward structure. This evaluation provides good evidence for use of a fixed reward, penalty-based policy for generating rewards for an agent in this application, resulting in a generally fast and robust training process.

Figure.9 illustrates that the Agent can achieve a consistent policy across a number of scenarios and with interacting dynamics. This consistency of performance supports the validity robustness of the approach, along with its generalizability to different applications. This is further supported by the multiple potential network configurations and the general resilience of the agent to adjustments to its parameters. This is additionally supported by Figure.10, which further demonstrates the agent's ability to generalize

to an optimal solution that represents a balance of the demands placed on it. It may also be indicative of a robust learning policy in the algorithm and is likely a combination of both these factors. Furthermore, the best performing network architecture was, by contemporary standards, a simple model, which suggests that the approach may be greatly expanded in terms of scope and capability by incorporating more advanced network architectures moving forwards.

Besides, the software developed during this research, when utilised with the appropriate software package, forms an easy to implement the tool with which to experiment with reinforcement agents in a customisable simulation package. When utilised with an appropriate simulation package, this enables advanced control systems to be evaluated in an isolated environment, facilitating both development and implementation of the approach. This furthers the scope for the application of reinforcement learning technologies into manufacturing systems and provides a foundation for others to build on this work, which is anticipated and hoped for by the authors, to further expand the merits of this approach. The primary limitation of the presented framework and reinforcement learning implementation is its relative simplicity to other more current demonstrations, and the extension of the learning elements proposed to make use of more advanced algorithms may enable more advanced and effective policies to be developed.

7. Conclusion

To conclude, the work presented is intended to illustrate the applicability of reinforcement learning to the problem of robotic control within manufacturing. Presented within the focus of utilizing intelligence within data processing, the results indicate that there is potential for distributed agents to improve the adaptability of robotic systems, specifically in cases where there is significant variation introduced by human operators.

The work extends the theory in this area, by providing a novel method to develop and implement a reinforcement learning-based intelligent agent and demonstrating the efficacy of the approach. The

presented results illustrate that an intelligent agent utilising a relatively basic learning algorithm is capable of building a model representing an environment and selecting an action policy to optimize its performance; and that such an agent's policy may be used within these interactions, to provide robotic operators with enough adaptability to minimise the performance disparity and reduce the idle time observed whilst maintaining the appropriate level of production. This has tangible benefits to manufacturing systems, in terms of facilitating lean operation. The work also demonstrates that effective simulation can enable reinforcement learning agents to be developed and trained in a controlled environment as opposed to an online and real-world setting, potentially speeding this process whilst minimizing risk, and provides a robust methodology to do so.

Concerning the future direction of this work, the modular nature of the proposed implementation intentionally provides scope for more advanced algorithms to be implemented for data analysis, and more capable agents to be developed, and more detailed representations built, to further leverage the potential benefits of the approach. It is also suggested that more work is warranted into understanding more passive modes of interactions between robotic operators and their human collaborators is also warranted, across a number of different human-robot-interaction scenarios.

References

1. Xu, L.D., E.L. Xu, and L. Li, *Industry 4.0: state of the art and future trends.* International Journal of Production Research, 2018. **56**(8): p. 2941-2962.
2. Schönsleben, P., F. Fantana, and A. Duchi, *What benefits do initiatives such as Industry 4.0 offer for production locations in high-wage countries?*, in *CIRP 50th Conference on Manufacturing Systems*. 2017, Elsevier: Taichung, Taiwan.
3. Lee, J., H.-A. Kao, and S. Yang, *Service innovation and smart analytics for industry 4.0 and big data environment.* Procedia Cirp, 2014. **16**: p. 3-8.
4. Christensen, H.I., et al., *Next Generation Robotics.* arXiv preprint arXiv:1606.09205, 2016.
5. Esmaeilian, B., S. Behdad, and B. Wang, *The evolution and future of manufacturing: A review.* Journal of Manufacturing Systems, 2016. **39**: p. 79-100.
6. Khalid, A., et al., *A methodology to develop collaborative robotic cyber physical systems for production environments.* Logistics Research, 2016. **9**(1): p. 23.

7. Lorist, M.M., et al., *Mental fatigue and task control: planning and preparation.* Psychophysiology, 2000. **37**(5): p. 614-625.
8. Lorist, M.M., et al., *Motor fatigue and cognitive task performance in humans.* The Journal of physiology, 2002. **545**(1): p. 313-319.
9. Blake, M., *Time of day effects on performance in a range of tasks.* Psychonomic science, 1967. **9**(6): p. 349-350.
10. Testu, F. and R. Clarisse, *Time-of-day and day-of-week effects on mnemonic performance.* Chronobiology international, 1999. **16**(4): p. 491-503.
11. Lee, J., B. Bagheri, and H.-A. Kao, *A cyber-physical systems architecture for industry 4.0-based manufacturing systems.* Manufacturing Letters, 2015. **3**: p. 18-23.
12. Hoffman, G. *Evaluating fluency in human-robot collaboration*. in *International conference on human-robot interaction (HRI), workshop on human robot collaboration*. 2013.
13. Sundar, R., A. Balaji, and R.S. Kumar, *A review on lean manufacturing implementation techniques.* Procedia Engineering, 2014. **97**: p. 1875-1885.
14. Soliman, M. and T.A. Saurin, *Lean production in complex socio-technical systems: a systematic literature review.* Journal of Manufacturing Systems, 2017. **45**: p. 135-148.
15. Sutton, R.S. and A.G. Barto, *Reinforcement learning: An introduction*. 2018: MIT press.
16. Mnih, V., et al., *Human-level control through deep reinforcement learning.* Nature, 2015. **518**(7540): p. 529-533.
17. Mnih, V., et al. *Asynchronous methods for deep reinforcement learning*. in *International conference on machine learning*. 2016.
18. Zhong, R.Y., et al., *Big data analytics for physical internet-based intelligent manufacturing shop floors.* International journal of production research, 2017. **55**(9): p. 2610-2621.
19. Li, B.-h., et al., *Applications of artificial intelligence in intelligent manufacturing: a review.* Frontiers of Information Technology & Electronic Engineering, 2017. **18**(1): p. 86-96.
20. Rüßmann, M., et al., *Industry 4.0: The future of productivity and growth in manufacturing industries.* Boston Consulting Group, 2015: p. 14.
21. Lee, J., B. Bagheri, and C. Jin, *Introduction to cyber manufacturing.* Manufacturing Letters, 2016. **8**: p. 11-15.
22. Russom, P., *Big data analytics.* TDWI best practices report, fourth quarter, 2011: p. 1-35.
23. Kritzinger, W., et al., *Digital Twin in manufacturing: A categorical literature review and classification.* IFAC-PapersOnLine, 2018. **51**(11): p. 1016-1022.
24. Ding, K., et al., *Defining a digital twin-based cyber-physical production system for autonomous manufacturing in smart shop floors.* International Journal of Production Research, 2019. **57**(20): p. 6315-6334.
25. Liu, Q., et al., *Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system.* Journal of Manufacturing Systems, 2020.
26. Lee, E.A. *Cyber physical systems: Design challenges*. in *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*. 2008. IEEE.
27. Wang, L., M. Törngren, and M. Onori, *Current status and advancement of cyber-physical systems in manufacturing.* Journal of Manufacturing Systems, 2015. **37**(Part 2): p. 517-527.

28. Schuh, G., et al., *Collaboration Mechanisms to increase Productivity in the Context of Industrie 4.0.* Procedia CIRP, 2014. **19**: p. 51-56.
29. Monostori, L. and B. Kádár. *Agent-based control of manufacturing systems*. in *Intelligent Processing and Manufacturing of Materials, 1999. IPMM'99. Proceedings of the Second International Conference on*. 1999. IEEE.
30. Permin, E., et al., *Self-optimizing production systems.* Procedia CIRP, 2016. **41**: p. 417-422.
31. Leitao, P., et al., *Smart Agents in Industrial Cyber–Physical Systems.* Proceedings of the IEEE, 2016. **104**(5): p. 1086-1101.
32. Cao, Y., et al., *An overview of recent progress in the study of distributed multi-agent coordination.* IEEE Transactions on Industrial informatics, 2013. **9**(1): p. 427-438.
33. Leitão, P. and F. Restivo, *ADACOR: A holonic architecture for agile and adaptive manufacturing control.* Computers in industry, 2006. **57**(2): p. 121-130.
34. Van Brussel, H., et al., *Reference architecture for holonic manufacturing systems: PROSA.* Computers in industry, 1998. **37**(3): p. 255-274.
35. Leitao, P., *A holonic disturbance management architecture for flexible manufacturing systems.* International Journal of Production Research, 2011. **49**(5): p. 1269-1284.
36. Young, J.E., et al., *Evaluating human-robot interaction.* International Journal of Social Robotics, 2011. **3**(1): p. 53-67.
37. Park, H.-S. and N.-H. Tran, *An autonomous manufacturing system based on swarm of cognitive agents.* Journal of Manufacturing Systems, 2012. **31**(3): p. 337-348.
38. Huang, Z., et al., *Industry 4.0: Development of a multi-agent system for dynamic value stream mapping in SMEs.* Journal of Manufacturing Systems, 2019. **52**: p. 1-12.
39. Bochmann, L., et al., *Human-robot collaboration in decentralized manufacturing systems: An approach for simulation-based evaluation of future intelligent production.* Procedia CRIP, 2017. **62**: p. 624-629.
40. Marik, V. and D. McFarlane, *Industrial adoption of agent-based technologies.* IEEE Intelligent Systems, 2005. **20**(1): p. 27-35.
41. Laird, J.E. and R.E. Wray III. *Cognitive architecture requirements for achieving AGI.* in *Proceedings of the Third Conference on Artificial General Intelligence*. 2010.
42. Kotseruba, I., O.J.A. Gonzalez, and J.K. Tsotsos, *A review of 40 years of cognitive architecture research: Focus on perception, attention, learning and applications.* arXiv preprint arXiv:1610.08602, 2016: p. 1-74.
43. Salvucci, D., E. Boer, and A. Liu, *Toward an integrated model of driver behavior in cognitive architecture.* Transportation Research Record: Journal of the Transportation Research Board, 2001(1779): p. 9-16.
44. Evans, J.S.B. and K.E. Stanovich, *Dual-process theories of higher cognition: Advancing the debate.* Perspectives on psychological science, 2013. **8**(3): p. 223-241.
45. Wiltshire, T.J., et al. *Leveraging social judgment theory to examine the relationship between social cues and signals in human-robot interactions*. in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. 2014. SAGE Publications Sage CA: Los Angeles, CA.
46. Wiltshire, T.J., et al., *Enabling robotic social intelligence by engineering human social-cognitive mechanisms.* Cognitive Systems Research, 2016.
47. Anderson, J.R., *ACT: A simple theory of complex cognition.* American Psychologist, 1996. **51**(4): p. 355.
48. Laird, J.E., A. Newell, and P.S. Rosenbloom, *Soar: An architecture for general intelligence.* Artificial intelligence, 1987. **33**(1): p. 1-64.

49. Isla, D., et al. *A layered brain architecture for synthetic creatures*. in *International Joint Conference on Artificial Intelligence*. 2001. LAWRENCE ERLBAUM ASSOCIATES LTD.
50. Ali, S., et al., *Distributed control systems security for CPS*, in *Cyber Security for Cyber Physical Systems*. 2018, Springer. p. 141-160.
51. Bannour, F., S. Souihi, and A. Mellouk, *Distributed SDN control: Survey, taxonomy, and challenges.* IEEE Communications Surveys & Tutorials, 2018. **20**(1): p. 333-354.
52. Monostori, L., *Cyber-physical production systems: Roots, expectations and R&D challenges.* Procedia CIRP, 2014. **17**: p. 9-13.
53. Monostori, L., et al., *Cyber-physical systems in manufacturing.* CIRP Annals-Manufacturing Technology, 2016. **65**(2): p. 621-641.
54. Shi, D., et al., *Intelligent scheduling of discrete automated production line via deep reinforcement learning.* International Journal of Production Research, 2020: p. 1-19.
55. Wang, J., et al., *Deep learning for smart manufacturing: Methods and applications.* Journal of Manufacturing Systems, 2018. **48**: p. 144-156.
56. Sharp, M., R. Ak, and T. Hedberg Jr, *A survey of the advancing use and development of machine learning in smart manufacturing.* Journal of Manufacturing Systems, 2018.
57. Abadi, M., et al., *Tensorflow: Large-scale machine learning on heterogeneous distributed systems.* arXiv preprint arXiv:1603.04467, 2016.
58. Team, E.D.j.D., *Deeplearning4j: Open-source distributed deep learning for the JVM, Apache Software Foundation License 2.0.* . 2018.
59. Miškuf, M. and I. Zolotová. *Comparison between multi-class classifiers and deep learning with focus on industry 4.0*. in *Cybernetics & Informatics (K&I), 2016*. 2016. IEEE.
60. Kiumarsi, B., et al., *Optimal and autonomous control using reinforcement learning: A survey.* IEEE transactions on neural networks and learning systems, 2018. **29**(6): p. 2042-2062.
61. Carvalho, T.P., et al., *A systematic literature review of machine learning methods applied to predictive maintenance.* Computers & Industrial Engineering, 2019. **137**: p. 106024.
62. Jordan, M. and T. Mitchell, *Machine learning: Trends, perspectives, and prospects.* Science, 2015. **349**(6245): p. 255-260.
63. Hernavs, J., et al., *DEEP LEARNING IN INDUSTRY 4.0–BRIEF OVERVIEW.* Novi Sad, 2018, 2018. **21**(2): p. 1.
64. Wuest, T., et al., *Machine learning in manufacturing: advantages, challenges, and applications.* Production & Manufacturing Research, 2016. **4**(1): p. 23-45.
65. Schmidhuber, J., *Deep learning in neural networks: An overview.* Neural networks, 2015. **61**: p. 85-117.
66. LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning.* Nature, 2015. **521**(7553): p. 436-444.
67. Medsker, L. and L. Jain, *Recurrent neural networks.* Design and Applications, 2001. **5**.
68. Sak, H., A. Senior, and F. Beaufays. *Long short-term memory recurrent neural network architectures for large scale acoustic modeling*. in *Fifteenth Annual Conference of the International Speech Communication Association*. 2014.
69. Duan, Y., et al. *Benchmarking deep reinforcement learning for continuous control*. in *International Conference on Machine Learning*. 2016.
70. van Otterlo, M. and M. Wiering, *Reinforcement learning and markov decision processes*, in *Reinforcement Learning*. 2012, Springer. p. 3-42.

71. Watkins, C.J. and P. Dayan, *Q-learning.* Machine learning, 1992. **8**(3-4): p. 279-292.
72. Hester, T., et al. *Deep q-learning from demonstrations*. in *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.
73. Chen, C., V. Ying, and D. Laird, *Deep q-learning with recurrent neural networks.* stanford cs229 course report, 2016. **4**: p. 3.
74. Sutton, R.S., et al. *Policy gradient methods for reinforcement learning with function approximation*. in *Advances in neural information processing systems*. 2000.
75. Lowe, R., et al., *Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments.* arXiv preprint arXiv:1706.02275, 2017.
76. Konda, V.R. and J.N. Tsitsiklis. *Actor-critic algorithms*. in *Advances in neural information processing systems*. 2000.
77. Mnih, V., et al., *Playing atari with deep reinforcement learning.* arXiv preprint arXiv:1312.5602, 2013.
78. Kusiak, A., *Convolutional and generative adversarial neural networks in manufacturing.* International Journal of Production Research, 2019: p. 1-11.
79. Hausknecht, M. and P. Stone. *Deep recurrent q-learning for partially observable mdps*. in *2015 AAAI Fall Symposium Series*. 2015.
80. Arviv, K., H. Stern, and Y. Edan, *Collaborative reinforcement learning for a two-robot job transfer flow-shop scheduling problem.* International Journal of Production Research, 2016. **54**(4): p. 1196-1209.
81. Chien, C.-F., Y.-S. Lin, and S.-K. Lin, *Deep reinforcement learning for selecting demand forecast models to empower Industry 3.5 and an empirical study for a semiconductor component distributor.* International Journal of Production Research, 2020: p. 1-21.
82. Cimini, C., et al., *A human-in-the-loop manufacturing control architecture for the next generation of production systems.* Journal of Manufacturing Systems, 2020. **54**: p. 258-271.
83. Hart, S.G. and L.E. Staveland, *Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research*, in *Advances in psychology*. 1988, Elsevier. p. 139-183.
84. Djuric, A.M., R. Urbanic, and J. Rickli, *A Framework for Collaborative Robot (CoBot) Integration in Advanced Manufacturing Systems.* SAE International Journal of Materials and Manufacturing, 2016. **9**(2016-01-0337): p. 457-464.
85. Eckardt, M., A. Buchheim, and T. Gerngross, *Investigation of an automated dry fiber preforming process for an aircraft fuselage demonstrator using collaborating robots.* CEAS Aeronautical Journal, 2016. **7**(3): p. 429-440.
86. Agravante, D.J., et al. *Collaborative human-humanoid carrying using vision and haptic sensing*. in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. 2014. IEEE.
87. Rozo, L., et al., *Learning physical collaborative robot behaviors from human demonstrations.* IEEE Transactions on Robotics, 2016. **32**(3): p. 513-527.
88. Wang, Y. and F. Zhang, *Trends in Control and Decision-Making for Human–Robot Collaboration Systems*. 2017, Springer.
89. Sheridan, T.B., *Human–Robot Interaction.* http://dx.doi.org/10.1177/0018720816644364, 2016.
90. Hancock, P.A., et al., *A Meta-Analysis of Factors Affecting Trust in Human-Robot Interaction.* http://dx.doi.org/10.1177/0018720811417254, 2011.
91. Terziyan, V., S. Gryshko, and M. Golovianko, *Patented intelligence: Cloning human decision models for Industry 4.0.* Journal of manufacturing systems, 2018. **48**: p. 204-217.

92. Liu, H. and L. Wang, *Gesture recognition for human-robot collaboration: A review*. International Journal of Industrial Ergonomics, 2018. **68**: p. 355-367.
93. Compamy, T.A., *Anylogic*. 2018.