The International Journal of Advanced Manufacturing Technology (2022) 120:6547–6563
https://doi.org/10.1007/s00170-022-09164-6

ORIGINAL ARTICLE

A digital twin ecosystem for additive manufacturing using a real-time development platform

Minas Pantelidakis[1] · Konstantinos Mykoniatis[1] · Jia Liu[1] · Gregory Harris[1]

Received: 18 January 2022 / Accepted: 31 March 2022 / Published online: 13 April 2022

Abstract

Additive manufacturing is often used in rapid prototyping and manufacturing, allowing the creation of lighter, more complex designs that are difficult or too expensive to build using traditional manufacturing methods. This work considers the implementation of a novel digital twin ecosystem that can be used for testing, process monitoring, and remote management of an additive manufacturing–fused deposition modeling machine in a simulated virtual environment. The digital twin ecosystem is comprised of two approaches. One approach is data-driven by an open-source 3D printer web controller application that is used to capture its status and key parameters. The other approach is data-driven by externally mounted sensors to approximate the actual behavior of the 3D printer and achieve accurate synchronization between the physical and virtual 3D printers. We evaluate the sensor-data-driven approach against the web controller approach, which is considered to be the ground truth. We achieve near-real-time synchronization between the physical machine and its digital counterpart and have validated the digital twin in terms of position, temperature, and run duration. Our digital twin ecosystem is cost-efficient, reliable, replicable, and hence can be utilized to provide legacy equipment with digital twin capabilities, collect historical data, and generate analytics.

Keywords Additive manufacturing · Digital twin · Fused deposition modeling · Simulation · Unity 3D

1 Introduction

Digital twin (DT) technology is one of the fastest-growing concepts of Industry 4.0. According to a 2018 Gartner study conducted, 75% of organizations implementing IoT already use DT or plan to do so within a year [1]. The DT concept refers to a digital informational construct that mirrors the behavior of an observable physical or cyber-physical system in a real-time simulation environment [2]. The digital information is a “twin” of the information contained in the physical system and can be connected to the observable system throughout its lifecycle [3]. According to a recent study, the DT market was valued at $3.8 billion in 2019 and is expected to increase to $35.8 billion by 2025 [4]. The research in DT technology has been a hot area in both academia and industry, and a variety of tools, frameworks, and architectures have been proposed and built for accelerating DT applications [5–10]. Moreover, with the emergence of DTs in different industries, efforts have been made to review and categorize DTs by domain expertise [11, 12] or algorithms [13] for better understanding and developing DTs. As our awareness of the benefits of a DT grows and the tools for constructing a DT improve, the definition of the term DT continues to evolve. In this paper, DT is described as a fully connected functional digital representation of an observable manufacturing asset that runs in real-time (or near real-time) mode and is driven by data and a set of attributes from the physical system [2].

The DT market is being driven by the growth of Industry 4.0 technologies (e.g., IoT [14], cloud computing, additive manufacturing, modeling, and simulation [15], extended reality, artificial intelligence [16]) and the desire to reduce costs and shorten product development time [17].

✉ Konstantinos Mykoniatis
kmykoniatis@auburn.edu

Minas Pantelidakis
mzp0122@auburn.edu

Jia Liu
lzj0040@auburn.edu

Gregory Harris
gah0015@auburn.edu

[1] Industrial and Systems Engineering, Auburn University, 357-359 W Magnolia Ave, Auburn, AL 36832, USA

Additionally, DTs can assist with process monitoring, virtual commissioning, predictive maintenance, lifecycle management, process diagnosis, and production responses [18–23]. DTs also provide information interfaces between humans and machines to improve operators' capability and productivity in manufacturing [24–26]. Moreover, DTs can address the difficulties of productivity and quality control in industrial processes and allow organizations to act on a digital model for testing, and optimization [27]. Executing testing and optimization can accelerate prototyping, generate optimal decisions, eliminate instabilities caused to the real system, and assess a system's performance and efficacy.

At the same time, additive manufacturing (AM) is quickly growing, while also improving scalability and strategic options. Leaders in manufacturing are implementing AM across their organizations and taking advantage of the increased flexibility this technology provides. AM is considered an emerging and disruptive Industry 4.0 technology that offers superior customization and flexibility in designing and manufacturing complex products, greatly improves the competitiveness of manufacturing, and reduces lead time and material waste [28]. Manufacturers that desire to benefit from AM initiatives may discover that success requires cross-functional engagement and a willingness to move beyond the status quo in terms of product and process design, and in support of novel and innovative approaches. Utilizing a 3D real-time development platform for generating a DT for an AM process facilitates design and operation for highly customized and low-volume production. Fused deposition modeling (FDM) [29], also known as material extrusion or fused filament fabrication, is the most widely used AM process in prototyping, low-volume production for non-critical parts, training, and education. This can be attributed to its low risk, convenience, low cost, and friendly user interface [30–33]. An FDM AM machine typically heats filament of thermoplastic material (e.g., acrylonitrile butadiene styrene), extrudes it through a nozzle onto a movable build platform, and builds a 3D product in a layer-wise manner based on digital 3D models [32]. This process creates layers of material that adhere to one another. Layer bonding is controlled through pressure, temperature control, and chemical agents [28].

Interconnectivity and interoperability of industrial machinery are improving [34]. The wide adoption of data semantic standards (e.g., MTConnect [35]) further accelerates these changes. However, not every manufacturer has the resources or incentive to upgrade to a contemporary solution and therefore continue to utilize their trusted and consistent legacy equipment. Small and medium-sized enterprises (SMEs) in particular could greatly benefit from solutions that bring legacy equipment to the digital era. Such solutions should undoubtedly be reliable, straightforward, inexpensive, and require minimum technical expertise.

The objective of this research is to provide insight to integrate real-time development platform capabilities, data acquisition, analytics, and advanced modeling and simulation techniques into developing DTs of various manufacturing processes and systems. Specifically, in this work, we present a digital twin ecosystem (DTE) for an FDM AM machine, a Lulzbot Taz Workhorse 3D printer. We describe the implementation and assessment of a DTE in a simulated virtual environment using a real-time development platform (Unity 3D). The DTE is comprised of two different approaches. The first approach uses data collected by Octoprint,[1] an open-source printer host software. The second approach is a replicable sensor-data-driven approach, that can be generalized and assist in the digitalization of legacy machines. We verify and validate our sensor-data-driven approach against the Octoprint-data-driven approach, which we consider as ground truth, as Octoprint has direct access to the firmware of the printer. Both approaches rely on state-of-the-art open-source and free-of-charge software. The sensor-driven approach utilizes widely available, reliable, and cost-efficient sensors, mounted externally, in a non-disruptive manner. It also enables communication and synchronization with other devices and sharing of process data via a local network. Both approaches result in a novel data-driven soft-real-time DTE of the 3D printer, implemented with the Unity 3D real-time game development platform. The simulated virtual environment is configurable and allows remote access to machine information such as key process parameters, operational conditions, built-in sensor data, and user-added sensor signals. It also provides on-demand reports of tracked process variables and can collect and store historical data. We assess the real-time mimicking ability of the sensor-data-driven DT module in terms of extruder position and temperature in static and dynamic conditions, by comparing it to the Octoprint-data-driven DT module. We also evaluate the DTE in terms of 3D printer task duration and status. The response time of the DTE is also assessed. To the authors' knowledge, the developed DTE is at the frontier of FDM DT research. We developed a digital replication of an FDM physical system in a real-time development platform, synchronized the activities between the physical process and its digital twin, and established an ecosystem with bidirectional information flow.

The rest of this paper is structured as follows: In Sect. 2, we provide an overview of the current state of research regarding DTs for manufacturing and explore the different approaches to build a DTE. In addition, we highlight the benefits and limitations of existing DT approaches and define the scope of our research in terms of the published literature. In Sect. 3, we describe the implementation of the DTE and give a detailed explanation of our methodology.

[1] https://octoprint.org/

In Sect. 4, we evaluate the developed DTE and report the results. Finally, in Sect. 5, we discuss our conclusions and future research directions.

2 Related work

2.1 Digital twin background

DTs have been heralded as a major milestone towards the transition to Industry 4.0, where advanced digitalization drives informed decision-making. Manufacturing benefits from such a system in multiple ways (e.g., flexibility in design and manufacturing, dynamic recalibration of equipment, insights into performance aspects of the production line and manufacturing process, and energy footprint reduction). As such, new production models have emerged, where ubiquitous connectivity of components provides data flow from product conceptualization and design to production and distribution [36].

The definition of a DT within AM is still controversial [37], with some researchers defining it as a virtual system that mimics a physical system [38–41], while others adopt a simulation-centric approach [42–45]. The physical behavior of FDM involves heat transfer, materials phase changes, and machine mechanics that complicate the development of DTs [46, 47]. Moreover, existing works about FDM mainly focus on individual parts of the FDM process (e.g., the cooling down of the deposited filament, the flow in the nozzle, the bond between layers [48–51]) instead of examining the process as a whole.

There is a consensus of thought regarding the importance of incorporating sensor data into DTs, as a means towards the acquisition of real or near-real-time data from physical processes [52–55]. However, this concept is still in its development stage. “DTs” developed with this concept only have sensor data from the physical FDM machine, but lack the digital replication of the physical system in the virtual space; therefore, they are somewhat difficult to be differentiated from the traditional sensor-based process monitoring, such as [14, 32, 56].

In legacy equipment that does not inherently implement closed-loop control with networking capabilities, integration of digitalization poses a bigger challenge. Installing external sensors and networking solutions seems like a promising direction [57, 58]. Another body of research explores side-channels (e.g., acoustics, energy footprint), as a non-intrusive way to reveal information about the state and behavior of physical entities [59]. However, such an approach would be out of the scope of this work.

A DTE system typically consists of two major components. The first component collects data from physical assets, processes, optionally stores locally or on the cloud, and serves them continuously (e.g., streaming), or when queried, via a network. This component also has access to the machine; therefore, it can potentially intervene in the process. This component can be split into multiple sub-components, as described in the work of [57] and [60]. In [57], the authors use five different layers: (i) embedded devices, (ii) data acquisition and control, (iii) local data storage, (iv) networking, (v) cloud storage. In [60], the authors use the 5C structure, which consists of five layers: (i) connection, which describes data acquisition; (ii) conversion, which refers to transforming raw data to a useful format; (iii) cyber, which describes the generation of analytics, (iv) cognition, meaning self-awareness, and autonomous failure detection; and (v) configuration, referring to self-adaption to improve efficiency. These layers provide functionality that is beneficial to the operation of the DT. However, an effective DTE could omit certain layers of the referenced architectures, depending on system requirements, the scale of the data being processed, and its level of maturity. At its basic form, this component should communicate with a machine, process data, and make them available via a network. Therefore, in our work, we refer to this component as the data acquisition-processing-distribution component (APDC). Section 3 describes APDC in more detail.

The second major component of a DTE is a system that utilizes available data to conduct a real-time simulation of the physical entity, presents the algorithmically extracted insight to users, and could be also used for optimization studies and “what-if” analysis. This component is referred to as layer vi emulation and simulation in the work of [57]. In this research, we use the term virtual representation component (VRC), further described in Sect. 3.

In the last decade, a substantial body of research utilized 3D development platforms, such as Unity 3D and Epic Unreal game engines to conduct simulation studies and develop DT applications for various case studies, ranging from industrial simulation studies [61–63] to robotics [64, 65], autonomous vehicles [66], smart cities [67], and remote surgery [68]. 3D real-time development platforms feature a certain set of characteristics (e.g., networking, rendering engine, physics engine, event system, animation system, custom logic design) that makes them particularly suitable candidates for DTs. This work utilizes the Unity 3D development platform to implement the VRC of a DT for an FDM AM machine and open-source software and cost-efficient hardware for the APDC.

2.2 Additive manufacturing background

Additive manufacturing (also known as rapid prototyping or 3D printing) is a manufacturing technique that involves building up layers of a material to form a solid product. Figure 1 illustrates a 3D printing process. While there are

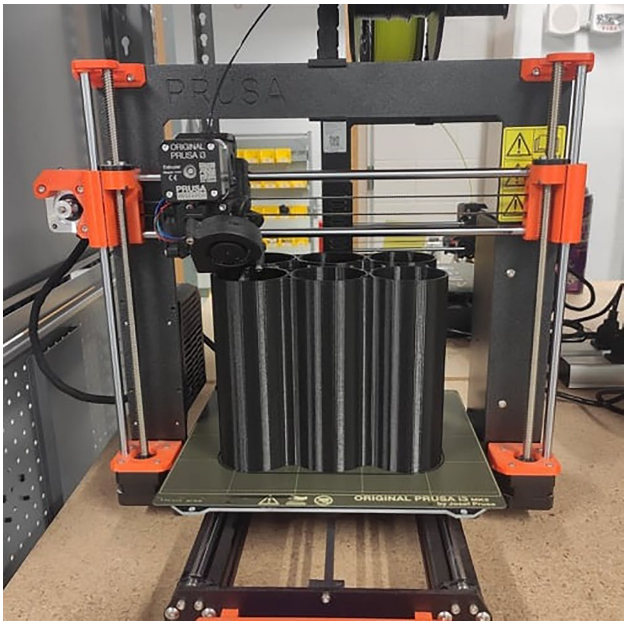


Fig. 1 Image captured during the 3D printing process. The 3D printer extrudes layers of filament to produce a solid object

several 3D printing technologies (e.g., direct metal laser sintering, stereolithography, selective laser sintering, digital light process), here, we concentrate on the process of fused deposition modeling (FDM), which is one of the most popular and accessible AM processes. During the COVID-19 pandemic, hundreds of FDM machines at different locations in the USA provided nearly one million pieces of safe personal protective equipment for local medical providers through the 3D Printing COVID-19 Rapid Response Initiative. It has great potential to be deployed in distributed manufacturing and to act as a catalyst for ubiquitous manufacturing in the USA [69, 70].

The general approach remains the same across all these AM methods, whether the end product is a rapid prototype or a fully functional part. The initial stage in the AM process is to create a digital model. The digital model is most typically created using computer-aided design (CAD) tools. Through 3D or structured light scanning, reverse engineering can also be employed to create a digital model. When designing for AM, there are various factors to consider such as geometry limits, structural requirements, and orientation.

An important step in the AM process that differentiates it from subtractive manufacturing methods is the requirement to convert a CAD model into a stereolithography file (STL). An STL file describes an object's surfaces using triangles (polygons). Once an STL file has been created the file is imported into a slicer program (e.g., Cura LulzBot Edition) and is converted to G-code. G-code is a programming language for numerical control (NC), which is used to control automated machine tools in computer-aided manufacturing (CAM) including 3D printers, and CNC machines. The G-code commands are tailored for the printer's firmware and dictate all aspects of the printing process (e.g., initial calibration, temperature levels, acceleration, feed rate, extrusion rate, movement coordinates). Therefore, a G-code file is the most essential component of the printing process and an indirect way to monitor the system. It is worth noting that slicing engines can also generate estimates of the amount of time needed to print the 3D object. These estimates often lack accuracy but can be further improved with plug-ins available for most printer host software solutions.

Since most FDM 3D printing machines are composed of many small, delicate components, proper maintenance, and regular calibration are essential for producing accurate prints. After the print has started, most AM machines do not need monitoring. The machines operate autonomously, with problems occurring usually when the machine runs out of material or there is a software issue.

3 Methodology

This section describes the methodology that we followed to develop a novel DTE using an open-source 3D printer web controller application and Unity 3D real-time development platform. We implement the DTE in a modular manner so that each component is a self-reliant entity while being able to work as part of the ecosystem. The two major components of the DTE architecture are (i) the data acquisition-processing-distribution component (APDC), and (ii) the virtual representation component (VRC). The orchestrated interweaving of the above components enables the DTE for continuous synchronization between the physical system and its digital counterpart. The use of only open-source and free-of-charge technologies was one of the main principles behind our design and implementation. We strongly believe that DT enabling technologies should be widely accessible to the public so that the Technology Readiness Level (TRL) of these tools further matures. A high-level diagram of the DTE is illustrated in Fig. 2. The following sections describe the different components of our methodology in further detail.

3.1 Data acquisition-processing-distribution component

The data acquisition-processing-distribution component (APDC) consists of a digital interface that hosts Octoprint and implements a Representational State Transfer (REST) Application Programming Interface (API) to serve external sensor data. The APDC collects, processes, and distributes data collected both from Octoprint and from

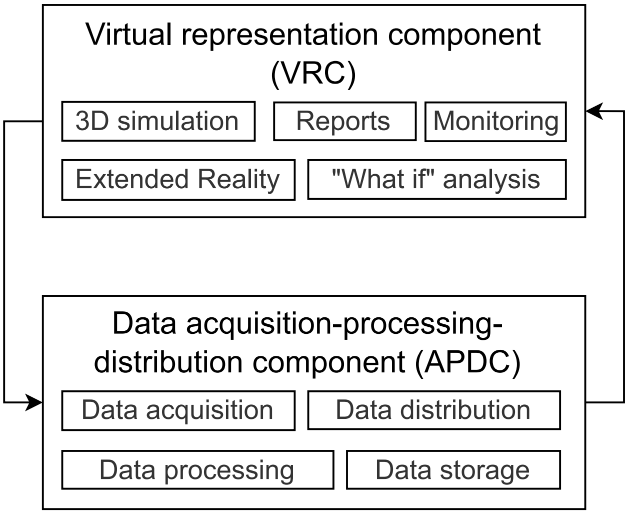


Fig. 2 High-level diagram of the DTE. The two major components are (**i**) the virtual representation component (VRC), and (**ii**) the data acquisition-processing-distribution component (APDC)

external sensors. Data collected from Octoprint are used to build a stateful and detailed DT approach, as well as verify and validate our sensor-data-driven DT approach. The external sensors are used to build a sensor-data-driven DT approach, meant for legacy equipment, with no inherent interconnectivity capabilities. The following sections introduce the subcomponents of the APDC and describe them in detail. A high-level view of the APDC architecture is illustrated in Fig. 3. The left rectangle area illustrates the APDC component which consists of the Octoprint web controller and the Sensor data REST API. The right rectangle area represents the 3D printer module that contains Marlin firmware, and the externally mounted sensors (thermocouple and IR distance sensors). The relationship of data exchange between Octoprint and Marlin is bidirectional, while the user-attached sensors are utilized to monitor the system through the Sensor data REST API.

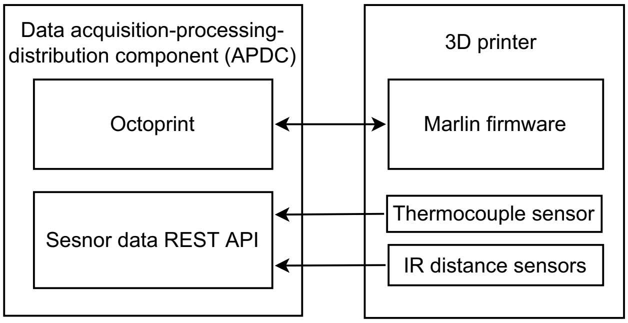


Fig. 3 The data acquisition-processing-distribution component (APDC) of the DTE. This component collects data directly from the printer and the externally mounted sensors

3.1.1 Acquisition-processing-distribution component

To build the APDC, we use Raspberry Pi 3, because of its small size, low cost, and adequate computational power. The Raspberry Pi is a bare printed circuit board (PCB) equipped with all the building components of a typical computer, including the most common I/O, CPU, GPU, RAM, SD card support, and Wi-Fi connectivity to name a few. The operating system of our Raspberry Pi 3 is Raspberry Pi OS, previously known as Raspbian. We developed the sensor API using the Flask[2] framework. Flask is a Python-based microweb framework that excels at rapid prototyping and easy maintenance. The sensor data become available once the sensor API is queried from an external service.

3.1.2 Octoprint web controller

The APDC hosts Octoprint, an open-source extendable web interface used to monitor and control consumer 3D printing machines. Apart from a web interface, Octoprint also provides a REST API that can be queried via hypertext transfer protocol (HTTP) requests. Octoprint provides information regarding (i) job operations, (ii) file operations, and (iii) printer operations. Below, we provide more details on how we combine and utilize parts of these three information categories to collect data that enable the Octoprint data-driven DT module.

- Job operations are used to receive information related to the currently loaded file, its name, its size, and upload date. Job operations are also used to receive and verify the estimated printing time (calculated by the slicing engine), the type of printing filament, the current G-code line being interpreted as a percentage of the total file lines (print completion percentage), the current G-code file line, printing time elapsed, and remaining printing time.
- File operations are used to retrieve a full list of uploaded files, along with their metadata, as well as statistics related to successful and failed prints for each loaded file.
- Printer operations are used for verification and validation (V&V) purposes as well as to retrieve current printer states and historical temperature data for the extruder head and the printing bed.

3.1.3 External sensors

A critical factor in a 3D printing process is to appropriately set up and maintain temperature, both for the printer head and the printing bed. The sensor-data-driven DT approach

[2] https://flask.palletsprojects.com/en/2.0.x/

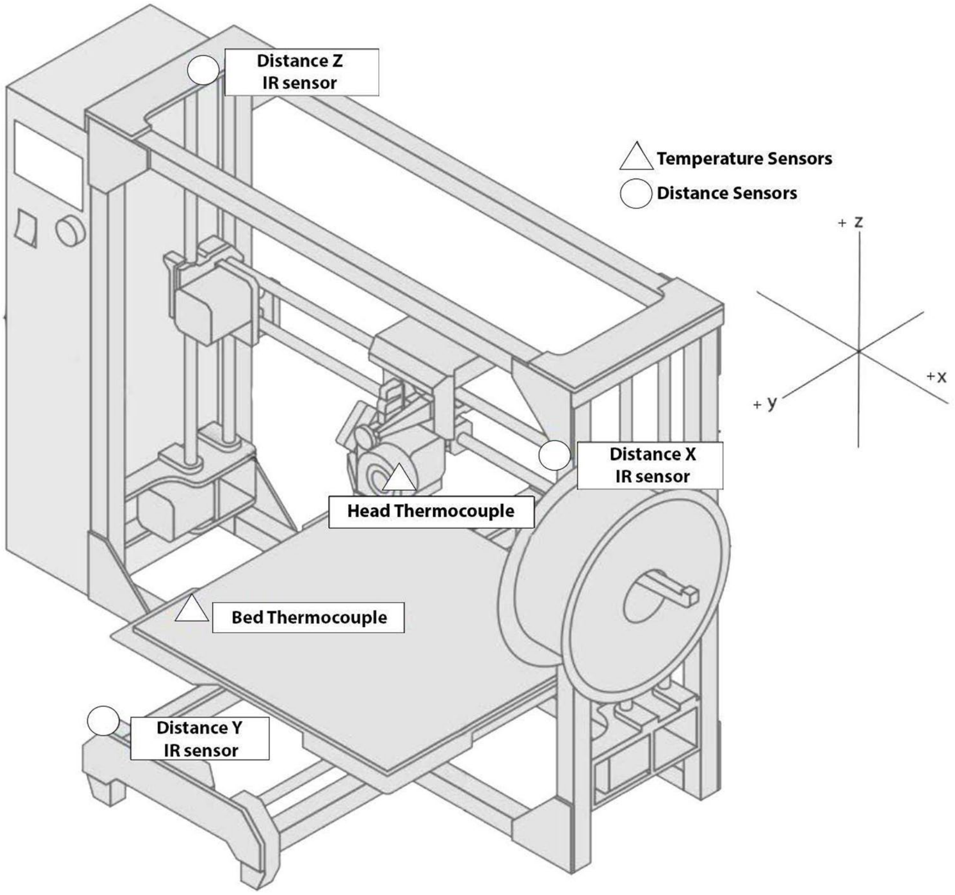


Fig. 4 Taz Workhorse 3D printer model that highlights external sensor installation position

utilized external sensors for the following reasons: (i) to verify and validate the integrity of our architecture and its capability to be applied to legacy equipment, and (ii) to provide an environment for rapid prototyping and reduced data collection downtime. Replacing an internal sensor can be time-consuming. Therefore, we configured the system in a plug-and-play fashion, which simplified replacing a sensor in our pipeline. To receive real-time temperature data, we used the Yocto-Thermocouple USB temperature sensor. The sensor provides two inputs for thermocouples. We utilized one input for the printer head, and the other for the printing bed. The sensor also features internal flash memory, which can be used to store and retrieve historical temperature data. The response time of the sensor is less than 30 ms, with a temperature range between −75 and 260 °C, which makes it a good option for our sensor-data-driven DT. Another vital component of the printing process is the precise movement of the various parts involved. Our FDM AM machine has two different moving parts, namely the extruder head and the printing bed. To monitor the movement of the aforementioned parts, we installed and tested infrared (IR) distance sensors, namely Adafruit VL53L0X Time of Flight Micro-LIDAR. An isometric view of our 3D printer that shows the sensor positions is illustrated in Fig. 4. The temperature sensors are shown with a triangle symbol and the IR distance sensors with a circle. Figure 5 illustrates an image compilation of a real 3D printer that showcases the different positions of the externally mounted sensors.

3.2 Virtual representation component

Unity is a popular 3D real-time development platform, used in various applications ranging from virtual humans [71] and automotive showcases to training agents with Deep Reinforcement Learning [72]. It provides a realistic physics engine, a robust event system, animation system, advanced lighting settings, and the ability to implement custom logic using a programming language (e.g., C#). This platform was utilized to implement the virtual representation component (VRC) of the DTE. A 3D model of the printer was imported into the 3D development platform. First, we fixed orientation issues, changed the pivot point, and grouped the related parts of the printer. For example, all parts of the extruder head were grouped under a new game object called Extruder. Game objects act as containers for components, which accomplish the real functionality. For example, the "Extruder" object is a self-contained modeling construct that allows defining that construct's characteristics, data, behavior, user interface, and animation. Creating a new game object also helps

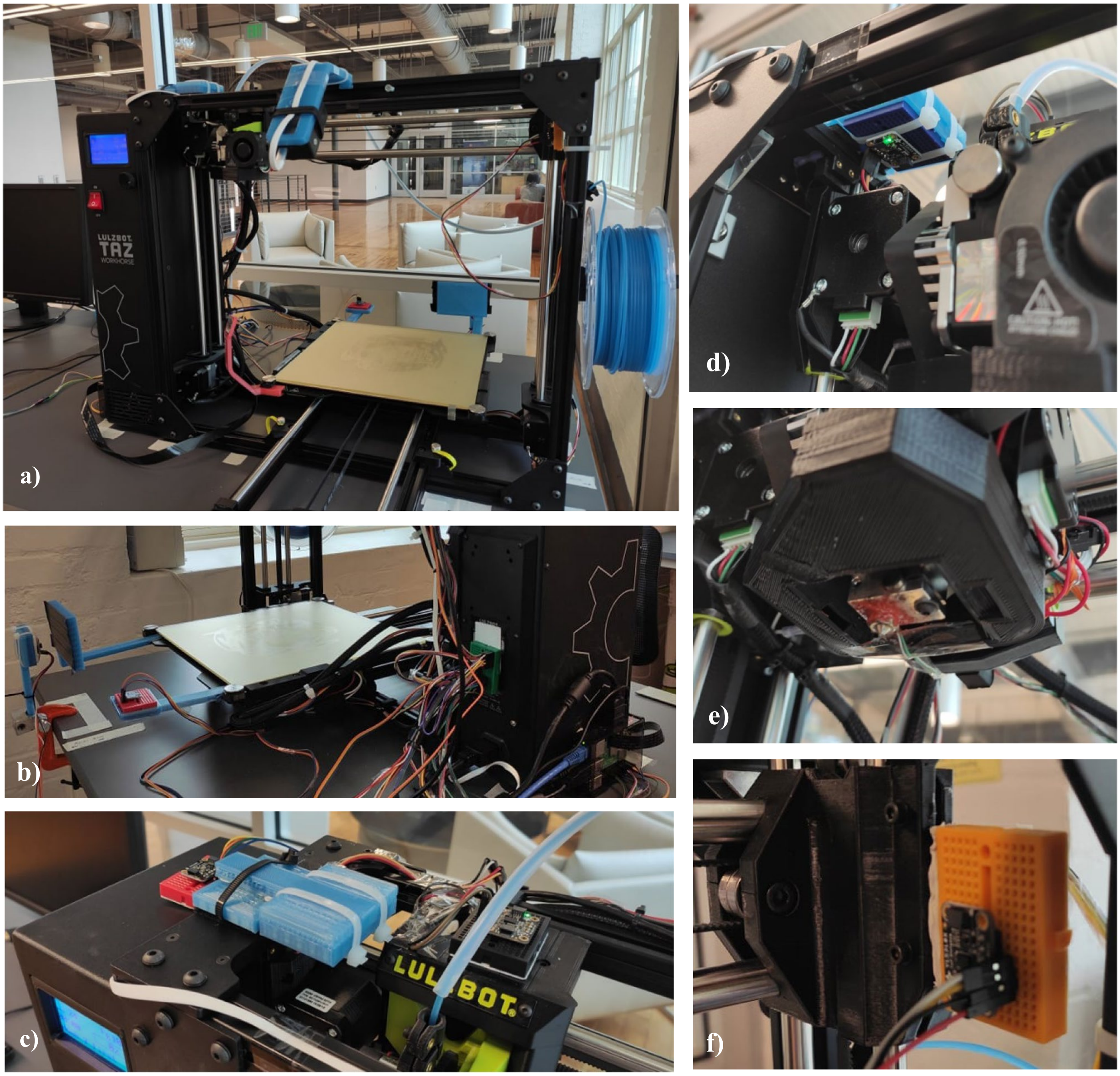


Fig. 5 **a** Printer front view, **b** Printer back view with y distance IR sensor, **c** Printer top view, **d** Z distance sensor from below, **e** Thermocouple sensor attached to the nozzle of the printer head, **f** x distance IR sensor

reorient the model, since the new object's pivot point can be altered without visual modifications to the original model's meshes. Unity's animation system was not only used for visualization purposes but also for V&V of different 3D printer processes. The Octoprint-data-driven approach is stateful, meaning that it keeps track of the current state of the printer, using data directly from its firmware. The sensor-data-driven approach is stateless and utilizes our external sensors to drive the DT, without keeping track of the current state of the printer. The different approaches are explained in Sects. 3.2.1 and 3.2.2.

3.2.1 Octoprint data-driven digital twin

The VRC accesses Octoprint (hosted in the APDC) using an open-source C# wrapper class of the Octoprint REST API. Any class that "wraps" or "encapsulates" the functionality of another class or component is referred to as a wrapper class. This is useful because it provides a level of abstraction from the underlying class or component's implementation. The C# project was compiled into a single Dynamic Link Library (DLL) file using MS Visual Studio. The DLL file is a library that contains code and data that can be used by many

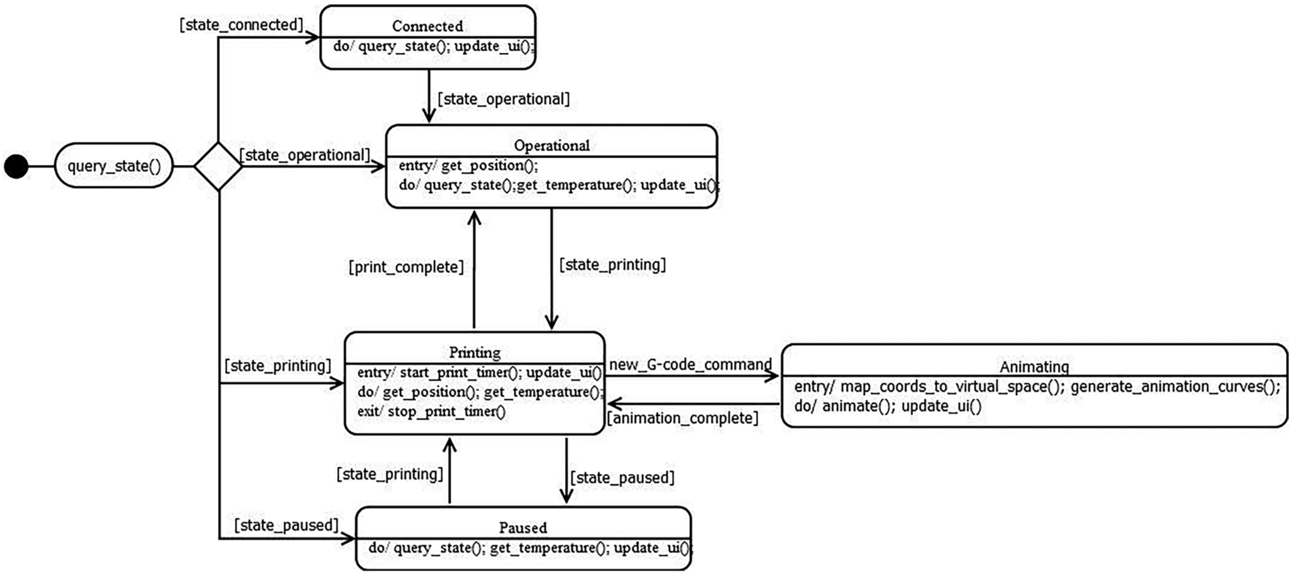
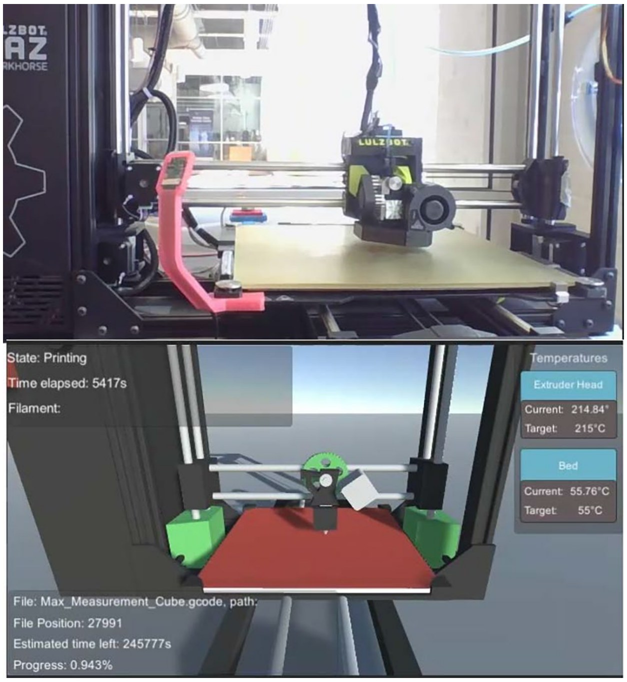

[state_connected]
Connected
do/ query_state(); update_ui();
[state_operational]
query_state()
[state_operational]
Operational
entry/ get_position();
do/ query_state();get_temperature(); update_ui();
[print_complete]
[state_printing]
[state_printing]
Printing
entry/ start_print_timer(); update_ui()
do/ get_position(); get_temperature();
exit/ stop_print_timer()
new_G-code_command
[animation_complete]
Animating
entry/ map_coords_to_virtual_space(); generate_animation_curves();
do/ animate(); update_ui()
[state_printing]
[state_paused]
[state_paused]
Paused
do/ query_state(); get_temperature(); update_ui();

Fig. 6 UML state chart that describes the logic executed in the Octoprint-data-driven Unity DT

programs at the same time. Thus, we imported the DLL file into the Unity project's assets folder to enable access to the Octoprint REST API using high-level C# routines. To achieve synchronization between the LulzBot Workhorse and its stateful DT, a combination of event-based and periodic logic execution is used. Octoprint is queried multiple times per second to determine the printer state and the values of various variables of interest, such as current G-code command, extruder temperature, bed temperature, and estimated time of print completion. The G-code is loaded in a buffer, and the currently executed command is identified based on the feedback from Octoprint. The concept is that the stateful DT of the printer is engaged in some specific kind of action at any given time. The available actions depend on the type of actions the real 3D printer performs, such as idling, printing, translating on XYZ, etc. These actions are referred to as states, in the sense that the DT model is in a "state" where it is printing or idling, just to name a few. In general, the DT will have restrictions on proceeding to the next state rather than being able to switch immediately from one state to another. When a defined event occurs and certain conditions are met, a transition between two states is triggered. For example, an animation can only occur when the DT is printing and not when it is in the idle state. Thus, it should never switch straight from the idle state to the animation state. The options for the next state that our DT model can enter from its current state are referred to as state transitions. Taken into consideration, the set of states, the transitions, and the variables of interest involved with the current state, comprise a finite state machine (FSM). We used a unified modeling language (UML), which is a general-purpose, developmental modeling language that is designed to provide a standard way to depict a system's architecture. Figure 6 illustrates a UML state chart that is used to model the dynamic nature of the different DT states of the FSM. The most important states are the following: "Connected," which is when a successful connection is established with the printer; "Operational," which means that the printer is ready to execute the printing process; "Printing," which means that the printer is currently in the printing process; and "Paused," which means that the print has paused for some reason. The most critical state is the "Printing" state, where the DT remains for most of the print. When a new G-code command is received, the DT enters the "Animating" state. After the physical coordinates are mapped to virtual space, the movement animation curves per axis are generated in real-time before the animation is triggered. Animation curves provide the option to model movement more accurately (e.g., linear, constant, and ease-in–out movement). To simulate realistic animations, we identify the coordinate limits of the physical machine for each axis and translate them to limits within virtual space. Both the physical and the virtual position limits are displayed in Table 1 under columns "Physical limit" and "Virtual limit" respectively. The formula that converts physical to virtual coordinates is shown in (1). It receives

Table 1 Position limits for the physical machine and the unity DT

Type	*Physical limit*	*Sensor val*	*Virtual limit*
X min	−49.9	386	−1.15
X max	295	0	1.15
Y min	−16.64	−20.0	−1.17
Y max	308	301	0.88
Z min	0	351	−1.24
Z max	299	1	1



State: Printing
Time elapsed: 5417s
Filament:

Current: 214.84°
Target: 215°C

Current: 55.76°C
Target: 55°C

File: Max_Measurement_Cube.gcode, path:
File Position: 27991
Estimated time left: 245777s
Progress: 0.943%

Fig. 7 The 3D printer and its stateful DT in unity engine

a coordinate in physical space (coord_p), and, given physical and virtual coordinate minimum and maximum values (min_p, max_p, min_v, max_v), maps the physical coordinate to virtual space (coord_v). This equation is common for all three axes (X, Y, and Z). Physical and virtual limits are denoted by subscripts p and v respectively. Figure 7 illustrates the DT of the AM FDM, as it is displayed in the Unity 3D development platform.

$$\mathbf{coord_v} = \mathbf{min_v} + \frac{(\mathbf{coord_p} - \mathbf{min_p})(\mathbf{max_v} - \mathbf{min_v})}{\mathbf{max_p} - \mathbf{min_p}} \tag{1}$$

3.2.2 Sensor data-driven digital twin

The sensor-data-driven DT utilizes signals from the externally-mounted sensors. The VCR accesses real-time sensor data using our sensor REST API (hosted in the APDC). External sensors can be mounted in several ways depending on various criteria, such as sensor specifications and durability, a possible collision with moving parts, and the availability of mounting space. The external sensor mounting affects the position sensor values. Therefore, the relationship between position sensor values and coordinates is not fixed but varies depending on the installation setting.

To estimate the equivalent physical coordinate value given sensor readings, we devised approximation functions. These functions are calculated by fitting the curves generated by plotting the empirical data of the physical coordinate values over sensor values for each axis. The approximation functions for all axes appear to be linear and are illustrated in Eqs. (2) to (4).

$$\boldsymbol{X}_{\mathbf{physical}} = -0.8935 \times \boldsymbol{X}_{\mathbf{sensor}} + 295 \tag{2}$$

$$\boldsymbol{Y}_{\mathbf{physical}} = 1.0113 \times \boldsymbol{Y}_{\mathbf{sensor}} + 3.587 \tag{3}$$

$$\boldsymbol{Z}_{\mathbf{physical}} = -0.8544 \times \boldsymbol{Z}_{\mathbf{sensor}} + 299.85 \tag{4}$$

where $\boldsymbol{X}_{\mathbf{physical}}$, $\boldsymbol{Y}_{\mathbf{physical}}$, and $\boldsymbol{Z}_{\mathbf{physical}}$ are the physical coordinates for X-axis, Y-axis, and Z-axis respectively, and $\boldsymbol{X}_{\mathbf{sensor}}$, $\boldsymbol{Y}_{\mathbf{sensor}}$, and $\boldsymbol{Z}_{\mathbf{sensor}}$ are the sensor values for each axis.

A raw sensor value is given as input to the corresponding approximation function (i.e., Eqs. (2) to (4)), and the result is a physical coordinate. The physical coordinate is then mapped to virtual space using (1). Consequently, the DT adjusts its moving components, according to the new virtual coordinate. Regarding temperature, no manipulation is required. The temperature value in the DT comes from the temperature sensor reading. The sensor-data-driven DT does not utilize an FSM approach, but is stateless, meaning that the DT always monitors the position and temperature of the 3D printer, without identifying the current state of the machine.

4 Evaluation

In this section, we discuss our findings and evaluate the sensor-data-driven DT in terms of extruder position and temperature, by comparing it to the Octoprint-data-driven DT in static and dynamic conditions. We also evaluate the Octoprint-data-driven DT in terms of print duration against manually collected measurements. Finally, we assess the response time of the DTE. We assume that the true machine values for position and temperature are reported via Octoprint as it has access to the printer firmware containing the instructions that tell the device how to interface with other computer hardware. Our experiments follow the methodology presented in Fig. 8. To ensure the correctness of both DT approaches, we observe the behavior of the DTE through animation over time and compare it with the physical 3D printer. The sensor-driven DT properties match the actual 3D printer properties during the simulation runtime. Therefore, we conclude that twinning between the virtual and the physical asset has been successfully achieved and the DTE is valid. The successful twinning and validation of the DTE can be observed in our online demo[3] video.

[3] https://www.youtube.com/watch?v=Q-HI-AuiK7c

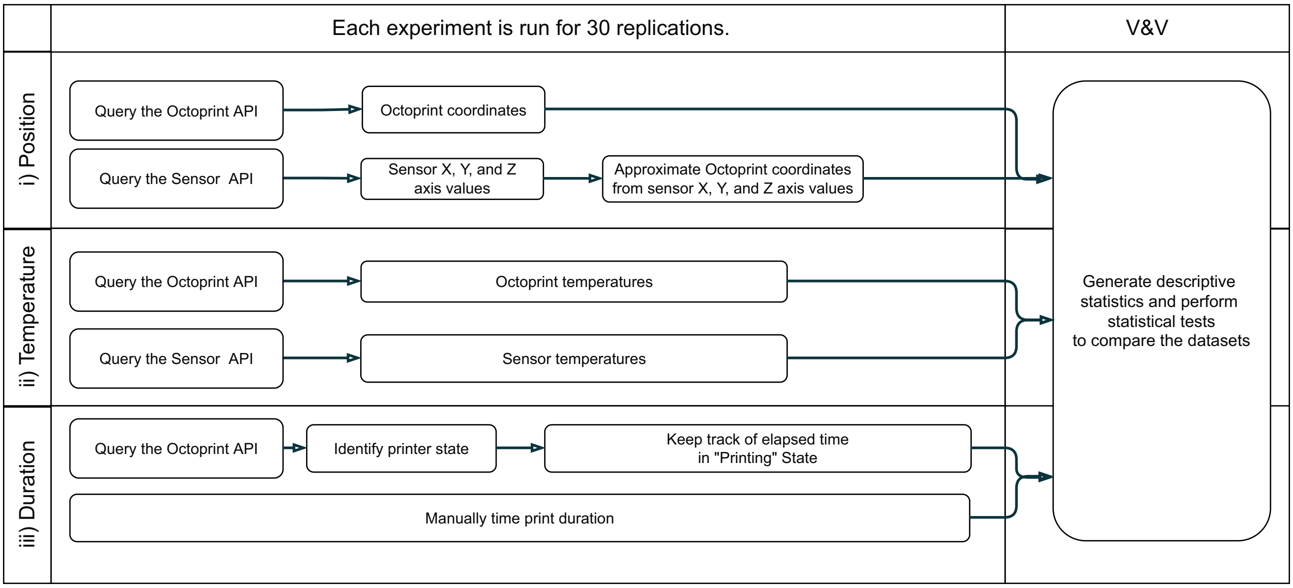

Each experiment is run for 30 replications.
V&V
i) Position
Query the Octoprint API
Octoprint coordinates
Query the Sensor API
Sensor X, Y, and Z axis values
Approximate Octoprint coordinates from sensor X, Y, and Z axis values
ii) Temperature
Query the Octoprint API
Octoprint temperatures
Query the Sensor API
Sensor temperatures
iii) Duration
Query the Octoprint API
Identify printer state
Keep track of elapsed time in "Printing" State
Manually time print duration
Generate descriptive statistics and perform statistical tests to compare the datasets

Fig. 8 Methodology used for DT verification and validation. The DT is evaluated in terms of: **i**) position, **ii**) temperature, and **iii**) print duration

4.1 Position assessment

To evaluate the position-mimicking capability of the sensor-data-driven DT, we compare position data retrieved from our sensor API against data acquired by Octoprint. First, we examine the success of the twinning in static conditions, which is detailed in Sect. 4.1.1. Then, we examine the position data in a transitory regime, explained in Sect. 4.1.2.

4.1.1 Position assessment in static conditions

To evaluate the position-mimicking capability of the sensor-data-driven DT in static conditions, we placed the extruder head and bed on (X, Y, Z) = (50, 53, 19) in physical space. Values are measured in millimeters. We then utilize our sensor data rest API to compare the data from the sensor-data-driven DT against Octoprint data. We collected sensor readings and physical position approximation values in 30 replications. Table 2 presents the mean and standard deviation for sensor values. Table 3 shows the mean and standard deviation for physical coordinate approximations. For the X coordinate, which was initially set to 50, we get a mean of 47.31. For the Y coordinate, which was initially set to 53, we get a mean of 52.94. For the Z coordinate, which was initially set to 19, we get a mean of 22.2. After 30 replications, the DT is on average 2.69 mm off on the X-axis, 0.06 mm off on the Y-axis, and 3.2 mm off on the Z-axis. These differences were expected due to the limitations of the currently installed IR distance sensors, which usually operate over a range of 50 to 1200 mm and can generally achieve between 3 and 12% ranging accuracy depending on the operational conditions and environmental factors (i.e., measurement distance, light conditions, angle). To verify and validate the virtual extruder and bed position and assess the correctness of our code and whether the DT behaves reasonably compared to the real 3D-printer, we applied an Agent-Based Modeling graphical representation technique [73]. Figure 9 presents a graphical representation of the position approximation experiments. Blue circles correspond to the approximated physical coordinate. The solid line is the mean of the measurements, and the dashed line is the true physical coordinate (Octoprint coordinate). As it is observed from the face validity [74] of the graphical representation, we conclude that the DT component positioning behaves sufficiently accurately.

Table 2 Mean and standard deviation of sensor readings for constant physical coordinates ($X = 50$, $Y = 53$, $Z = 19$)

Sensor X-axis		Sensor Y-axis		Sensor Z-axis	
Mean	SD	Mean	SD	Mean	SD
277.2	2.565	48.8	1.827	325	3.5

Table 3 Mean and standard deviation of physical coordinate approximations for constant physical coordinates ($X = 50$, $Y = 53$, $Z = 19$)

X_{physical} approx		Y_{physical} approx		Z_{physical} approx	
Mean	SD	Mean	SD	Mean	SD
47.31	2.292	52.94	1.848	22.2	2.993

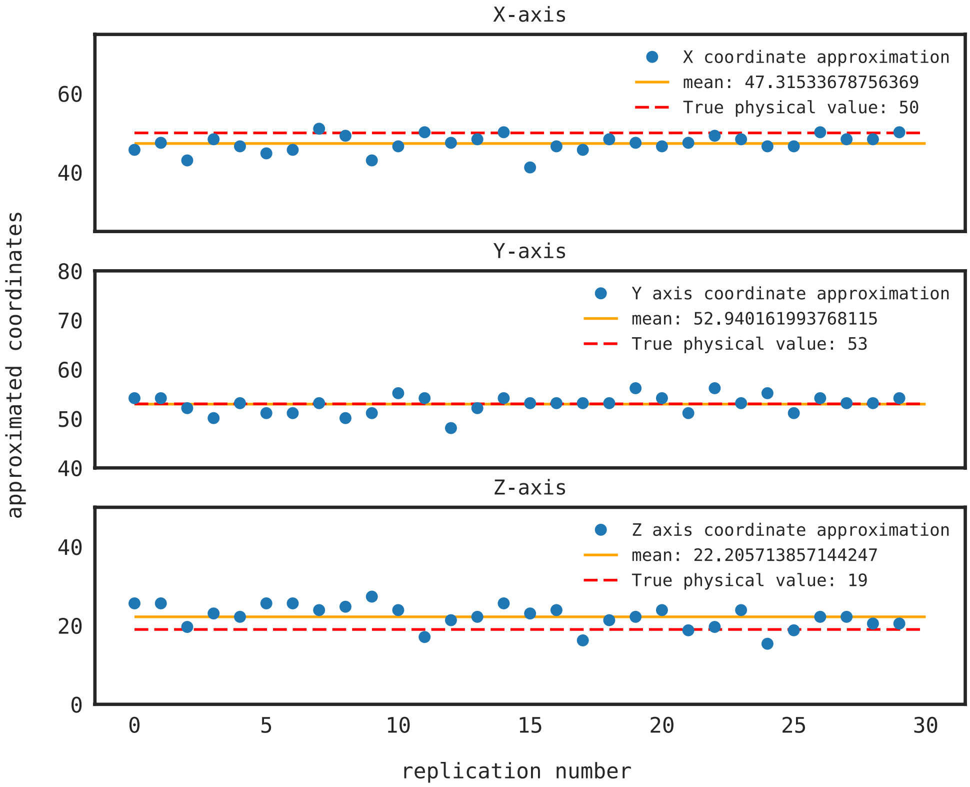


Fig. 9 Physical coordinate approximations from sensor values

4.1.2 Position assessment in a transitory regime

To evaluate the 3D linear movement position-mimicking capability of the sensor-data-driven DT in a transitory regime, we place the extruder head and bed on $(X, Y, Z) = (0, 0, 30)$ in physical space. Values are measured in millimeters. We then issue G-Code commands to increment the position of all axes by two millimeters simultaneously until the axes reach $(X, Y, Z) = (200, 200, 230)$. In essence, we obtain 100 consecutive position samples while recording data from Octoprint and our sensor API. The experiment results are illustrated in Fig. 10. The dashed line corresponds to the true physical coordinate (Octoprint coordinate). The solid line corresponds to readings taken from the sensor-data-driven DT, after the respective axis approximation functions (i.e., Eqs. (2) to (4)) are applied. As can be seen in Fig. 10, the approximated coordinates closely imitate the true physical coordinates. The best-performing IR sensor is the one that records the Y-axis, since it measures the surface with the largest area. The readings from the X-axis IR sensor improve as the extruder head reaches the sensor mounting position. Readings from the Z-axis IR sensor appear to fluctuate the most, which is attributed to unoptimized sensing settings (surface area and material of the measured object, and ambient lighting).

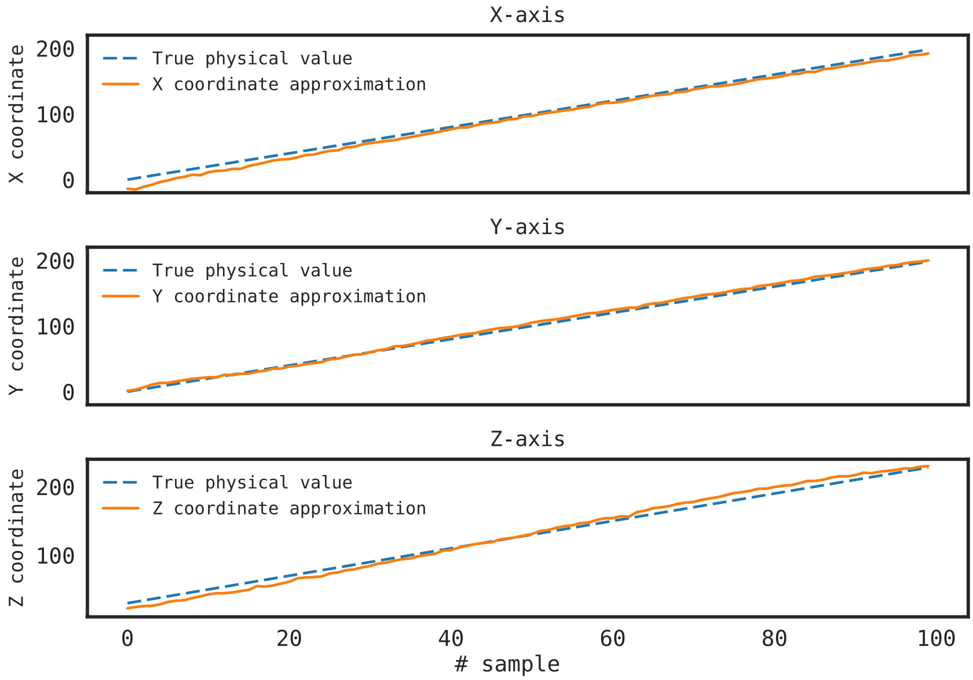
4.2 Temperature assessment

To assess the temperature-mimicking capability of the sensor-data-driven DT, we compare temperature data retrieved from our sensor API against data acquired by Octoprint. First, we design an experiment to assess the temperature readings in the steady state (i.e., when the temperature has stabilized), which is described in Sect. 4.2.1. Subsequently, we conduct another experiment to examine temperature in-between two consecutive temperature tracks, which is discussed in Sect. 4.2.2.

4.2.1 Temperature assessment in steady state

In this experiment, we observe the embedded extruder head sensor readings at the steady state. Polylactic acid (PLA) is one of the most common materials used in 3D printing and our filament material of choice. PLA works well when printing between 180 and 230 °C, which is typically the printer's operation temperature. We performed two experiments, one at 180 °C and another one at 230 °C, with 30 replications for each temperature level. The results are illustrated in Table 4. The manufacturer specifies a ± 2.2 °C error limit for the Yocto-thermocouple sensor. The observed sensor values are within machine specifications. Figure 11

Fig. 10 Octoprint and sensor-DT position coordinates recorded during a 3D linear translation



graphically illustrates the experiment replications. Blue circles correspond to the temperature values acquired by the external sensors. The solid line corresponds to the mean temperature across the experiments, while the dashed line is the true temperature (Octoprint temperature). As it is observed from the face validity of the graphical representation and the mean and standard deviation of Table 4, we conclude that the DT temperature behaves sufficiently accurately.

4.2.2 Temperature assessment in between two consecutive temperature tracks

In this experiment, we initially set the temperature to 180 °C. When the temperature stabilizes, we start recording data from both the Octoprint and the sensor-data-driven DT. Then, we increase the temperature to 230 °C and record the temperature transition. The results of this experiment are illustrated in Fig. 12. The dashed line corresponds to the true temperature (Octoprint temperature), while the solid line corresponds to the temperature retrieved from the sensor-data-driven DT. Notice how the temperature initially rises beyond the 230 °C mark, which is recorded both by Octoprint and our externally mounted temperature sensor. Once a temperature change is issued, the temperature does not directly reach the new temperature value, but oscillates around the new value with the amplitude of the oscillation decreasing over time, until a steady state is reached. Instead of waiting for the temperature to stabilize again, we adjust the temperature to 180 °C. The temperature drop to 180 °C lasts longer compared to the temperature increase to 230 °C. This is an expected outcome, since the temperature rise is achieved by heating the nozzle of the printer, while the temperature drop is achieved by exposure to ambient air and the active cooling from the extruder fans. Once the temperature starts oscillating around the 180 °C mark, we execute the temperature sequence again (i.e., we increase the temperature to 230 °C and then decrease it to 180 °C). As can be seen, the sensor-data-driven DT accurately and timely mimics the temperature of the 3D printer.

Table 4 Mean and standard deviation of the temperature sensor readings at two different temperature levels

180 °C		230 °C	
Mean	SD	Mean	SD
181.82	0.191	230.89	0.092

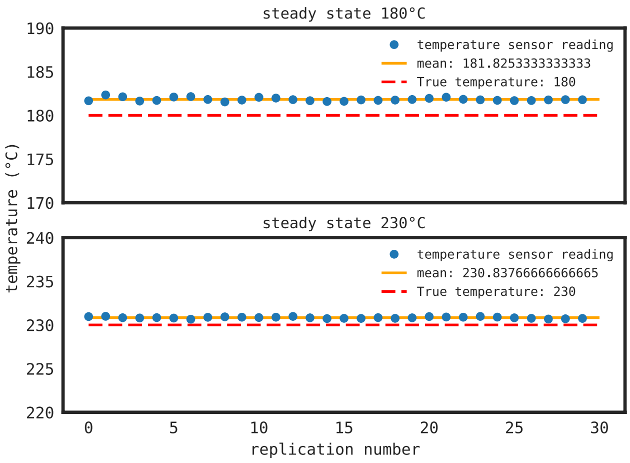


Fig. 11 Temperature readings at two different levels (180 and 230 °C)

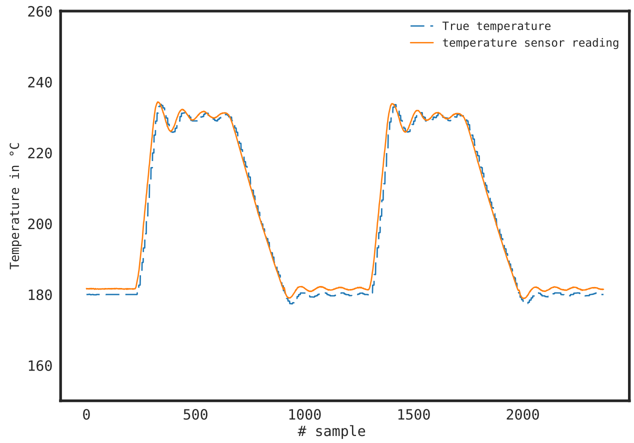


Fig. 12 Temperature readings in-between two consecutive temperature tracks (180 and 230 °C)

4.3 Printer state duration assessment

To confirm that the state durations are accurately reflected in the Octoprint-data-driven DT, we wrote a custom G-code file that describes a print with no temperature adjustments and executed 30 print repetitions. We manually measure the duration of the print by observing the printer and comparing it against the print duration generated in the Octoprint-data-driven DT. Figure 13 shows side-by-side plots, one for each measurement method. Blue circles correspond to the measured values across the replications, and the solid line represents the mean of the measurements. The results, in tabular format, are illustrated in Table 5. The manually measured duration for the print has a mean of 25.7133 s and a standard deviation of 0.157. The Octoprint-data-driven DT duration for the print has a mean of 25.9865 s and a standard deviation of 0.219. Thus, we conclude that the print duration mimicking capability of the DT is sufficiently accurate.

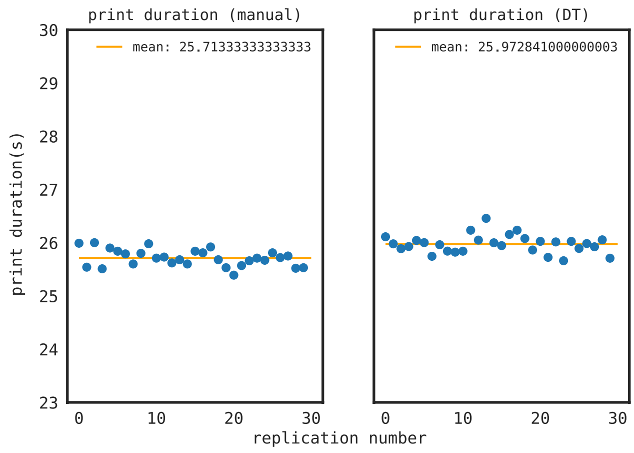


Fig. 13 Print duration measured manually (left) and in the DT (right)

Table 5 Print duration mean and standard deviation. Measurements are taken manually for the machine, and automatically from inside the Unity DT

Print duration (manual)		Print duration (dt)	
Mean	SD	Mean	SD
25.71333 (sec)	0.1571	25.98651 (sec)	0.219

4.4 Extruder speed assessment

The current DTE setup allows the movement speed to be directly estimated from position feedback. If we know two consecutive positions, we can deduce speed by dividing the distance traveled by the elapsed time. However, if feedback is not available, speed estimation becomes a much harder task. For speed to be successfully estimated without feedback from the machine, one needs to know the feed rate setting of the printer, the acceleration setting, and the translation distance. As such, experiments were designed to determine and evaluate the effect of the aforementioned variables on the average speed. The results of our experiments are discussed below.

Initially, the acceleration was set to 500 mm/s^2. Performing 100-mm translations with a feed rate of 3000 the extruder can reach an average speed of 118.34 mm/s. However, when performing 250-mm translations with the same settings, it can reach an average speed of 171.23 mm/s. Given the above, we determine that a higher speed can be achieved when the extruder head travels for a longer distance.

When the translation distance is not large enough, increasing the feed rate does not increase the average speed. The maximum speed of the printer is 300 mm/s; however, this is not always achievable. The extruder must accelerate for a certain amount of time to reach the target speed. For example, reaching a speed of 300 mm/s with an acceleration of 500 mm/s^2 would take 0.6 s. The distance covered within this time is 90 mm. It is past the 90-mm point that the extruder head will travel at the maximum speed. The extruder must also decelerate before stopping. Deceleration requires some distance for which the printer does not operate at maximum speed. The longer the translation, the less acceleration, and deceleration affects the average speed. From the above, we determine that the average speed within a translation has a non-linear relationship with the feed rate for small translation distances and low acceleration settings. To determine how acceleration affects the relationship between feed rate and extruder speed, we set the acceleration to the maximum allowed value of 9000 mm/s^2 and perform the same experiment with the new acceleration setting. The results are presented in Fig. 14. The solid line corresponds to the experiment with acceleration set to 500 mm/s^2 and distance to 100 mm. The dashed line corresponds to acceleration set

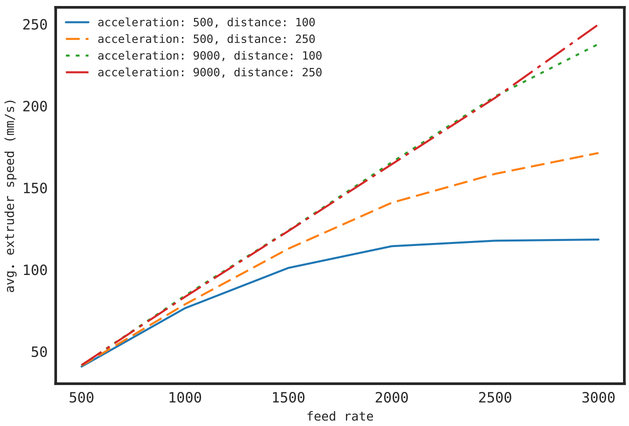


Fig. 14 Impact of acceleration, translation distance, and feed rate on the average speed of the extruder

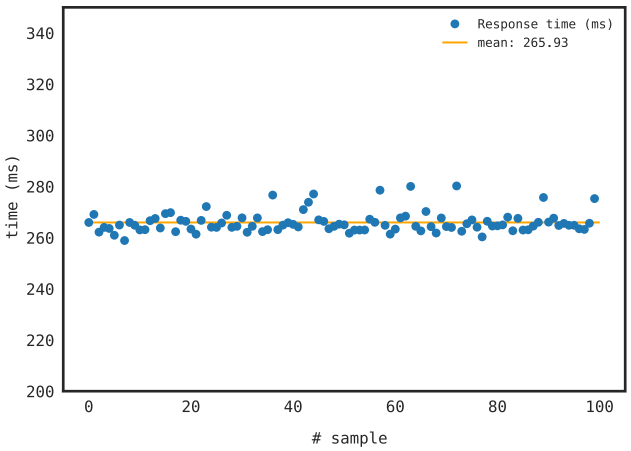


Fig. 15 Response time of the DTE for 100 consecutive samples

to 500 mm/s^2 and distance to 250 mm. The dotted line corresponds to acceleration equal to 9000 mm/s^2 and a distance 100 mm. The dashed-dotted line corresponds to acceleration equal to 9000 mm/s^2 and translation distance 250 mm. We can observe that the relationship between feed rate and average speed approximates a linear curve as translation distance and acceleration increase, with acceleration having the most significant effect.

4.5 Response time assessment

Response time assessment considers measurements of the amount of time it takes for a server to respond to client requests. In our DTE, we define response time as the amount of time that passes between a request from the VRC and a response from the Sensor data REST API of the APDC over a WiFi network. For example, if the VRC requests data to update the virtual extruder temperature of the AM DT, the response time is how long it takes for the sensor data REST API to fulfill that request.

To assess the response time of our DTE, we conduct a response time experiment by performing 100 consecutive requests and calculating the time interval from the moment the request is sent until the response is received. Then, we record our results. For all 100 requests, we receive a successful response and no errors. The response time test results are illustrated in Fig. 15. Blue circles represent the different response time measurements, and the solid line corresponds to the mean response time. The mean response time of our DTE is calculated as 265.94 ms and the standard deviation as 4.08 ms. The mode is 263 ms, and the minimum and maximum response times are calculated as 258.87 ms and 280.21 ms respectively. The response time is adequate to monitor incoming telemetry and could be utilized to detect anomalies in a timely manner. Furthermore, the mean response time of our DTE successfully facilitates the software and hardware requirements to create a near-real-time DT of an AM processes.

5 Conclusions and future work

In this paper, we described the development of a novel digital twin ecosystem that can be used for testing, process monitoring, and remote management of an AM FDM machine in a simulated virtual environment. The developed DTE can capture the overall operation and performance of the FDM machine and therefore could be used for in-process analysis and optimization. To develop a realistic DTE, one needs to study, identify, and model the internal physics that govern the operation of the machine. In our work, we applied a DTE architecture, composed of two key components: the data acquisition-processing-distribution component (APDC) and the virtual-representation component (VRC). Using the APDC and VRC we were able to coordinate and interconnect the components that comprise the Digital Twin Ecosystem (DTE) and achieve continuous synchronization between the physical asset and its digital system, with a mean response time of 265.94 ms. We verified and validated the DTE and its mimicking capabilities using sensor data against internal real-time machine data and we evaluated the twinning of the virtual and physical assets statically and dynamically. We examined and evaluated the 3D linear position of the extruder in static and transitory regime conditions, the temperature in steady-state and in between two consecutive temperature tracks and the impact of acceleration, translation distance, and feed rate on the average speed of the extruder. The developed DTE allows for automatic detection of incoming telemetry in near real-time. In particular, the mean response time of the DTE is 265.94 ms. Notably, the response time that can be achieved is highly reliant on the implementation. The utilized hardware, software, and communication method can be further optimized to reduce response time. In the future, we plan to examine more variables, such as the relationship between feed rate, acceleration, and translation distance, as well as the temperature rise and drop rate. In addition, we plan to explore environmental factors (e.g., atmospheric temperature, humidity, light conditions), as well as the relationship between different filaments, extrusion speeds, and print quality. Our overall future goal is to use this DTE to predict the product quality even before it is printed, reduce the time and cost in experimental testing of optimal printing conditions, and assist in the optimization of part printing and tool path.

IR distance sensors have a higher accuracy the closer they are placed to the measured object. Furthermore, according to the manufacturer, ambient lighting and the material of the measured object affect the quality of the readings. Best readings are obtained from shiny objects, in good lighting conditions. The environmental conditions during the experiments were not fully controlled. More specifically, printer surfaces and room lighting have not been adjusted to optimize sensor reading accuracy. Such modifications will be implemented and studied in future research endeavors.

Another future research direction is to augment the developed DT with extended reality (XR) capabilities. The transformation is straightforward since Unity 3D real-time development platform offers extensive XR support. An XR version of the 3D printer's DT has the potential to serve remote educational purposes. Schools, universities, and SMEs could benefit from such a system. For example, a remote augmented reality DT can be utilized when the environment is not suitable for machinery, or there are financial constraints. Furthermore, the interconnectivity capabilities of modern head-mounted displays (HMDs) allow for collaborative work between individuals and teams.

Acknowledgements The authors would like to acknowledge John Osho and Anna Hyre for assisting with the hardware setup and programming.

Funding This work was partially supported by the National Institute of Standards and Technology (NIST), Grant 70NANB18H267, and the Interdisciplinary Center of Advanced Manufacturing Systems (ICAMS) funded by the US Army, Grant W52PLJ-20–9-3045.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Conflict of interest The authors declare no competing interests.

References

1. Costello K, Omale G (2019) gartner.com. Stamford Conn. Available: https://www.gartner.com/en/newsroom/press-releases/2019-02-20-gartner-survey-reveals-digital-twins-are-entering-mai. [Accessed Oct 2021]
2. Mykoniatis K, Harris GA (2021) A digital twin emulator of a modular production system using a data-driven hybrid modeling and simulation approach. J Intell Manuf 1–13.
3. Grieves M, Vickers J (2017) Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems. Transdiscipl Perspect Complex Syst Springer 85–113
4. MarketsandMarkets (2020) Digital Twin Market by Technology, type (product, process, and system), application (predictive maintenance, and others), industry (aerospace & defense, automotive & transportation, healthcare, and others), and geography - global forecast to 2026
5. Tiwari N, Kaur MJ, Mishra VP (2021) Architecture and use cases of digital twins towards smart manufacturing. Int J Publ Sect Perform Manag 8(3):264–272
6. Lehner D, Pfeiffer J, Tinsel EF, Strljic MM, Sint S, Vierhauser M, Wortmann A, Wimmer M (2021) Digital twin platforms: requirements, capabilities, and future prospects. IEEE Softw 01:0–0
7. Conde J, Munoz-Arcentales A, Alonso A, Lopez-Pernas S, Salvachua J (2021) Modeling digital twin data and architecture: a building guide with fiware as enabling technology. IEEE Internet Comput 01:1–1
8. Lehner D, Sint S, Vierhauser M, Narzt W, Wimmer M (2021) AML4DT: a model-driven framework for developing and maintaining digital twins with automationML. In 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Vasteras, Sweden
9. Kostromin R, Feoktistov A, Voskoboinikov M (2021) Service-oriented tools for automating digital twin development
10. Hyre A, Harris G, Osho J, Pantelidakis M, Mykoniatis K, Liu J (2022) Digital twins: representation, replication, reality, and relational (4Rs). Manuf Lette 31:20–23
11. Tao F, Zhang H, Liu A, Nee AY (2018) Digital twin in industry: State-of-the-art. IEEE Trans Industr Inf 15(4):2405–2415
12. Uhlenkamp JF, Hribernik K, Wellsandt S, Thoben KD (2019) Digital twin applications: a first systemization of their dimensions. In 2019 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC) , Valbonne Sophia-Antipolis, France
13. Pronost G, Mayer F, Marche B, Camargo M, Dupont L (2021) Towards a framework for the classification of digital twins and their applications. In 2021 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Cardiff, United Kingdom
14. Mykoniatis K (2020) A real-time condition monitoring and maintenance management system for low voltage industrial motors using internet-of-things. Procedia Manuf 42:450–456
15. Angelopoulou A, Mykoniatis K, Boyapati NR (2020) Industry 4.0: The use of simulation for human reliability assessment. Procedia Manuf 42:296–301
16. 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
17. West TD, Blackburn M (2017) Is digital thread/digital twin affordable? A systemic assessment of the cost of DoD's latest manhattan project. Procedia Comput Sci 114:47–56
18. Uhlemann TH-J, Schock C, Lehmann C, Freiberger S, Steinhilper R (2017) The digital twin: demonstrating the potential of real time data acquisition in production systems. Procedia Manuf 9:113–120

19. Liu Z, Meyendorf N, Mrad N (2018) The role of data fusion in predictive maintenance using digital twin. AIP Conf Proc
20. Qi Q, Tao F (2018) Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison. Ieee Access 6:3585–3593
21. Uhlemann TH-J, Lehmann C, Steinhilper R (2017) The digital twin: Realizing the cyber-physical production system for industry 4.0. Procedia Cirp 61:335–340
22. Aivaliotis P, Georgoulias K, Alexopoulos K (2019) Using digital twin for maintenance applications in manufacturing: state of the art and gap analysis. In 2019 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), Valbonne Sophia-Antipolis, France
23. Xu Y, Sun Y, Liu X, Zheng Y (2019) A digital-twin-assisted fault diagnosis using deep transfer learning. IEEE Access 7:19990–19999
24. Löcklin A, Jung T, Jazdi N, Ruppert T, Weyrich M (2021) Architecture of a human-digital twin as common interface for operator 4.0 applications. Procedia CIRP 104:458–463
25. Holzwarth V, Hirt C, Gisler J, Kunz A (2021) Virtual reality extension for digital twins of machine tools
26. Wang Y, Wang X, Tao F, Liu A (2021) Digital twin-driven complexity management in intelligent manufacturing. Digital Twin 1:9
27. Zhang H, Liu Q, Chen X, Zhang D, Leng J (2017) A digital twin-based approach for designing and multi-objective optimization of hollow glass production line. IEEE Access 5:26901–26911
28. Wong KV, Hernandez A (2012) A review of additive manufacturing. Int Sch Res Notices 2012
29. Daminabo SC, Goel S, Grammatikos SA, Nezhad HY, Thakur VK (2020) Fused deposition modeling-based additive manufacturing (3D printing): techniques for polymer material systems. Mater Today Chem 16:100248
30. Guo H, Lv R, Bai S (2019) Recent advances on 3D printing graphene-based composites. Nano Materials Science 1:101–115
31. Parandoush P, Lin D (2017) A review on additive manufacturing of polymer-fiber composites. Compos Struct 182:36–53
32. Rao PK, Liu JP, Roberson D, Kong ZJ, Williams C (2015) Online real-time quality monitoring in additive manufacturing processes using heterogeneous sensors. J Manuf Sci Eng 137
33. Rao P, Liu J, Roberson D, Kong Z (2015) Sensor-based online process fault detection in additive manufacturing. In ASME 2015 International Manufacturing Science and Engineering Conference, Charlotte, North Carolina, USA
34. Kurfess TR, Saldana C, Saleeby K, Dezfouli MP (2020) A review of modern communication technologies for digital manufacturing processes in Industry 4.0. J Manuf Sci Eng 142:110815
35. Vijayaraghavan A, Sobel W, Fox A, Dornfeld D, Warndorf P (2008) Improving machine tool interoperability using standardized interface protocols: MT connect
36. Rosen R, Von Wichert G, Lo G, Bettenhausen KD (2015) About the importance of autonomy and digital twins for the future of manufacturing. IFAC-PapersOnLine 48:567–572
37. Bartsch K, Pettke A, Hubert A, Lakamper J, Lange F (2021) On the digital twin application and the role of artificial intelligence in additive manufacturing: a systematic review. J Phys Mater
38. Montoya-Zapata D, Rodriguez JM, Moreno A, Posada J, Ruiz-Salguero O (2021) 2D linear finite element simulation of laser metal heating for digital twins. Int J Simul Multi Design Optim 12:11
39. Shrivastava A, Mukherjee S, Chakraborty SS (2021) Addressing the challenges in remanufacturing by laser-based material deposition techniques. Opt Laser Technol 144:107404
40. Stavropoulos P, Papacharalampopoulos A, Michail CK, Chryssolouris G (2021) Robust additive manufacturing performance through a control oriented digital twin. Metals 11:708
41. Gunasegaram D, Murphy A, Matthews M, DebRoy T (2021) The case for digital twins in metal additive manufacturing. J Phys Mater 4:040401
42. Debroy T, Zhang W, Turner J, Babu SS (2017) Building digital twins of 3D printing machines. Scripta Mater 135:119–124
43. Knapp G, Mukherjee T, Zuback J, Wei H, Palmer T, De A, DebRoy T (2017) Building blocks for a digital twin of additive manufacturing. Acta Mater 135:390–399
44. Sieber I, Thelen R, Gengenbach U (2021) Enhancement of high-resolution 3D inkjet-printing of optical freeform surfaces using digital twins. Micromachines 12:35
45. Xi Z (2021) Model predictive control of melt pool size for the laser powder bed fusion process under process uncertainty. ASCE-ASME J Risk Uncertainty Eng Syst Part B Mech Eng 8:011103
46. Azad MA, Olawuni D, Kimbell G, Badruddoza AZM, Hossain M, Sultana T (2020) Polymers for extrusion-based 3D printing of pharmaceuticals: a holistic materials–process perspective. Pharmaceutics 12(2):124
47. Shaqour B, Abuabiah M, Abdel-Fattah S, Juaidi A, Abdallah R, Abuzaina W, Qarout M, Verleije B, Cos P (2021) Gaining a better understanding of the extrusion process in fused filament fabrication 3D printing: a review. Int J Adv Manuf Technol 1–13
48. Xia H, Lu J, Dabiri S, Tryggvason G (2018) Fully resolved numerical simulations of fused deposition modeling. Part I: fluid flow. Rapid Prototyp J 24(2):463–476
49. Xia H, Lu J, Tryggvason G (2018) Fully resolved numerical simulations of fused deposition modeling. Part II – solidification, residual stresses and modeling of the nozzle. Rapid Prototyp J 24(6):973–987
50. Courter BJ, Savane V, Hansen CJ (2017) Finite element simulation of the fused deposition modelling process
51. Serdeczny MP, Comminal R, Pedersen DB, Spangenberg J (2019) Numerical simulations of the mesostructure formation in material extrusion additive manufacturing. Addit Manuf 28:419–429
52. Gaikwad A, Yavari R, Montazeri M, Cole K, Bian L, Rao P (2020) Toward the digital twin of additive manufacturing: Integrating thermal simulations, sensing, and analytics to detect process faults. IISE Trans 52:1204–1217
53. Scheffel RM, Frohlich AA, Silvestri M (2021) Automated fault detection for additive manufacturing using vibration sensors. Int J Comput Integr Manuf 34:500–514
54. Wei H, Mukherjee T, Zhang W, Zuback J, Knapp G, De A, DebRoy T (2021) Mechanistic models for additive manufacturing of metallic components. Prog Mater Sci 116:100703
55. Zou R, Liang X, Chen Q, Wang M, Zaghloul MA, Lan H, Buric MP, Ohodnicki PR, Chorpening B, To AC (2020) A digital twin approach to study additive manufacturing processing using embedded optical fiber sensors and numerical modeling. J Lightwave Technol 38:6402–6411
56. Alizadeh M, Esfahani MN, Tian W, Ma J (2020) Data-driven energy efficiency and part geometric accuracy modeling and optimization of green fused filament fabrication processes. J Mech Des 142:4
57. Redelinghuys A, Basson AH, Kruger K (2019) A six-layer architecture for the digital twin: a manufacturing case study implementation. J Intell Manuf 1–20
58. Karanjkar N, Joglekar A, Mohanty S, Prabhu V, Raghunath D, Sundaresan R (2018) "Digital twin for energy optimization in an SMT-PCB assembly line. IEEE Int Conf Internet Things Intell Syst (IOTAIS) 2018:85–89
59. Chhetri SR, Al Faruque AM (2020) IoT-enabled living digital twin modeling. Data-Driven Modeling of Cyber-Physical Systems using Side-Channel Analysis 155–182
60. Bagheri B, Lee J (2015) Big future for cyber-physical manufacturing systems. Des World 23

61. Kuts V, Otto T, Tahemaa T, Bondarenko Y (2019) Digital twin based synchronised control and simulation of the industrial robotic cell using virtual reality. J Mach Eng 19
62. Lacomblez M, Jeanne B, Havard V, Baudry D (2018) Co-simulation architecture between a digital twin and a virtual reality environment in an industrial context. In Advances in Manufacturing Technology XXXII: Proceedings of the 16th International Conference on Manufacturing Research, incorporating the 33rd National Conference on Manufacturing Research, September 11--13, 2018, University of Skovde, Sweden
63. Meier N, Muller-Polyzou R, Brach L, Georgiadis A (2021) Digital twin support for laser-based assembly assistance. Procedia CIRP 99:460–465
64. Matulis M, Harvey C (2021) A robot arm digital twin utilising reinforcement learning. Comput Graph 95:106–114
65. Hassel T, Hofmann O (2020) Reinforcement learning of robot behavior based on a digital twin. ICPRAM 381–386
66. Wang Z, Han K, Tiwari P (2021) Digital twin simulation of connected and automated vehicles with the unity game engine. In 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI) 1–4
67. White G, Zink A, Codeca L, Clarke S (2021) A digital twin smart city for citizen feedback. Cities 110:103064
68. Laaki H, Miche Y, Tammi K (2019) Prototyping a digital twin for real time remote control over mobile networks: application of remote surgery. IEEE Access 7:20325–20336
69. Choong YYC, Tan HW, Patel DC, Choong WTN, Chen CH, Low HY, Chua CK (2020) The global rise of 3D printing during the COVID-19pandemic. Nat Rev Mater 5(9):637–639
70. Tarfaoui M, Nachtane M, Goda I, Qureshi Y, Benyahia H (2020) 3D printing to support the shortage in personal protective equipment caused by COVID-19 pandemic. Materials 13(15):3339
71. Mykoniatis K, Angelopoulou A, Proctor MD, Karwowski W (2014) Virtual humans for interpersonal and communication skills' training in crime investigations. In International Conference on Virtual, Augmented and Mixed Reality
72. Sun L, Zhai J, Qin W (2019) Crowd navigation in an unknown and dynamic environment based on deep reinforcement learning. IEEE Access 7:109544–10955
73. Xiang X, Kennedy R, Madey G, Cabaniss S (2055) Verification and validation of agent-based scientific simulation models. In In Agent-directed Simulation Conference
74. Naylor TH, Finger JM (1967) Verification of computer simulation models. Manag Sci 14(2):B-92

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.