Computers in Industry 146 (2023) 103858

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.sciencedirect.com/journal/computers-in-industry

Real-time optimization for a Digital Twin of a robotic cell with human operators

Teresa Albini, Andrea Brocchi, Gianluca Murgia, Marco Pranzo *

Università degli Studi di Siena, DIISM, Via Roma 56, Siena, 53100, Italy

ARTICLE INFO

Keywords:
Digital Twin
Robotic cell with operator
Simulation–optimization algorithm
Reactive scheduling

ABSTRACT

In this paper, we develop a Digital Twin (DT) for a cell with a robotic arm serving seven stations and assisted by a human operator. The lack of reliable predictions of human behaviours requires constant monitoring of the cell in order not to affect the degree of context-awareness, autonomy, and adaptability of a DT. This monitoring is carried out through a real-time simulation and optimization algorithm, which continuously produces near-optimal decisions for machines and efficient recommendations for human operators. The proposed approach is tested in a large computational campaign based on real-world data. The results show that the behaviour of the operator scarcely affects the performance of the cell at least in the considered settings.

1. Introduction

Efficient modern manufacturing systems need to continuously react and adapt to changes provoked by both the external environment, i.e., the market and supply chain dynamics, and the internal environment, i.e., disturbances and disruptions in the manufacturing plant. In this context, the adoption of the Industry 4.0 paradigm aims at increasing the responsiveness of the factory floor to unexpected changes.

One key aspect of the Industry 4.0 is the introduction of the so called *Digital Twin* (DT) (Rosen et al., 2015), an accurate digital model representing the physical manufacturing system. Even if DTs can be applied for modelling different physical objects, from a single product to a whole city (Qi et al., 2021), their ability to provide a complete digital representation of physical manufacturing systems favours their use in production planning, simulation, and optimization. In particular, being based on the large amount of data generated in a smart factory, a data-driven DT (Friederich et al., 2022) is composed of three core components: model, state, and behaviour.

- The *model* is a digital representation of the physical manufacturing system and its components.
- The *state* is the real-time reflection of the physical manufacturing system as perceived by sensors and other IT systems.
- The *behaviour* is a description of the reactions of the manufacturing system in response to given inputs. This feature guarantees that the DT can constantly simulate the behaviour of the physical manufacturing system in a realistic way.

A DT includes high-resolution data collected by sensors, as well as information processed by other IT systems. These elements, coupled with the ability to simulate the behaviour of the physical manufacturing system, allow accurate predictions of the physical system. Hence, the combination of data collection with prescriptive analytics approaches ultimately allows for taking autonomous decisions in real-time (Rosen et al., 2015). However, to reach its full potential, a DT should be context-aware, autonomous, and adaptive (Hribernik et al., 2021).

Context-awareness is the ability to access and process any information describing the state of the entire physical system which includes not only the environmental data gathered by sensors or provided by ERP systems, but also information on the behaviours of human operators (Alexopoulos et al., 2016; Bisio et al., 2018). Nevertheless, the adoption of effective DTs may be hindered by the non-deterministic behaviours of human operators (Joo and Shin, 2019). The non-deterministic nature of the behaviour of human operators cannot be easily captured by sensors and other technologies, thus increasing the complexity of the models. In fact, when dealing with human operators, three main problems have to be addressed: *(i)* how to model human operators in the DT's state, *(ii)* how to predict their behaviours in the optimization algorithms and *(iii)* how to maintain the autonomy and agency of both the DT and human operators.

Once context information is accessible, it can be used in the DT's decision-making process, thus enabling its capability to react to environmental changes and take autonomous decisions. By monitoring the context status, a re-optimization process can be triggered when needed, and a new plan reflecting the mutated state can be generated in real-time. Clearly, autonomous decision-making requires the development

* Corresponding author.
E-mail address: marco.pranzo@unisi.it (M. Pranzo).

https://doi.org/10.1016/j.compind.2023.103858
Received 28 March 2022; Received in revised form 1 October 2022; Accepted 4 January 2023
Available online 18 January 2023

of real-time optimization algorithms, able to develop near-optimal decisions and provide useful suggestions for human operators.

Finally, a DT is adaptive when it can change its behaviour and its internal models are based on the context-awareness and autonomous decision making (Tao et al., 2018). In fact, the contextualized data streams collected from the field allow a continuous calibration and validation, i.e., a dynamical adaptation of the internal model parameters as soon as they start deviating from the nominal values.

Overall, a context-aware, adaptive and autonomous DT can react to changes and take optimized decisions faster than what would be possible when human intervention is needed.

In this paper, we introduce a context-aware, autonomous and adaptable DT of a robotic cell developed for MB Elettronica srl (in the following, MB), a company that designs, develops, and assembles Printed Circuit Boards (PCBs). The cell comprises a robotic arm assisted by a human operator, 3 workstations, a buffer, and 3 conveyors (input, output and fail). The robotic arm is in charge of moving the PCBs between the stations of the cell and directly performing some operations, whereas the human operator, oversees the management of the defective PCBs and the replenishment of the necessary materials for the cell. The DT aims at supporting the operational scheduling of the tasks in charge of two dedicated resources, i.e. the robotic arm and the human operator, trying to maximize the overall productivity of the cell. At this aim, sensors continuously record data, thus allowing informing the DT of the current state of the cell. The scheduling behaviour of the cell is autonomously optimized by the real-time optimization algorithm keeping into account and adapting to the behaviour of the human operator. The optimization process provides guidance to the human operator by signalling (without enforcing) the desired next moves. In fact, the algorithm evaluates the set of available moves by simulating the behaviour of two independent agents (i.e., the robotic arm and the human operator) and selects the most promising moves for both these agents, even if the human operator may not follow the DT's suggestion.

The contributions of the paper are threefold:

- We consider a robotic cell with a human operator in the PCBs production and we develop an autonomous, context-aware DT able to react in real-time to changes in the cell by updating the planned movements of the robot and suggestions for the operator (Section 3). The proposed architecture can be easily extended to consider different applications and scenarios.
- To deal with the uncertainty provoked by the possible moves of the human operator, we developed a reactive approach built on a simulation–optimization algorithm (Xu et al., 2016). This algorithm is based on the Approximate Dynamic Programming framework (Bertsekas, 2005) to give the DT a real-time autonomous decision-making capability (Section 4).
- We test the proposed approach on real-world data, thus showing its feasibility (Section 5). In detail, we show how the productivity of the cell is not very affected by the behaviour of the human operator as long as the operator is collaborative. These results may enrich the limited literature on DTs with human operators, and provide useful insights to manufacturing companies interested in the implementation of DTs for robotic cells.

The paper is organized as follows. In Section 2 we illustrate the main approaches discussed in the literature on the impact of DTs on the scheduling of manufacturing systems. In Section 3 we describe and introduce the problem at hand. In Section 4 we describe the algorithms we developed to solve the optimization problem and in Section 5 and Appendix we show our preliminary results. Finally, a discussion is carried out in Section 6 and conclusions and future research directions follow.

2. Literature review

DTs have been widely implemented in manufacturing to improve different processes, such as product design, layout planning, production scheduling and monitoring, and maintenance (Cimino et al., 2019; Kritzinger et al., 2018; Leng et al., 2021; Liu et al., 2021). The implementation of a DT to the solution of scheduling problems can be based on two different approaches (Negri et al., 2021): reactive and preventive.

In the reactive approach, when an issue occurs, new information is fed back to the scheduler to find a new solution. This assumes that a continuous recalculation must take place whenever a change in the physical production system is registered by the DT (Negri et al., 2021). In this regard, Eunike et al. (2022) present a DT architecture for constructing a decentralized scheduling system that is resilient to shop floor disruptions. Every time a disruption occurs, it will cause the DT to adjust the job execution sequence and forward the revised schedule to the machine. Liu et al. (2022) propose a DT-driven shop floor adaptive scheduling method able to capture dynamic events and issue warnings when dynamic events occur in the actual production process, and adaptively optimize the initial scheduling scheme. Even in this case, every time an abnormal event occurs, it triggers rescheduling, and re-executes the algorithm to obtain the latest results. In Tliba et al. (2022) the authors propose a DT-driven dynamic scheduling approach for a hybrid flow shop problem based on the readjustment of schedules in response to real-time events. After a scheduling run, if an internal or external event (or change) disrupts the current scheduling, the DT will update the parameter(s) impacted by this event either in the scheduling model or in the simulation model and, finally, reschedule the production accordingly. Leng et al. (2019) discuss a DT for a manufacturing system where the DT reacts to production events, such as new orders and changes in the workshop configuration. Some papers focus on the so called pure reactive scheduling or online scheduling, which is a generalization of reactive scheduling since it is based on recomputation that is carried out not only upon the realization of trigger events but also to periodically consider new information (Zhang et al., 2021a).

In other cases, data collected by a DT can be also used to implement a preventive approach to deal with the uncertainty that may affect operations. In particular, several robust approaches to scheduling have been proposed in the literature on DT applications in manufacturing. These approaches aim at reducing the impact of necessary rescheduling due to the uncertain dynamics that characterize the physical manufacturing system, which is constantly monitored by a DT (Rossit et al., 2019). At this aim, some approaches are based on the minimization of the adjustments to implement in each rescheduling (Fang et al., 2019), thus trying to reduce the organizational issues related to each change. Alternative approaches define stable scheduling schemes that should be recomputed only when an event in the physical manufacturing system, as detected by the DT, provokes a deviation from the expected performance of the system that overcomes a certain threshold (Rossit et al., 2019; Zhang et al., 2021b). This deviation can be evaluated by considering several performance dimensions, such as operational (Zhang et al., 2021b) and energy efficiency (Liang et al., 2018). Specifically, the expected performance of the manufacturing system can be computed starting from some critical factors for scheduling, such as the availability of workforce and equipment, production time, and energy consumption. The expected values of these critical factors can be computed through some prediction models that are based on big data collected over time by the DT in the physical manufacturing system (Tao and Zhang, 2017; Yu et al., 2021; Zhuang et al., 2018). During the production process, these values can be cyclically re-evaluated by using an updated version of the same prediction models, thus allowing the computation of scheduling schemes that can be more easily implemented in the physical manufacturing system. For example, Liang et al. (2018) propose a scheduling framework that includes a prediction

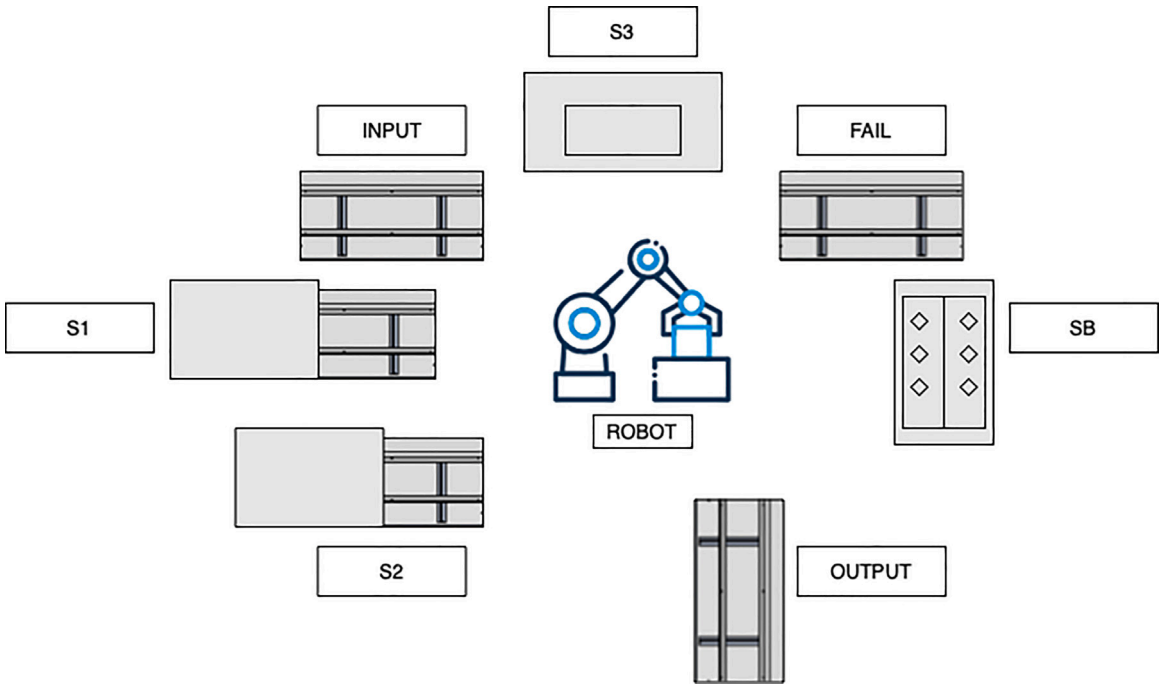

Fig. 1. Architecture of the cell.

model for the energy consumption of a shop floor, while Negri et al. (2021) combine scheduling with a prediction model for the equipment's health status. This latter framework, based on a genetic algorithm, assigns the jobs to the machines in the manufacturing system, supporting the identification of the scheduling alternatives that slow down the degradation of the machines, thus prolonging their lifespan. In general, these robust approaches can enable the implementation of more stable scheduling schemes, which may be extremely useful in planning at the tactic level and when the resources used in the production system can be easily assigned to alternative tasks. Nevertheless, the application of these robust approaches in scheduling problems at the operational level and/or in presence of dedicated resources may be less beneficial, especially if their computational time prevents fast rescheduling.

3. Problem description

The problem has been analysed within a project carried out by the Italian company MB. Fig. 1 shows a view of the architecture of the cell:

- S1 and S2 are two testing stations where, respectively, tests are performed on the components of the PCB by a Seica test machine and a visual inspection of the PCB components is performed by an Automated Optical Inspection (AOI) testing machine.
- S3 is a station where three different operations are performed on each PCB: a first silicone sealing, the insertion of the inductors, and a second silicone sealing. These operations are carried out without any interruption by the robotic arm, which is therefore busy for all their duration. The operator is in charge of refilling the S3 station when the silicone and inductors levels are low.
- SB is a buffer station which supports the robotic arm in the handling of PCBs during intermediate processing. The buffer can hold up to 6 PCBs.
- INPUT conveyor contains the PCBs to be processed, which are loaded by the operator.
- OUTPUT conveyor contains the finished PCBs, which are loaded by the robotic arm and should be taken away by the operator.
- FAIL conveyor contains the defective PCBs, which should be manually checked and reworked by the operator.

The processing flow is the following: PCBs are loaded by the operator on the INPUT conveyor, where they are read by an automatic reader. The robotic arm picks up the PCB in the first position from INPUT and moves it in an available test station, either S1 or S2, releasing the PCB on a conveyor that transports it inside the test machine. The tests (S1 and S2) can be performed in any order. Tests have two possible outcomes: *Pass* or *Fail*.

If a PCB fails a test, it is picked up by the robotic arm and left in FAIL. Here, the operator checks the defective PCB through a manual inspection in debug mode. If the result of this inspection is positive, the operator reinserts the PCB into the cell putting it in INPUT. Once back in the cell, the PCB will have to repeat the failed test and then continue with the remaining operations.

Only after positively passing both tests, a PCB can move in S3. The robotic arm places the PCB under the silicon dispenser, where the first silicone sealing is performed. Then, the robotic arm moves the PCB under the inductors dispenser. Once the inductors are inserted, the PCB is positioned again under the silicone dispenser for the second and final sealing. At the end of these operations, the robotic arm moves the completed PCB and releases it into the OUTPUT conveyor. Buffer positions SB can be used to hold some PCBs if desired.

In sum, the robotic arm is in charge of handling the PCBs from the various stations, as well as the silicone sealing and inductors insertion operations. Conversely, the operator is in charge of loading the PCBs into the INPUT conveyor, unloading the PCBs from the OUTPUT and the FAIL conveyors, reworking the defective PCBs, and refilling the silicon and inductors, when their levels drop under predetermined thresholds. During these last two operations, the robotic arm stops, allowing the operator to safely access the cell.

4. Architecture and algorithms of the digital twin

In this section, we describe the general algorithmic scheme adopted in the proposed DT. First, we introduce the general architecture of the DT, then we describe in detail all the main components.

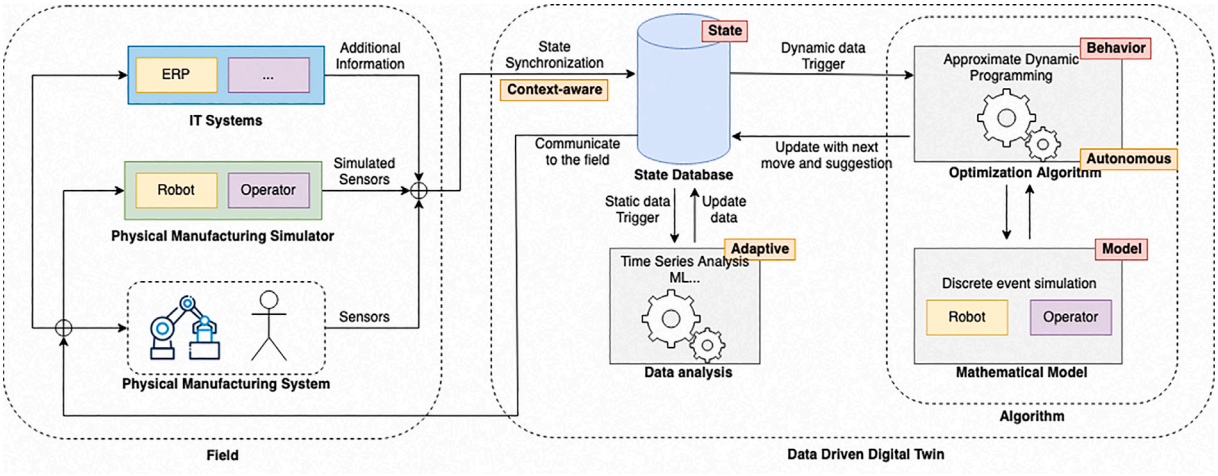

Fig. 2. Architecture of the DT.

4.1. The digital twin

In this section, we introduce the general architecture of the DT, shown in Fig. 2 and the general algorithmic scheme adopted in the proposed DT.

The main components of the DT are the State Database (DB) which may include Data Analysis processes and the Optimization Algorithm which contains the (Mathematical) Model. The DT interacts with the Field which comprises the Physical Manufacturing System, a Robotic Cell Simulator to test the DT when the physical manufacturing system is not available and possibly access to additional IT systems (e.g., ERP). Observe that, in a manufacturing cell, we can distinguish between two types of information: Dynamic and static. Dynamic information is data that changes frequently even over a short time horizon (e.g., the position of the robot) while static data changes slowly over time (e.g., degradation of a test cell). A DT needs data synchronization for both kinds of information. Changes in static data may trigger the Data Analysis process to maintain the model accuracy. Whereas, the frequent changes of dynamic data trigger a new run of the Optimization Algorithm while the cell is running. Therefore, the algorithm is called multiple times during the production and it is required to run within short CPU times to guarantee real-time reactions of the DT. More in detail, the flow of dynamic data is described in Fig. 3 and involves four actors.

As soon as an event is logged by the Sensors in the physical system (e.g., the robot changes position), the change is synchronized to the DB. Note that the sensors are only indirectly aware of the operator's actions, because they observe the results of the actions of the operator (e.g. the silicone level is refilled). Then, a new call of the algorithm is triggered to find the most promising move and the best suggestion for the operator. Internally the algorithm invokes a discrete event simulator to accurately predict the effects of different decisions. Hence, an extremely low computation time avoids idle times, thus running the algorithm on stale data. At the end of the execution, the results are communicated to the DB, which possibly communicates the next movement to the robot and the suggestion to the operator. Therefore, solving the optimization problem with high frequency allows a better synchronization of the planned movements with the current situation of the cell.

We describe all the components in detail below.

4.2. State database

The State DB contains all the dynamic and static information about the state of the system. This component is in charge of invoking the optimization algorithm whenever a state change is communicated by the sensors. Due to the high frequency updates to dynamic information, the algorithm is allowed at most one second of computation to achieve a real-time autonomous DT. On the other hand, as shown in Fig. 2, changes to static information could trigger the Data Analysis process, which may enable to work on more accurate data even under system changes.

4.3. Field

The field contains all the systems that can communicate to the DT changes in the state of the real world. More specifically, we include:

- The Physical Manufacturing System is connected to the DB through a set of sensors, continuously synchronizing the current state information. The state of all the stations and the robotic arm is directly monitored by the sensors in the cell and synchronized to the State database. Whereas the position of the human operator is only observed indirectly, i.e., sensors observe the results of the actions of the human operator.
- The Simulator is the digital version of the physical manufacturing system, which allows carrying out virtual tests, also when the physical system is not available. To account for the uncertainty of the physical production system, we developed a discrete-event simulation of the robotic cell based on two independent agents: the robotic arm and the human operator. In the simulation, the operations of the robotic arm are assumed to be deterministic and known in advance. The uncertainty is caused by three main characteristics of the cell:
  1. The duration of each operation carried out by the operator is assumed to be non-deterministic, but follows a known probability distribution.
  2. The algorithm provides suggestions to the operator, but the operator is free to override them, performing a different action. The simulator module assumes that the operator has a probability to accept the suggestion. If the suggestion is not accepted, the operator performs another available move.
  3. The outcome of the two tests that can be either *Pass* or *Fail* according to a known probability distribution.
- The IT systems include ERP, MES, and other information systems that provide information on the production lots and the technical characteristics of the cell. This produced information is static and it is used to define the instance to solve. Although there is no need to explicitly consider IT systems in the Simulator.

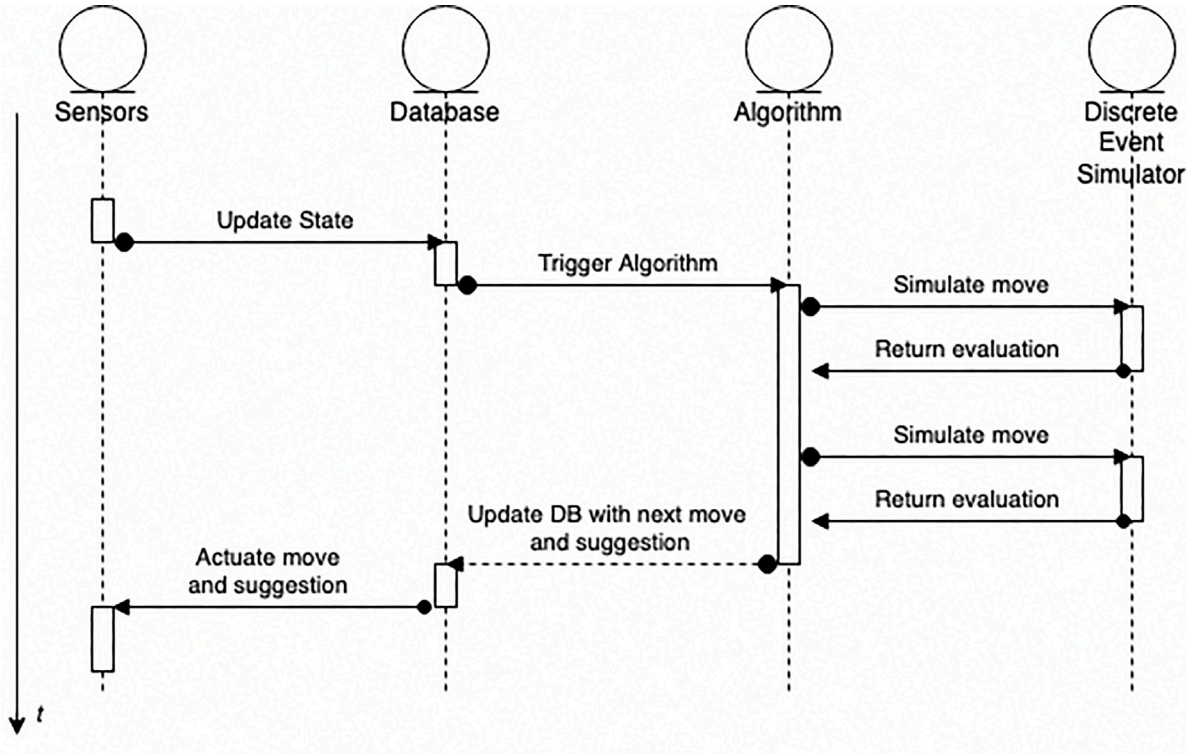

Fig. 3. Flow of dynamic data in the DT.

4.4. Optimization algorithm module

We have developed a dynamic optimization framework where, whenever the algorithm is triggered, it identifies the best available decision at the given moment. The optimization algorithm is triggered by the State database every time a change in the physical cell is reported by the sensors. The algorithm loads information about the current state of the system from the database and evaluates all the available moves choosing the most promising for the robot and the best suggestion, if any, to give to the operator. In order to accurately predict the future behaviour of the cell, it uses internally a simplified version of the cell simulator.

Let s be the current state at time t_0, and let $d \in D(s)$ be an available decision at state s where $D(s)$ is the set of possible decisions at s. $P(s)$ is the objective function associated with a state s. Let $s' = s \cup d$ be the state obtained by the current state s in which decision d has been applied. Conversely, let $\bar{s}$ be the predicted state at the end of the simulation starting from the state s' and ending the simulation at time $t_0 + T_p$.

When evaluating the available alternatives the optimization algorithm selects the best possible moves according to a score $P(s')$, thus allowing to identify the decision d to take at time t_0. The optimization algorithm runs an internal simulation to identify the most promising move. In each internal simulation, the algorithm takes decisions on behalf of the robotic arm and the human operator as they arise. The internal simulation is carried out for a limited time horizon T_p. At the end of T_p, the simulation is halted and the reached final state $\bar{s}$ is evaluated and the best decision d^* is identified

$$d^* = \min_{d \in D(s)} \{E(P(\bar{s}))\}$$

Then, the best decision d^* is returned and implemented in the physical system. The score P is computed as the lexicographic order of three criteria:

- **Numbers of PCBs Completed** ($NPCBC$) corresponds to the number of PCBs completed at $t_0 + T_p$.
- **Time to Complete** (TTC) corresponds to the sum of the flow times of each PCB, i.e., the sum of time spent by every PCB inside the cell.
- **Movement of Robot** (MOR) is the sum of the movement times of the robotic arm needed for the completion of all the PCBs.

More specifically, the lexicographic order selects the highest $NPCBC$; in case of a tie, the lowest TTC, and, in case of a further tie, the lowest MOR.

The pseudocode of the algorithm is described in detail in Algorithm 1. Note that, Algorithm 1 can be considered as an approximate dynamic program (Bertsekas, 2005) where, due to the presence of strict computational time constraints and the stochastic nature of the problem, we limit the lookahead to a single one-step lookahead policy.

Algorithm 1 Optimization Algorithm

```
Load information from the state database
Generate set D(s) of all possible moves
for d ∈ D(s) do
    Run internal simulation for T_p
    Let s̄ be the predicted state at the end of the simulation
    Evaluate P(s̄)
    Store the best d*
end for
Return d* to the state database
```

In the internal simulation, the algorithm performs choices on behalf of the robotic arm and the operator. The internal simulator used by the algorithm is a deterministic version of the physical cell simulator adopted by the DT. In fact, test operations are assumed to never fail, the operator always follows the suggestions and the duration of each move of the human operator is deterministic.

To identify the best moves of the robotic arm, a criterion W computing a score for each decision is used. More specifically, we tested three possible criteria W:

- **Closest To Completion** (CTC): Gives priority to the PCBs that are closest to being completed.
- **Closest To the Robot** (CTR): Gives priority to the PCBs closer to the current position of the robotic arm, thus minimizing its movement.
- **Shortest Processing Time** (SPT): Gives priority to the PCBs with the expected shortest processing time.

The suggestions for the operator are based on the predicted future operations of the robotic arm, and aim at avoiding blocks in the future moves of the robotic arm. The algorithm always provides the operator with an expected suggestion and a time by which the suggestion is expected. Therefore, the operator is free to choose whether to anticipate the suggestion and carry out the suggested operation immediately or not.

5. Computational results

To test the effectiveness of the algorithms discussed in the previous section, we first introduce the test campaign in Section 5.1, and then we describe the main results in Section 5.2. Further details on the results are presented in Appendix.

5.1. Test cases

Our computational campaign consists of simulating the production of 53 PCBs, an instance provided by MB that represents a typical situation for the cell. We considered three different types of PCBs (Type 1, Type 2, and Type 3), which are characterized by specific components. Out of a total of 53, 24 are Type 1, 16 are Type 2, and 13 are Type 3. They all require the same operations but differ in terms of processing time or silicone consumption. The numeric data are chosen on the basis of data collected in the real MB plant. Without loss of generality, the duration of the operations carried out by the operators is assumed to follow a uniform distribution.

Given this setting, we performed a set of 2430 experiments, each one defined by four key parameters:

- W: As discussed in the previous section, we tested three different criteria W for the selection of the next moves of the robotic arm: CTC, CTR and SPT.
- T_p: We selected three levels of the lookahead of the internal simulation, equal to 0, 1800, and 18,000 s. With the first value no internal simulation is performed, whereas the latter value is set to guarantee that the simulation runs to its completion by processing the whole set of PCBs.
- TS: The test success rate performed in the cell, precisely in S1 and S2 stations. In detail, we selected three levels of the success rate, equal to 80%, 85%, and 90%.
- OC: The operator compliance rate when performing the moves suggested by the algorithm. In line with the success rate of the tests, we selected three levels of the compliance rate of the operator, equal to 80%, 85% and 90%.

The values of TS and OC have been decided in accordance with MB Elettronica. By combining the levels of these four parameters, we obtained 81 different scenarios. Each scenario represents the production of the 53 PCBs, hence the optimization algorithm is run every time a decision has to be taken. To account for the randomness each scenario has been tested 30 times by using a specific random seed. To increase the comparability of the tests on these scenarios, we selected the same sequence of random seeds for the test of every scenario.

The algorithm has been implemented in Python 3 and the computational campaign was carried out by using Google Colab with Intel Xeon CPU at 2.20 GHz and 12 GB of RAM.

Table 1
Mean and standard deviation of the total processing time.

Criterion	Lookahead		
	$T_p = 0$	$T_p = 1800$	$T_p = 18000$
CTC	14781.77 (850.34)	14348.56 (617.98)	14360.41 (633.91)
CTR	14777.55 (843.34)	14150.24 (564.07)	14116.73 (580.18)
SPT	14747.34 (828.07)	14400.75 (635.69)	14393.85 (648.98)

Table 2
Mean and standard deviation of the total operator working time.

Criterion	Lookahead		
	$T_p = 0$	$T_p = 1800$	$T_p = 18000$
CTC	1445.13 (361.97)	1682.15 (361.63)	1655.46 (354.77)
CTR	1435.40 (359.17)	1647.76 (428.94)	1702.31 (402.94)
SPT	1501.79 (394.96)	1679.18 (369.96)	1694.81 (346.14)

Table 3
Mean and standard deviation of the total robot working time.

Criterion	Lookahead		
	$T_p = 0$	$T_p = 1800$	$T_p = 18000$
CTC	12236.10 (683.86)	12568.36 (458.02)	12558.90 (471.36)
CTR	11937.28 (707.73)	12565.44 (428.90)	12542.56 (403.59)
SPT	11781.42 (625.86)	12546.66 (484.28)	12508.28 (486.07)

5.2. Test results

Tables 1–4 present the descriptive results of our tests showing how they are affected by two main factors investigated in our analysis, which are W (the criteria used by the robotic arm for the selection of the next PCB to work) and T_p (the lookahead used by the internal simulation to identify the most promising move to perform in the cell).

Table 1 shows the mean and standard deviation of the total processing time, which is the total time spent by the cell to process the whole set of PCBs. It shows that the lowest total processing time occurs when the algorithm uses the CTR criterion and the maximum lookahead. In general, when the lookahead is zero, the total processing time is maximum. Whereas the reduction obtained passing from 1800 to 18000 s appears to be negligible or nonexistent. Concerning the criteria, CTR appears to be the best performing, at least when the lookahead is non-zero.

Table 2 shows the mean and standard deviation of the total operator working time, which is the total time spent by the operator to process the whole set of PCBs. It shows that the lowest total operator working time occurs when the algorithm uses the CTR criterion and the minimum lookahead. In general, when the T_P is zero, the total operator working time is minimum, but also in this case the increase obtained passing from 1800 to 18000 s appears to be limited or nonexistent. Concerning the criteria, CTR appears to reduce the overall involvement of the operator, at least when the lookahead is under 18000 s. When $T_p = 18{,}000$ CTC yields the best performance. By jointly observing Tables 1 and 2, we point out that the lowest total processing time is obtained when the operator is more involved in the cell operations.

Table 3 shows the mean and standard deviation of the total robot working time, which is the total time spent by the robotic arm to process the whole set of PCBs. It shows that the lowest total robot working time occurs when the algorithm uses the SPT criterion and the minimum lookahead. In fact, when the $T_p = 0$, the total robot working time is minimum. The maximum involvement of the robotic arm occurs when the $T_p = 1800$ s. Concerning the criteria, SPT appears to reduce the overall involvement of the robotic arm.

Table 4 shows the mean and standard deviation of the total CPU time, which is the time spent by the DT to compute and simulate the solution for the processing of the whole set of PCBs. It shows that the lowest total CPU time occurs when the algorithm uses the SPT

Table 4
Mean and standard deviation of the total CPU time.

Criterion	Lookahead		
	$T_p = 0$	$T_p = 1800$	$T_p = 18000$
CTC	0.33 (0.47)	21.21 (1.32)	124.62 (4.70)
CTR	0.33 (0.47)	21.21 (2.38)	123.01 (4.89)
SPT	0.31 (0.46)	20.69 (1.10)	121.42 (4.45)

Table A.5
Regression table of the total processing time.

Poisson regression model results	
C: CTR	−0.01*** (0.00) [−149.06]
C: SPT	0.00 (0.00) [17.00]
Lookahead	0.00*** (0.00) [−0.02]
Test success	−0.00*** (0.00) [−29.04]
Operator compliance	0.00 (0.000) [0.27]
Constant	9.76*** (0.03)
$N = 2430$	
Wald $\chi^2(5) = 205.12$***	
Log pseudo likelihood = −56219.38	
Pseudo $R^2 = 0.05$	

Robust standard errors in round brackets, Marginal effects in square brackets.
Significance level: *** $\alpha < 0.001$; ** $\alpha < 0.01$; * $\alpha < 0.05$; † < 0.1.

Table A.6
Regression table of the total operator working time.

Poisson regression model results	
C: CTR	−0.00 (0.01) [0.91]
C: SPT	0.02† (0.01) [30.92]
Lookahead	0.00*** (0.00) [0.01]
Test success	−0.00† (0.00) [−2.96]
Operator compliance	0.00 (0.00) [0.09]
Constant	7.49*** (0.14)
$N = 2430$	
Wald $\chi^2(5) = 76.95$***	
Log pseudo likelihood = −137843.53	
Pseudo $R^2 = 0.02$	

Robust standard errors in round brackets, Marginal effects in square brackets.
Significance level: *** $\alpha < 0.001$; ** $\alpha < 0.01$; * $\alpha < 0.05$; † < 0.1.

criterion and the minimum lookahead. As expected by increasing the duration of the internal simulation the required total CPU time grows. However, even when the full lookahead is adopted, the CPU time is around 143 s to simulate the full 53 PCBs instance, never requiring more than half a second to compute the best move. On average, the optimization algorithm is called 326 times. Hence, the time necessary for each call is well under one second, thus enabling the real-time real-world application of our algorithm. Concerning the criteria, SPT appears to slightly reduce the total CPU time, but we do not observe any relevant difference between the different criteria.

To further analyse how the four dependent variables are affected by the criteria and the lookahead, as well as by other control variables, we carried out four different regression models. Specifically, since all the dependent variables are non-negative count variables, we performed Poisson regression models. We computed robust standard errors to reduce the impact of heteroskedasticity. Appendix contains the detailed results of the regressions.

From these regressions, it turns out that only TS rate has a significant impact on the total processing time since a higher success rate in the tests reduces the number of PCBs to rework and, consequently, the total processing time. While TS significantly improves the performance, OC has always a not significant effect. A higher T_p increases the performance, but its effect tends to decrease over a certain threshold, thus suggesting setting a non-zero lookahead to reach good performance, but not excessively high to slow down the algorithm. Concerning the criteria, CTR reduces the total processing time of the cell and guarantees an efficient use of the robotic arm. The use of the robotic arm is minimized by using the SPT criterion, which also guarantees the minimization of the total CPU time.

6. Discussion

In this paper, we have developed and tested a DT which has been applied to a case study arising from a real production cell with a robotic arm and a dedicated operator. Our findings show that the proposed architecture and algorithm can be successfully applied in practice, and point out how its performance is specifically affected by some contextual factors.

The literature on the application of DT to the solution of scheduling problems proposed several different approaches, which are characterized by different ways to deal with the uncertainty that affects the planning of manufacturing tasks. Our proposed DT falls in the reactive approach to scheduling problems, which assumes recalculating a new scheduling scheme once new information from the physical manufacturing system is observed. A crucial aspect of the reactive approach is the CPU time, which can be extremely high, since every change in the physical manufacturing system involves a continuous recalculation of the solution. In this regard, several contributions develop more advanced algorithms, such as mixed integer linear programming (MILP) (Tliba et al., 2022) and genetic algorithms (Eunike et al., 2022; Liu et al., 2022), to give the DT a real-time autonomous decision making capability. Compared with our approach, most of the algorithms proposed in the literature tend to require more computation and thus to have a lower computational efficiency.

Indeed, our paper presents a simpler algorithmic approach while still providing reactive scheduling, since it proposes a real-time optimization and simulation algorithm that enables automatic scheduling adaptation in response to changes in the physical system. More specifically, at the algorithmic level, we use a simplified approximate dynamic program approach proposed by Bertsekas (2005) thus guaranteeing solutions in a limited CPU time.

Another original contribution of the present paper is that our system considers a human operator to support production. This raises the problem of the uncertainty of his actions, which has not been much explored in the literature so far. Some studies do not take into account the uncertainty associated with the human operator (Eunike et al., 2022; Negri et al., 2021; Zhang et al., 2021a). Others simply include this uncertainty as a disturbing factor in the production process, triggering the change in the scheduling (Fang et al., 2019). Conversely, our approach takes into account the uncertainty of the human operator's actions by developing a suggestion mechanism for the operator. The suggestion mechanism allows for performing accurate forecasts, while at the same time safeguarding human autonomy and agency. In fact, if the operator decides not to follow the suggestion, the algorithm simply adapts its future choices around the observed behaviour of the operator. Interestingly, results show that operator compliance with the suggestion is less relevant than other sources of uncertainty, at least in the considered settings.

Our work can also make a practical contribution to managers or supervisors of robotic cells as it can support them in implementing the physical system through a fast and effective scheduling algorithm. In fact, by connecting the DT to the Physical Manufacturing System, it can be used at the operational level to control and optimize the production of the cell in real-time. On the other hand, by connecting it with a Physical Manufacturing Simulator, it can be used both at tactical level and at the strategic level to predict the productivity of the cell to a new production mix or new products. Moreover, the proposed framework can be easily adapted to address manufacturing cells with different configurations.

7. Conclusions and future research

In this paper, we developed a DT for a robotic cell with the presence of a human operator. The proposed approach can be easily adapted to consider robotic cells with different layouts and patterns of interaction and collaboration between a robot and human operators.

A possible future research direction will be to test the approach in the actual physical system and to develop modules to address the DT adaptability more explicitly. A different line of research will aim at extending the approach to a more general configuration of the cell, or to completely different manufacturing settings.

CRediT authorship contribution statement

Teresa Albini: Data Curation, Formal analysis, Investigation, Validation, Writing – original draft, Writing – review & editing. **Andrea Brocchi:** Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Gianluca Murgia:** Conceptualization, Data curation, Formal analysis, Investigation, Project administration, Supervision, Writing – original draft, Writing – review & editing. **Marco Pranzo:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

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

Data availability

The data that has been used is confidential.

Acknowledgements

This work has been partially funded by Regione Toscana, Italy (POR CREO FESR Toscana 2014–2020) project CIS 5.0. We would like to express our sincere gratitude to Marco Valentini and Francesco Caselli from MB Elettronica for their assistance and data contribution to this work.

Appendix. Regression analysis

Tables A.5–A.8 show the results of the regression models considering as dependent variable, respectively, the total processing time, the total operator working time, the total robot working time, and the total CPU time.

Table A.5 confirms that the total processing time is significantly reduced by using the CTR criterion and increasing the lookahead. This effect is close to zero, thus confirming that the impact of lookahead tends to decrease over a certain threshold. Concerning the control variables, only TS has a significant impact since a higher success rate in the tests reduces the number of PCBs to rework and, consequently, the total processing time.

Table A.6 shows that the total operator working time is increased by using the SPT criterion, although this effect is scarcely significant, and by increasing the lookahead. Even in this case, the effect of lookahead is close to zero, thus confirming that it tends to decrease over a certain threshold. Concerning the control variables, only TS has a negative and scarcely significant impact since a higher success rate in the tests reduces the number of PCBs to rework and, consequently, the operator moves necessary for their processing.

Table A.7 shows that the total robot working time is increased by using the CTC, while the other criteria, especially the SPT, significantly reduce the total robot working time. Even in this case, the effect of lookahead is significant, but close to zero, thus confirming that it tends to decrease over a certain threshold. Concerning the control variables, only TS has a negative and significant impact since a higher success rate in the tests reduces the number of PCBs to rework and, consequently, the robotic arm moves necessary for their processing.

Finally, Table A.8 shows that the total CPU time is significantly reduced by using the SPT criterion. Even in this case, the effect of lookahead is significant, but close to zero, thus confirming that it tends to decrease over a certain threshold. Concerning the control variables, again only TS has a negative and significant impact since a higher success rate in the tests reduces the number of PCBs to rework and, consequently, the time spent by the algorithm to define the necessary operations for their processing.

Table A.7
Regression table of the total robot working time.

Poisson regression model results	
C: CTR	-0.01^{***} (0.00) [−105.68]
C: SPT	-0.01^{***} (0, 00) [−175.58]
Lookahead	0.00^{***} (0.00) [0.02]
Test success	-0.00^{***} (0.00) [−28.11]
Operator compliance	0.00 (0.00) [0.12]
Constant	9.61^{***} (0.03)
$N = 2430$	
Wald $\chi^2(5) = 310.71^{***}$	
Log pseudo likelihood = −46658.11	
Pseudo $R^2 = 0.08$	

Robust standard errors in round brackets, Marginal effects in square brackets.
Significance level: *** $\alpha < 0.001$; ** $\alpha < 0.01$; * $\alpha < 0.05$; † < 0.1.

Table A.8
Regression table of the total CPU time.

Poisson regression model results	
C: CTR	−0.01 (0.01) [−0.53]
C: SPT	-0.03^{***} (0.01) [−1.25]
Lookahead	0.00^{***} (0.00) [0.01]
Test success	-0.01^{***} (0.00) [−0.27]
Operator compliance	0.00 (0.00) [−0.00]
Constant	2.67^{***} (0.10)
$N = 2430$	
Wald $\chi^2(5) = 12808.57^{***}$	
Log pseudo likelihood = −13919.63	
Pseudo $R^2 = 0.83$	

Robust standard errors in round brackets, Marginal effects in square brackets.
Significance level: *** $\alpha < 0.001$; ** $\alpha < 0.01$; * $\alpha < 0.05$; † < 0.1.

References

Alexopoulos, K., Makris, S., Xanthakis, V., Sipsas, K., Chryssolouris, G., 2016. A concept for context-aware computing in manufacturing: The white goods case. Int. J. Comput. Integr. Manuf. 29, 839–849.

Bertsekas, D.P., 2005. Dynamic programming and suboptimal control: A survey from ADP to MPC. Eur. J. Control 11, 310–334.

Bisio, I., Garibotto, C., Grattarola, A., Lavagetto, F., Sciarrone, A., 2018. Exploiting context-aware capabilities over the Internet of Things for industry 4.0 applications. IEEE Netw. 32, 101–107.

Cimino, C., Negri, E., Fumagalli, L., 2019. Review of digital twin applications in manufacturing. Comput. Ind. 113, 103130.

Eunike, A., Wang, K.J., Chiu, J., Hsu, Y., 2022. Real-time resilient scheduling by digital twin technology in a flow-shop manufacturinag system. Procedia CIRP 107, 668–674.

Fang, Y., Peng, C., Lou, P., Zhou, Z., Hu, J., Yan, J., 2019. Digital-twin-based job shop scheduling toward smart manufacturing. IEEE Trans. Industr. Inform. 15, 6425–6435.

Friederich, J., Francis, D.P., Lazarova-Molnar, S., Mohamed, N., 2022. A framework for data-driven digital twins for smart manufacturing. Comput. Ind. 136, 103586.

Hribernik, K., Cabri, G., Mandreoli, F., Mentzas, G., 2021. Autonomous, context-aware, adaptive digital twins—state of the art and roadmap. Comput. Ind 133, 103508.

Joo, T., Shin, D., 2019. Formalizing human–Machine interactions for adaptive automation in smart manufacturing. IEEE Trans. Hum. Mach. Syst. 49, 529–539.
Kritzinger, W., Karner, M., Traar, G., Henjes, J., Sihn, W., 2018. Digital twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine 51, 1016–1022.
Leng, J., Wang, D., Shen, W., Li, X., Liu, Q., Chen, X., 2021. Digital twins-based smart manufacturing system design in Industry 4.0: A review. J. Manuf. Syst. 60, 119–137.
Leng, J., Zhang, H., Yan, D., Liu, Q., Chen, X., Zhang, D., 2019. Digital twin-driven manufacturing cyber–physical system for parallel controlling of smart workshop. J. Ambient Intell. Humaniz. Comput. 10, 1155–1166.
Liang, Y.C., Lu, X., Li, W.D., Wang, S., 2018. Cyber physical system and Big Data enabled energy efficient machining optimisation. J. Clean. Prod. 187, 46–62.
Liu, M., Fang, S., Dong, H., Xu, C., 2021. Review of digital twin about concepts, technologies, and industrial applications. J. Manuf. Syst. 58, 346–361.
Liu, L., Guo, K., Gao, Z., Li, J., Sun, J., 2022. Digital twin-driven adaptive scheduling for flexible job shops. Sustainability 14 (5340).
Negri, E., Pandhare, V., Cattaneo, L., Singh, J., Macchi, M., Lee, J., 2021. Field-synchronized digital twin framework for production scheduling with uncertainty. J. Intell. Manuf. 32, 1207–1228.
Qi, Q., Tao, F., Hu, T., Anwer, N., Liu, A., Wei, Y., Wang, L., Nee, A.Y.C., 2021. Enabling technologies and tools for digital twin. J. Manuf. Syst. 58, 3–21.
Rosen, R., Wichert, G.von., Lo, G., Bettenhausen, K.D., 2015. About the importance of autonomy and digital twins for the future of manufacturing. IFAC-PapersOnLine 48, 528–539.
Rossit, D.A., Tohmé, F., Frutos, M., 2019. Industry 4.0: Smart scheduling. Int. J. Prod. Res. 57, 3802–3813.
Tao, F., Zhang, M., 2017. Digital twin shop-floor: A new shop-floor paradigm towards smart manufacturing. IEEE Access 5, 20418–20427.
Tao, F., Zhang, H., Liu, A., Nee, 2018. Digital twin in industry: State-of the-art. IEEE Trans. Industr. Inform. 15, 2405–2415.
Tliba, K., Diallo, T.M.L., Penas, O., Khalifa, R.B., Yahia, N.B., Choley, J.Y., 2022. Digital twin-driven dynamic scheduling of a hybrid flow shop. J. Intell. Manuf. http://dx.doi.org/10.1007/s10845-022-01922-3.
Xu, J., Huang, E., Hsieh, L., Lee, L.H., Jia, Q.-S., Chen, C.-H., 2016. Simulation optimization in the era of industrial 4.0 and the industrial internet. J. Simul. 10, 310–334.
Yu, H., Han, S., Yang, D., Wang, Z., Feng, W., 2021. Job shop scheduling based on digital twin technology: A survey and an intelligent platform. Complexity 8823273.
Zhang, J., Deng, T., Jiang, H., Chen, H., Qin, S., Ding, G., 2021a. Bi-level dynamic scheduling architecture based on service unit digital twin agents. J. Manuf. Syst. 60, 59–79.
Zhang, M., Tao, F., Nee, A.Y.C., 2021b. Digital twin enhanced dynamic job-shop scheduling. J. Manuf. Syst. 58, 146–156.
Zhuang, C., Liu, J., Xiong, H., 2018. Digital twin-based smart production management and control framework for the complex product assembly shop-floor. J. Adv. Manuf. Technol. 96, 1149–1163.