Resource Requirements of an Edge-based Digital Twin Service: An Experimental Study

Virtual Reality & Intelligent Hardware

• Article •

December 2022 Vol. 4 No. 6: 506–520
10.1016/j.vrih.2022.05.005

Resource requirements of an edge-based digital twin service: an experimental study

Federico MUNGARI[1], Milan GROSHEV[2], Carla Fabiana CHIASSERINI[1*]

1. *Department of Electronics and Telecommunications, Politecnico di Torino, Turin* 10129*, Italy;*
2. *Departamento de Ingenieria Telematica, Universidad Carlos III de Madrid, Madrid* 28911*, Spain*

Received 3 March 2022; **Revised** 14 April 2022; **Accepted** 24 May 2022

Abstract: Digital Twin (DT) is a pivotal application under the industrial digital transformation envisaged by the fourth industrial revolution (Industry 4.0). DT defines intelligent and real-time faithful reflections of physical entities such as industrial robots, thus allowing their remote control. Relying on the latest advances in Information and Communication Technologies (ICT), namely Network Function Virtualization (NFV) and Edge-computing, DT can be deployed as an on-demand service in the factories close proximity and offered leveraging radio access technologies. However, with the purpose of achieving the well-known scalability, flexibility, availability and performance guarantees benefits foreseen by the latest ICT, it is steadily required to experimentally profile and assess DT as a Service (DTaaS) solutions. Moreover, the dependencies between the resources claimed by the service and the relative demand and work loads require to be investigated. In this work, an Edge-based Digital Twin solution for remote control of robotic arms is deployed in an experimental testbed where, in compliance with the NFV paradigm, the service has been segmented in virtual network functions. Our research has primarily the objective to evaluate the entanglement among overall service performance and VNFs resource requirements, and the number of robots consuming the service varies. Experimental profiles show the most critical DT features to be the inverse kinematics and trajectory computations. Moreover, the same analysis has been carried out as a function of the industrial processes, namely based on the commands imposed on the robots, and particularly of their ion-level, resulting in a novel trade-off between computing and time resources requirements and trajectory guarantees. The derived results provide crucial insights for the design of network service scaling and resource orchestration frameworks dealing with DTaaS applications. Finally, we empirically prove LTE shortage to accommodate the minimum DT latency requirements.

Keywords: Digital twin; Edge; NFV; Testbed; Performance evaluation

Supported by the EU through the 5Growth project (856709); the UNICO 5G I+D 6G-EDGEDT-01/02; the Spanish Ministry of Economic Affairs and Digital Transformation; NPRP-S 13th Cycle (NPRP13S-0205-200265) from the Qatar National Research Fund (a member of Qatar Foundation).

Citation: Federico MUNGARI, Milan GROSHEV, Carla Fabiana CHIASSERINI. Resource requirements of an edge-based digital twin service: an experimental study. Virtual Reality & Intelligent Hardware, 2022, 4(6): 506–520

***Corresponding author**, carla.chiasserini@polito.it

1 Introduction

The Industry 4.0 revolution, also referred to as Industry of the Future, has emerged as the new paradigm shift in the industrial manufacturing systems. New manufacturing approaches have been developed to overcome the limitations of the effectiveness and repeatability of manpower employment[1]. In particular, industrial robots, e.g., robotic arms, and automation are increasingly being leveraged to replace human effort for repetitive tasks towards workerless plants, leaving efficient production planning and operation monitoring to manufacturing domain experts[2]. In addition, the Industry of the Future envisions the digital transformation of modern industries towards service-oriented industries. Manufacturing processes will thus become more flexible, safe, productive, cost-effective and smartly coordinated in unprecedented ways by being offered as on-demand services[3]. Industry 4.0 therefore requires digitized knowledge[4] about industrial machinery to open production environments to smart manufacturing services.

In this regard, the Digital Twin (DT) undoubtedly stands out as a cornerstone service, as it enables the desired digital transformation. DT is a computerized model that replicates the state and evolution of a physical entity or process, with which a continuous interaction is established[5]. Hence, a DT is the living, smart and evolving virtual counterpart of a physical system such as robotic arms. The advantages brought by this breakthrough concept are innumerable. Depending on the operated service, the manufacturing DT can be defined as [6]: (1) Monitoring DT when the virtual replica provides information about the evolving status of its physical counterpart, e.g., [7,8], thus enabling visualization and fault detection capabilities; (2) Simulation DT when simulation or predictive tools are run in the virtual world by exploiting 3D or machine learning-based models, e.g., [9–11]; (3) Operational DT when the interaction with the digital twin via frictionless interfaces is harnessed by domain experts to carry out remote motion control (i.e., the use-case evaluated here) and data aware management of the abstracted target, thus disrupting the interaction with industrial machinery.

To provide the twinned replica with reliable information seamlessly and in real-time, the DT service must be integrated with the latest advances in information and communication technology[12]. Edge-computing, Network Function Virtualization (NFV) and 5G aim to meet the requirements of Industry 4.0 and DT applications in terms of low latency, high reliability and bandwidth, scalability and connection density. Edge computing allows DT applications to offload their compute-intensive logic (e.g., object detection, motion planning) towards the edge of the network. This enables DT to be executed in close proximity to the factories offering low-latency, high bandwidth and reliability, and context aware services[13,14] while enabling the deployment of cost-effective robots. In turn, the NFV paradigm enables the functional partitioning of the DT application into a wide range of re-configurable Virtual Network Functions (VNFs) deployable on top of general-purpose hardware. Finally, Industry 4.0 claims 5G as a key enabler to fulfill the real-time Key Performance Indicators (KPIs) set by the DT achieving wire-like performance. The 5G networks are architectured to simultaneously support different types of service profiles in a shared infrastructure such as enhanced Mobile BroadBand (eMBB), massive Machine Type Communication (mMTC), and Ultra-Reliable Low Latency Communication (URLLC).

Although Edge computing is envisioned as a technology to run on-demand services, in addition to providing high flexibility, enhanced scalability, and agile management, an effective characterization of the provided services is required by automated resource scaling and orchestration frameworks[15]. Assessing the resource profile of the service DT and exploring its dependencies are critical to reap the benefits of automated service resizing tools, i.e., resource usage savings and performance guarantees even in correspondence to service load surge. Accordingly, this research aims to carry out an experimental analysis of a DT as a Service (DTaaS) solution for robotic arms remote control that integrates the computing, virtualization and wireless technologies. To do so, a DT solution for robotic arms is decomposed into VNFs and deployed over an experimental

testbed that integrates Software Defined Radio (SDR) link solution for 3GPP LTE. The implementation of Edge-based DT in a real testbed allows us to observe trade-offs across different deployment configurations. Specifically, the main contributions of this work are the following: (1) the edge-based, virtualized implementation of a DTaaS prototype for robotic arm; (2) an insight about the relationship between the resource requirements and the service demand load; (3) the study of the impact of the application and the workload of the robotic arm on the resource profile of the DT service.

The rest of the paper is organized as follows. Section 2 reviews the related works and motivation. Section 3 firstly introduces the Edge-based Digital Twin proposed as a DTaaS and then an overview about the evaluated command abstraction-levels imposed on the robots. The Edge-based DT solution has been deployed on the testbed presented in Section 4. Finally, Section 5 presents the service resource profile and performance evaluation results, whose main insights are discussed in Section 6 along with future plans.

2 Background and motivation

The operational Digital Twin primarily finds application in manufacturing robotic applications, pushed by growing interest from the industrial verticals. Thanks to the latest advances on Cloud first, and Edge-computing then, as well as NFV and 5G, the manufacturing sector can benefit from the DT for the service-oriented industries attainment. DT can be provided as a network service, which entails its flexible decomposition into Virtual Network Functions (VNFs) deployed on commodity hardware. Edge computing empowers the deployment of lightweight and cost-effective robots by offloading the intelligence in the Edge, compute and storage capacity is leveraged to support manufacturing applications[14]. Moreover, NFV and Edge-computing concepts are natively envisaged by 5G, which is embraced by Industry 4.0 as the first communication technology to simultaneously meet: (1) flexibility, cost-effective deployment and maintenance, mobility support and worker safety[16] peculiar to wireless technologies, as well as (2) stringent real-time guarantees characteristic of such wired networks as Fieldbus[17]. These potential benefits come in addition to the advantages that the use of industrial robots brings, namely, large workspaces, flexibility, and high degree of freedom, which is why their applications are so numerous and continuously increasing.

It is therefore not surprising that manufacturing DT and DT -enabled robotics have recently attracted much attention in both industry and academia. The study in [18] takes a data-driven approach to automate smart manufacturing cell systems, while [19] proposes a reference model for s designing smart assembly processes and discusses a corresponding DT-based application framework. The work in [20], on the other hand, evaluates the benefits of DT-based collaborative human-robot systems for industrial assembly. [21] relies on the Cloud to offload a complex and novel simultaneous localization and mapping framework, while [22] presents a Cloud-based system for remote configuration of industrial robots. Computationally critical features of an industrial robotics surface blending system are detected and offloaded to the Edge in [23], Edge-based DT solutions for automation and metal additive manufacturing systems are presented in, respectively, [24] and [25]. An Edge-based prototype of a DT that integrates Edge computing, NFV and 5G connectivity is presented in [26] and [6] where the authors investigate different offloading strategies and wireless technologies (e.g., Wi-FI, LTE, 5G) to support the Edge Robotic Digital Twin.

Nevertheless, the works in [18–20]do not discuss the role and the benefits introduced by the latest advances in ICT, i.e., Edge computing, NFV ad 5G. Most of the recent works[21,22] focus on Cloud deployments benefits, which however is prone to network dynamics and is not viable for time-sensitive operations. The studies in [23–25], instead, rely on the Edge for monitoring and collective robot learning DT applications rather then robot remote control. Finally, the works in [26] and [6] evaluate a robotic Edge-based DT solution through prototype implementation, but, unlike our study, limit their analysis to scenarios with a single robot.

In summary, differently from previous work, we present the experimental profile and performance assessment of a robotic remote control Edge-based DT solution deployed as a service, hence decomposed into VNFs, and offered on a small testbed setup integrating a SDR link solution for 3GPP LTE to simulated robotic arms. This work does not envisage an explicit industrial field or application, rather it is easily generalizable and extensible to all industrial applications that can benefit from the remote control and motion planning of industrial robotic arms. So doing, our study mainly contributes to identifying the most critical and computationally intensive network modules of a robotic DTaaS solution, and it provides novel insights on which parameters influence the overall service requirements, such as the robotic command abstraction level. To the best of our knowledge, no existing work has addressed this aspect. The findings presented here can bring significant benefits to agile and automated DT services orchestration frameworks, which may enable real-time service resize to realize network service cost-effectiveness and performance guarantees.

3 Edge-based digital twin

In this work, we focus on an Edge-based DT solution designed for the remote control of a robotic arms factory. It is worth stressing that this use case is of high practical relevance, as manufacturing processes are manageable and optimizable devising production plans to be remotely imposed on industrial machinery, e.g., robotic arms. In this section, we first introduce the assessed Edge-based DT solution in Section 3.1, then we give an overview of the eligible commands and the involved VNFs composing the service in Section 3.2.

3.1 The Edge-based digital twin service solution

The DTaaS solution must be designed according to the NFV paradigm. To this end, while configuring the DTaaS solution, we first identify the largest possible extent of VNFs in which a DT service can be decomposed, in order to experimentally explore the service requirements with the highest possible granularity and, hence, determine the critical service components and profile their behavior in different scenarios. The assessed Edge-based DT solution is reported in Figure 1 and includes six VNFs, as described below.

Driver VNF: this module runs low-level functions that directly interface with the robot hardware and sensors. The Driver VNF is in charge of: (1) reading data from the robot sensors and sending them to the rest of the VNFs, and (2) executing control commands received from the Control VNF. The Driver VNF is the only DT building block that cannot be offloaded from the robot (Figure 1), in spite of the remaining virtual functions which enable robot control, visualization, and maintenance.

Control VNF: it allows low-level robot manipulation by running a hard-real-time-compatible control loop towards the Driver VNF following a control frequency at a timescale ranging between 10ms and 100ms. The

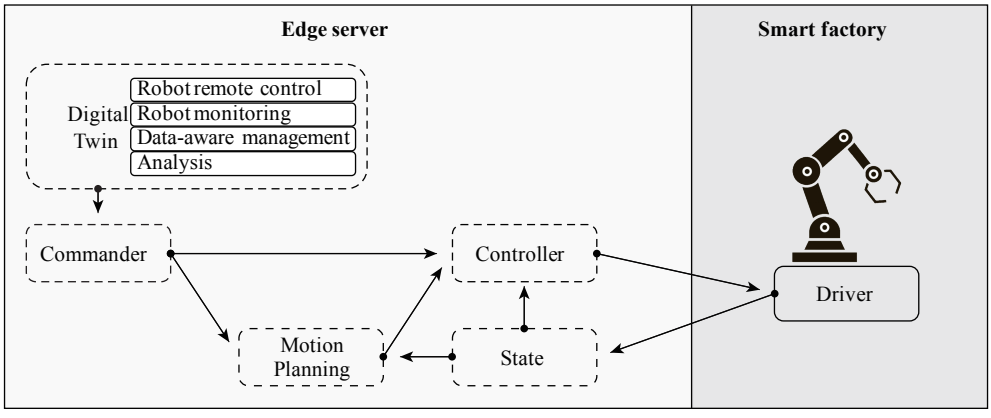


Figure 1 Edge-based digital twin solution configuration scheme.

communication between Control and Driver VNFs must necessarily take place in real time, and, hence, has strict latency constraints. The Control VNF offloading is therefore a sensitive and demanding operation, and its performance can be significantly affected by the communication latency.

State VNF: it is in charge of gathering information from the Driver VNF, hence of computing the direct kinematic. Namely, the State VNF keeps track of multiple coordinate frames provided by the Driver VNF over time, e.g., joint angles, and provides a high-level representation of the robot to the higher-level VNFs.

Motion Planning (MP) VNF: the MP VNF is the core component for planning the robotic arm's transitions. It receives as input both the current and the desired robot end-effectors configuration from, respectively, the State VNF and the Commander VNF, and runs the inverse kinematics and trajectory creation. This trajectory is composed of a series of navigation commands that are passed to the Control VNF to execute them towards the robot Drivers VNF. Both the inverse kinematic and the trajectory computations are associated with a high computational burden, making the MP VNF the best suited to offloading.

Commander VNF: it manages all the commands issued by the Digital Twin VNF, validating the corresponding parameters and handling concurrent requests. As highlighted in Figure 1, commands that require motion planning features, e.g., inverse kinematic computation or trajectory planning, are redirected to the Motion Planning VNF. Straightforward tasks are instead redirected directly to the Control VNF. The Commander VNF finally returns to the Digital Twin VNF potential output messages and status information.

Digital Twin (DT) VNF: the DT VNF provides high-level abstraction of the robot and of the entire service stack, hosting and keeping updated robot's virtual replica, and enabling user applications to interact with the robot through the DT, i.e., get human-understandable view of the robot, gather analytics data, and enable remote control. In the latter case, commands generated by the DT VNF are sent to the Commander VNF for parameters validation and concurrent requests solving.

3.2 Robot commands overview and Service VNFs interactions

Clearly, command generation and transmission play a critical role in performance evaluation and resource profiling of the Digital Twin service for remote robot control. Undoubtedly, not all commands have the same complexity in terms of the associated VNF computational overhead and thus the time required to validate, process, schedule, and execute the robot motion. Commands differ mainly in: (1) the scope of the associated motion trajectory, and (2) the level of abstraction of the command, the impact of which on the DT service is evaluated below. As depicted in Figure 2, there are three different types of robotic arm commands, each of

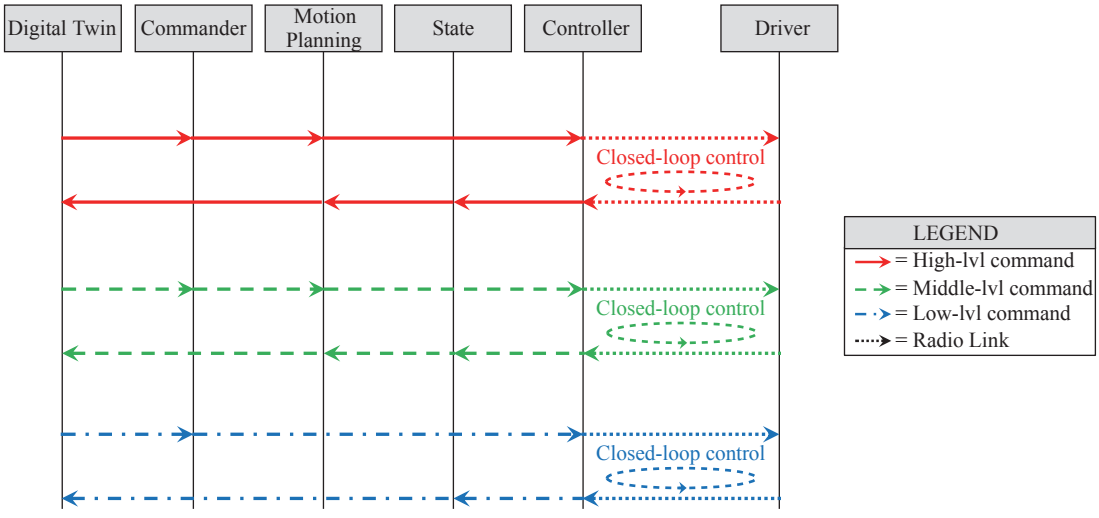


Figure 2 Flow graph of the interactions between the VNFs composing the Edge-based DT service, when high-level, middle-level and low-level commands are generated by the DT application.

which requires a different interaction between the VNFs that form the Edge-based DT service.

High-level commands: the intended robotic arm position is expressed in the robotic workspace. With regard to the robotic arm use case, the command is composed of 6 entries, half of which are the positions in the Cartesian space, while the remaining ones express the rotation and inclination of the gripper (roll, pitch, and yaw). The resolution of the inverse kinematics is therefore required when this command is issued by the Digital Twin VNF, as the workspace position is required to be mapped in the joint space. Only at the end of the inverse kinematics computation the MP VNF has gathered the information required for the trajectory computation, which will be conveyed to the Controller VNF. Hence, since the time requirements involved in motion planning are negligible, the high-level commands are particularly suitable for applications that do not require high precision in position control, e.g., polishing, grinding, and welding[27]. When the navigation trajectory execution ends, the Motion Planning VNF gets notified, which in turn informs the Digital Twin VNF. The resulting workflow is reported with a red solid line in Figure 2.

Middle-level command: in this first case study, the desired position of the robot is expressed as end-effectors configuration, i.e., through the intended angles of the axes making up the mechanical arm. Such a command needs to be at first validated by the Commander VNF, then conveyed to the Motion Planning, which in turn computes the navigation trajectory for the robotic arm. The inverse kinematics computation, i.e., the joint configuration corresponding to a given end-effector pose and orientation in the workspace, is not required in this scenario. As in the high-level command case, the Digital Twin VNF is informed about the command execution completion by the Motion Planning VNF. Thus, the involved workflow is the same as the high-level command's one and it is reported with a green dashed line in Figure 2.

Low-level command: in this last low-level case study, the desired robotic arm position is again expressed as intended end-effectors configuration, but in contrast with the middle-level command scenario, the command it is directly forwarded to the controller, as highlighted in Figure 2 with blue line. Being the MP VNF not involved, its functionalities are lost: since a suitably safe and smooth trajectory for the robot transition is not computed, there is no guarantee that the saturation limits and the resonant modes of the controlled robotic arm are not, respectively, violated and excited by the joint generalized forces exerted by the actuators during the transition, incurring the risk of a mechanical failure. Consequently, this command finds application when the robots are issued with repetitive and well-know safe transitions, or additional security mechanisms are present. On the other side, being the MP VNF the most computationally expensive component of the DT service, the overall Edge requirements are mitigated, making this low-level case study especially suitable for scenarios requiring low response and execution times, along with high precision, e.g., waterjet cutting, drilling, and milling processes. The first use-case in fact requires low response times since the pressure of the waterjet can cause deviations in the positions of the robots, while drilling and milling processes deal with large machining forces and durable materials to cut, thus requiring high path accuracy[27].

4 Testbed design and implementation

In order to experimentally evaluate the Edge-based Digital Twin service described above, we have designed and configured a lab testbed setup. The testbed is depicted in Figure 3 and consists of two main blocks: the edge host, which we will refer to as Edge Machine, in charge of offering computational resources to the DT service, as well as providing LTE connectivity to the second block of the architecture termed Robots Machine, i.e., the physical robots that conversely request and consume the service. The Edge Machine is equipped with 16GB RAM and an Intel(R) Core(TM) i7-7700HQ 4CPU@2.80 GHz processor. Conversely, the Robots Machine was supplied with 16GB RAM and an Intel(R) Core(TM) i7-8550U 4CPU@1.80 GHz processor.

Each of the VNFs composing the Edge-based Digital Twin service is deployed over a dedicated Virtual

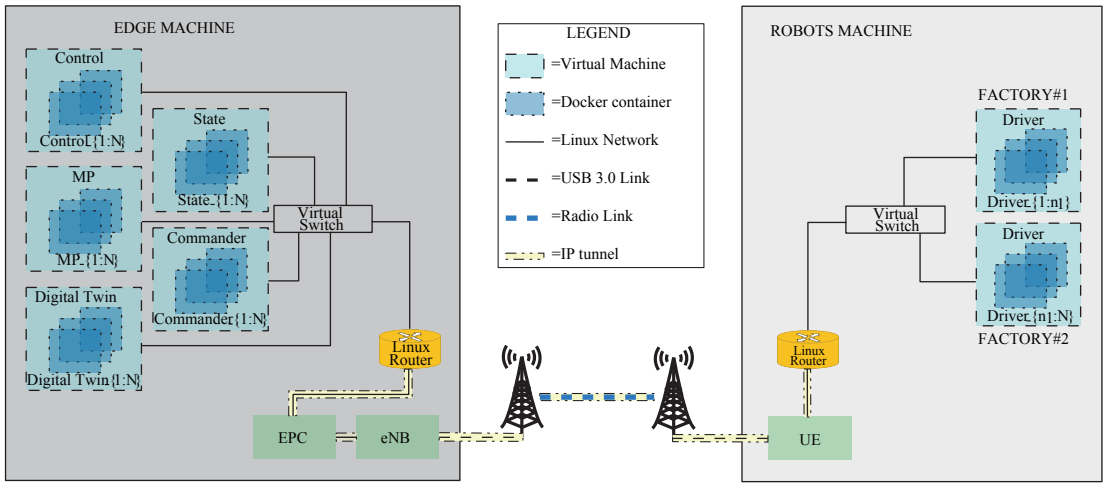


Figure 3 Testbed architecture. The Edge machine hosts Control, State, Motion Planning (MP), Commander and Digital Twin (DT) VNFs, while offering Digital Twin service and providing LTE connectivity to the Robot Machine. The latter hosts Driver VNF consuming the Edge-based Digital Twin service.

Machine (VM), running Ubuntu Server 20.04 LTS and provided with 2GB of RAM and 1vCPU. The Digital Twin, Commander, MP, State and Controller VNFs are deployed at the Edge Machine. The VNF driver is instead hosted on the Robots Machine.

Connectivity between the Edge and Robots Machines is based on srsRAN[23], an open-source SDR LTE full-stack implementation. Precisely, each physical machine is provided with an ETTUS Universal Software Radio Peripheral (USRP) B210 board using USRP Hardware Driver (UHD) v3.15. srsRAN enables 5G NSA solution, which however is not suitable for the current testbed, and the LTE Release 9 compliant implementation of virtualized EPC, eNB and UE functions. It can handle up to 20MHz FDD channels.

The remotely controlled robots deployed in the testbed are Niryo One[1] simulated robotic arms. Niryo One is a 6-axis open source collaborative robot designed for R&D. Niryo One leverages on the open-source Robot Operating System (ROS)[24], the widely recognized as de-facto standard middleware for robotic systems. ROS envisages the definition of a peer-to-peer network over TCP among ROS nodes, i.e., computing processes. One ROS node across the stack serves as master, thus acting as a nameservice. The Niryo One ROS stack is compliant with the service splitting presented in 3.1. For each robot registering to the service, an instance of each VNF composing the Niryo One ROS stack is deployed on Docker containers hosted by the corresponding VM. In our design, the DT VNF acts as the ROS master node.

Niryo provides the opportunity to run its collaborative robot arms in simulation mode. In the latter case, the hardware layer, as well as its functionalities and dependencies, is disabled, and the commands and the trajectories given to the robot are perfectly executed. Even though the robotic arms are simulated, their obligations, namely, the saturation limits for the generalized forces exerted by the actuators (e.g., torque) and the corresponding excitation of the structure's resonant modes, are acknowledged. Additionally, a 3D human understanding view of the simulated robotic arm is provided by means of the Rviz module and the Niryo One Studio graphical HMI. Although the interaction with this avatar allows for direct control of the physical robots, we interfaced with the latter via the programming interface on the ground of automation and reproducibility. In our experiments, we set the robotic arm speed to its maximum value of 0.4m/s for horizontal and vertical movements and of 90°/s for rotation. The closed-loop frequency has been set to 50Hz.

The high and the middle abstraction level commands can be configured in ROS with, respectively, "move pose" and "move joints" commands by leveraging the ROS Python API. Moreover, the ROS API enables low-

[1] https://niryo.com/product/niryo-one

level commands by means of JointTrajectory messages addressed to the Control VNF through the action server exposed by the Commander VNF. Hence, low-level commands will be referred to as "ROS API" commands. All the traffic between Edge VNFs and Driver VNFs is routed through the LTE radio link. However, since Edge VNFs and Driver VNFs belong to unrelated private networks, it has been necessary to leverage on a tunneling protocol.

Wired testbed: Before performing experimental evaluations on the testbed described above, we considered a simplified scenario without LTE connectivity to analyze and evaluate a broader range of scenarios. Undoubtedly, replacing the communication medium between Edge and Robots Machines with an Ethernet cable has mitigated two crucial limitations of the wireless connectivity implemented in the testbed. First and foremost, no computational resources are required to run srsRAN and keep the LTE radio link active, which enables the simulation of a higher number of Niryo One robotic arms. Furthermore, the wired channel provides guarantees of latency and stability, enabling longer-lasting experiments over time, and, hence, the achievement of as consistent as possible resource profiling. In the wired testbed, a 16GB RAM and Intel(R) Core(TM) i7-8550U 4CPU@1.80 GHz Edge and a 8GB RAM and Intel(R) Core(TM) i7-8250U 4CPU@1.60 Robots Machines have been connected to the same network through a router, by means of Gigabit Ethernet cables. At last, we mention that the VMs have been bridged to the same network, and that each of the VMs has been assigned 2GB of RAM and 2vCPUs. In this configuration, the system was able to support a maximum number of robots equal to 8 and 10 in the cases of high and medium command abstraction-levels, respectively.

5 Edge-based DT service profiling and experimental analysis

In this section, we first outline and justify the experimental setup we chose and whereby we obtained our profiling and experimental results (Section 5.1). We then present a CPU (Section 5.2) and memory usage (Section 5.3) profiling to investigate the requirements of the VNFs building up the DT service; further, we examine the execution times entailed by the commands under study (Section 5.4). This initial collection of experiments was conducted based on the wired testbed. We also present the configuration of the testbed including LTE connectivity and we explore the corresponding resource requirements by simulating a maximum number of 3 Niryo One robotic arms (Section 5.5). We evaluate the system performance in terms of latency at both the network and the application level (Section 5.6). Finally, we elaborate on the various potential types of industrial fields and applications that can benefit the most from this work (Section 5.7).

5.1 Experimental settings

In response to the need to gather knowledge about DT services required by enhanced and agile network resource orchestrator frameworks, we aim to experimentally evaluate and characterize an Edge-based DT solution for the remote control of manufacturing robotic arms. Our analysis therefore aims not only to obtain a resource profile of the service under study in relation to the service demand load, but also to inspect new trade-offs such as the one presented in Section 3.2. The results of such an analysis will contribute to the development of novel tools for resource scaling, and thus to the fulfilment of the benefits introduced by recent advances in network and service virtualization, namely, high flexibility, enhanced scalability and agile management, in addition to resource savings and performance guarantees.

We study the requirements and performance of the Edge-based DT service VNFs and their correlation with two parameters playing a crucial role in the intended analysis. The first parameter whose impact is analyzed in this work is the number of robots concurrently requesting the Digital Twin service. Specifically, we want to: (1) analyze its correlation with service VNFs demand of resource; (2) investigate the relationship with the involved latency at different network architecture logical levels. The same analysis is carried out in relation to

the three-command case study outlined in Section 3.2 and imposed on the robots.

To fairly perform the latter study, we ensured that the robots followed the same trajectory regardless of the command entity imposed, even if it was computed by the motion planning VNF, i.e., in the "move joints" and "move pose" use cases, or precomputed, i.e., in the case of commands over the ROS API.

The commands are sent by the DT VNFs, each of which executes a Python script conveying instructions to the corresponding robotic arm. More precisely, a new instruction is formalized by the DT VNFs once the previous one is completely carried out by the robotic arm, keeping the robots in continuous motion, without introducing intervals of inactivity between instructions. Concurrently, the virtual machines hosting the service VNFs execute Python-based benchmarking code, thus collecting information about the CPU and RAM usage of the virtual systems. The library exploited for system monitoring is *psutil*[2], a multi-platform library for the collection of information about systems or running processes resource consumption.

5.2 CPU profiling

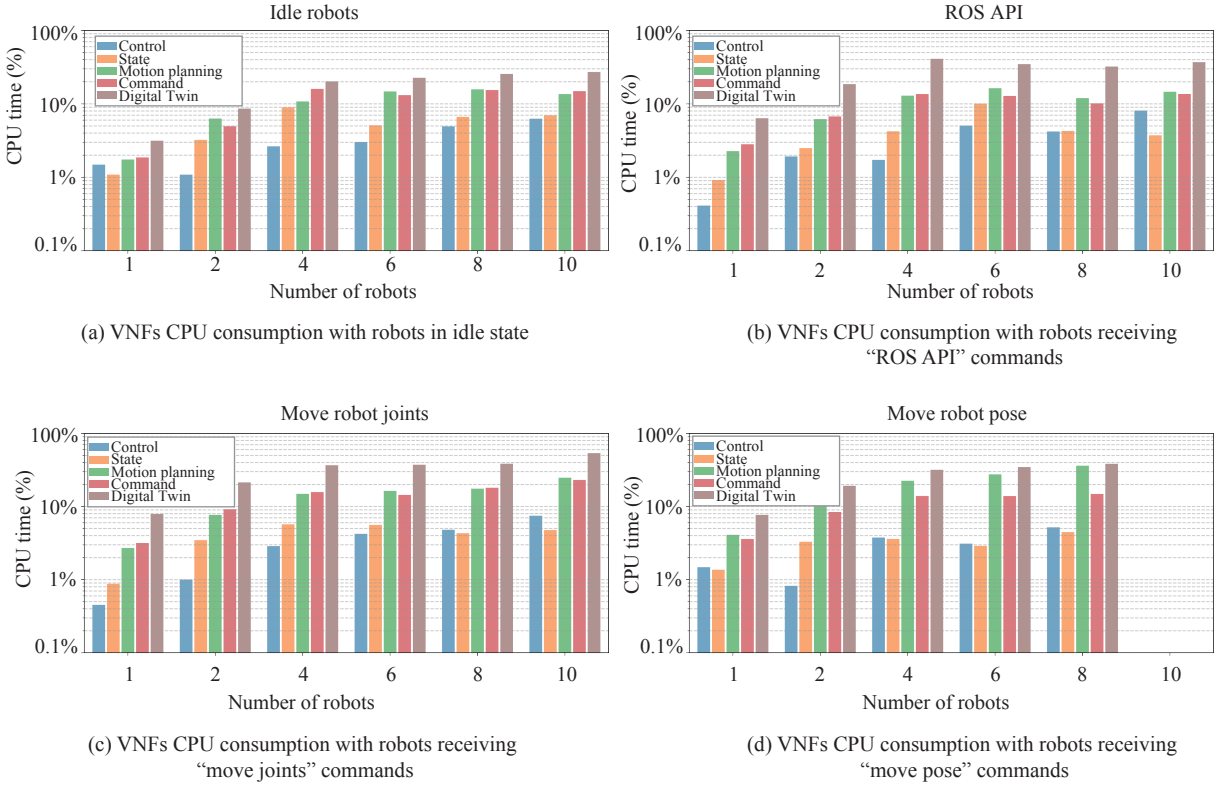
Here we examine the VMs' computational requirements when hosting the offloaded DT service VNFs. In our analysis, we thereby account for the overhead involved by the instantiation of the Docker containers to implement the VNFs, so as to characterize the DT service in its thoroughness. We emphasize that for this first analysis we relied on the wired testbed as described in Section 4.

Figure 4 shows how the CPU time required by the VMs varies when a variable number of Niryo One robots are controlled by the Edge-based DT service; in particular, the following cases are considered: (a) idle (i.e., the robotic arms are registered to the service without receiving any instruction), (b) receive "ROS API" instructions, (c) receive "move joints" instructions, and (d) receive "move pose" instructions. This "idle" analysis aims to provide a resource demand reference, as it highlights the effort required to support a robot registered to the DT application, therefore to keep up the containerized VNFs and the peer-to-peer ROS network, besides supporting the control closed-loop which is always up and running. Also, notice that in Figure 4 the CPU time has been normalized with respect to the experimental measurement time (i.e., 15 minutes) and to the number of vCPUs assigned to each virtual machine. So doing, a 100% CPU time measurement indicates the saturation of all the assigned vCPUs throughout the entire duration of the system tracing.

By comparing Figure 4a and Figure 4b, we observe that the only VNF to require a greater allocation of resources when the robots are instructed via "ROS API" commands instead of standing idle is the Digital Twin VNF. The consumption of the latter is in fact slightly, but systematically, higher with moving robots.

This is due to the execution of Python scripts used to instruct the robots. Conversely, by looking at Figure 4c and, in particular, in Figure 4d the virtual machine being notably proved is the one hosting Motion Planning VNFs. As outlined in Section 3, in fact, the Motion Planning VNFs are required to plan the movement trajectory in the "move joints" use case, to which the Inverse Kinematics is added in the case of "move pose" commands. In fact, the latter scenario involves the highest computational load. Specifically, the CPU time associated with the VM hosting MP VNFs serving 8 robotic arms is, on average, 17.05% when the robots "move joints" commands are imposed; 35.13% with "move pose", instead. In the latter scenario, it is not possible to continuously support 10 robotic arms. In order to examine this constraint, we report in Figure 5 the CPU consumption temporal evolution of the MP VM when 8 Niryo One robots are remotely controlled through their Digital Twin replica by means of "move joints" (green curve) or "move pose" (blue curve) commands. The CPU consumption has been sampled with a 500ms periodicity. We observe that the Motion Planning VM's CPU consumption is characterized by a standard deviation σ, which is substantial when

[2] https://pypi.org/project/psutil/



(a) VNFs CPU consumption with robots in idle state

(b) VNFs CPU consumption with robots receiving "ROS API" commands

(c) VNFs CPU consumption with robots receiving "move joints" commands

(d) VNFs CPU consumption with robots receiving "move pose" commands

Figure 4 VNFs CPU consumption (y axis) by varying the number of served simulated robots (x axis) and the command abstraction-level. Control VNF consumption is reported in blue, State VNF in orange, Motion Planning VNF in green, Command VNF in red, Digital Twin in brown.

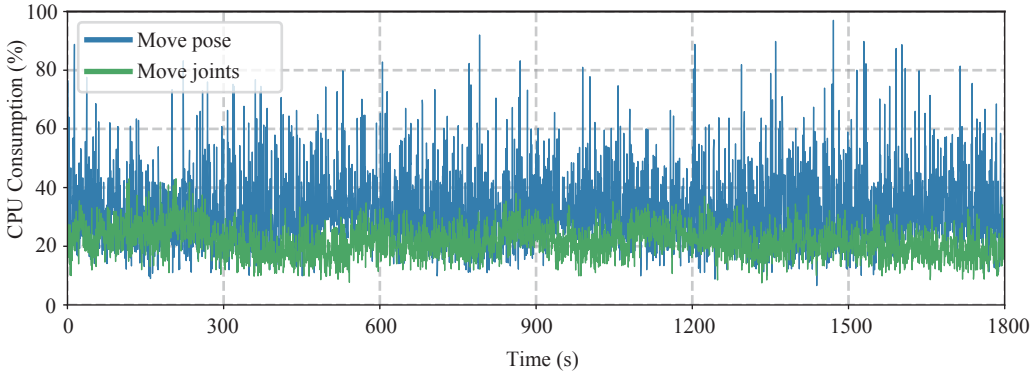


Figure 5 Evolution over time of the CPU consumption associated to the VM hosting Motion planning VNFs when 8 robots registered to DT service are receiving "move pose" (blue line) or "move joints" (green line) commands.

processing commands expressed in the workspace, i.e., for "move pose", $\sigma = 25.8\%$. Conversely, the standard deviation associated with CPU consumption is significantly lower when commands are expressed in the joint space, i.e., for "move joints", $\sigma = 10.5\%$. Finally, we note that the CPU consumption peaks approach the computational capacity in the first case, indicating that indeed it is not possible to simulate 10 robots unless a larger amount of computing resources were available.

Ultimately, we point out that the choice of commands, used to remotely control the robot, has not a large impact on the remaining VNFs. Indeed, (1) since the direct kinematics resolution is not required by the application under study, the State VNFs never get overloaded; (2) the Commander VNF feature, i.e., validating the commands given by the Digital Twin VNF, is not a resource-intensive task.

5.3 RAM profiling

Table 1 reports the average RAM usage associated with the DT service VNFs measured via the psutil library and averaged over a 15-minute time interval. Unlike what was observed above about the computational demand, RAM usage is not correlated with the motion imposed to the robots, i.e., the RAM occupancy remains unchanged whether the Niryo One robots are in idle state or whether they are controlled remotely. Furthermore, comparing the memory usage when 1 or 8 robots are connected to the Edge-based DT service, we note that the RAM does not exhibit any scalability problem.

Table 1 VNFs RAM usage when 1 or 8 robots are registered to the service and for different command abstraction-levels

	RAM usage [MB]							
	1 Niryo One robot				8 Niryo One robots			
	Idle	Joints	Pose	ROS API	Idle	Joints	Pose	ROS API
Control VNF	475 MB	474 MB	477 MB	475 MB	1060 MB	1110 MB	1029 MB	1084 MB
State VNF	341 MB	342 MB	343 MB	338 MB	727 MB	727 MB	728 MB	727 MB
MP VNF	344 MB	345 MB	345 MB	342 MB	781 MB	788 MB	789 MB	784 MB
Commander VNF	591 MB	587 MB	594 MB	592 MB	997 MB	979 MB	996 MB	940 MB
Digital Twin VNF	742 MB	810 MB	801 MB	794 MB	925 MB	827 MB	867 MB	829 MB

5.4 Command execution time

Next, we investigate the command execution times, namely the time elapsed since the generation of the command and the notification of its correct execution. We compare again the three commands under study, in order to experimentally demonstrate strengths and weaknesses of each of them. To complete this evaluation, we rely again on the wired testbed. Figure 6 shows the average execution times as the number of robots registered for the service varies.

Above all, we observe that the command execution time is not affected by the number of robotic arms consuming the service, suggesting that as long as the correct amount of resources is allocated and guaranteed for the service, the robots enjoy the same quality of experience. We also note that execution times differ considerably by changing the way to interact with the robots: (1) firstly, the time requirement difference between “move joints” and “move pose” commands is given by the inverse kinematics computation; (2) the “ROS API” commands, since they do not invoke the Motion Planning VNF to which the most resource-intensive features are associated, involve a significantly limited execution time, proving that “ROS API” commands are more suitable for tasks where low execution times and high precision are required.

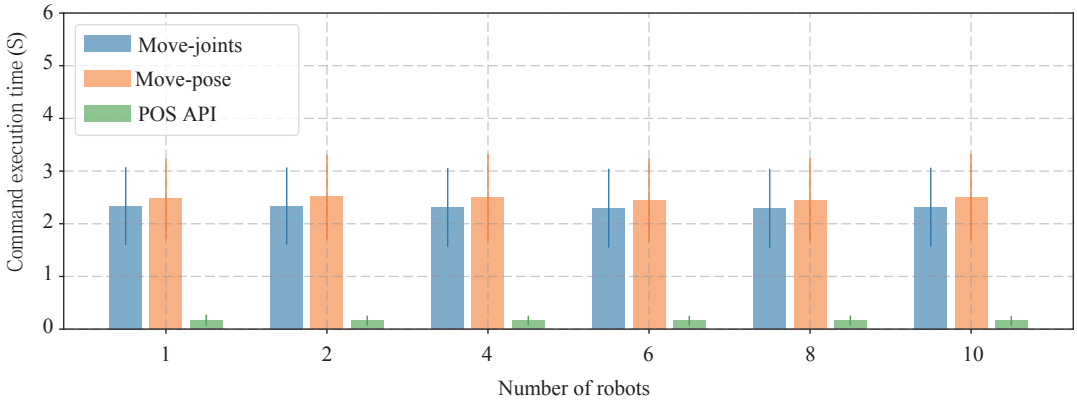


Figure 6 Command execution time (y axis) by varying the number of served simulated robots (x axis) and the command abstraction-level. “move joints” command is reported in blue; “move pose” command is reported in orange; “ROS API” command is reported in green.

5.5 LTE profiling

In the implemented testbed (Figure 3), we have configured the Edge to offer a 10-MHz LTE connectivity to the Niryo One robots in order to provide the Digital Twin service. Since the performance assessment of the service in relation to LTE channel quality is beyond the scope of this work, we considered in our experiments an ideal communication channel characterized by a signal-to-noise-ratio (SNR) that is always higher than 27dB, both in uplink and in downlink. Under such a testbed configuration, we were able to simulate the remote control of up to 3 robotic arms.

In order to fully-characterize the edge requirements, we profile the SDR LTE full-stack implementation offered by srsRAN. Table 2 presents the CPU time of the virtual eNB by varying again the number of robots with an instantiated Digital Twin replica and remotely interaction method. The CPU time has been normalized according to the duration of the experiment, which lasted 3 minutes. Thus, a 100% CPU time measurement indicates the saturation of a single CPU supplying the Edge machine during the entire duration of the system tracing. The measurements reported in the table show that srsRAN computational requirements marginally grow with the number of connected robots. The reason behind this behavior lies in the traffic loads involved in the analyzed scenarios. In fact, for 1, 2 and 3 robots consuming the service, traffic loads of respectively about 1.1, 1.8 and 2.4 Mbps were measured.

Table 2 Virtual srsRAN eNB CPU time variation when 1, 2 or 3 robots are registered to the service, by varying the command abstraction-level

	Virtual eNB CPU time [%]		
Number of active robots	"move joints"	"move pose"	"ROS API"
1	29.56%	29.13%	28.95%
2	31.65%	30.43%	30.00%
3	33.45%	32.33%	31.35%

5.6 Network and application latencies

Figure 7 reports the round-trip time (RTT) values at network (a) and application (b) levels, and measured when leveraging on the LTE-based testbed. As expected, in light of the traffic load reported in the preceding section, along with the ideal channel hypothesis which guarantees near-to-zero packet losses over TCP, we note that the latency values are not affected by the number of robots requesting the service. However, we point out a crucial insight about the LTE limitations when supporting a Digital Twin service. The network and application RTT values that were measured in our experiments are comparable to the Niryo One control period, which was set to its minimum configurable value i.e., 20ms. From this observation we conclude that when leveraging 4G, the robotic arm remote control through its Digital Twin replica is viable only depending upon the chosen control frequency. When the robot use case scenario requires low control periods, i.e., under 30ms, hence high precision in the trajectory execution is mandatory, LTE cannot be serviceable. This paves the way towards the most recent radio access technologies, e.g., 5G or Wi-Fi 6E, which establish themselves as the most promising wireless technologies to meet the latency requirements involved by Digital Twin applications.

5.7 Application scenarios of the experimental results

Despite relying on a specific robotic arm model (i.e., 6-axis Niryo One collaborative robotic arms), the obtained results are easily generalizable and extensible to all industrial applications that can benefit from advanced path planning and motion control capabilities of industrial robotic arms. Such applications are innumerable and continuously growing in every manufacturing sector, including automotive, food, wood, plastics, and electronics[28]. The most common industrial applications encompass assembly processes (i.e., part gripping, placement and welding), subtractive manufacturing systems (e.g., polishing, grinding, machining,

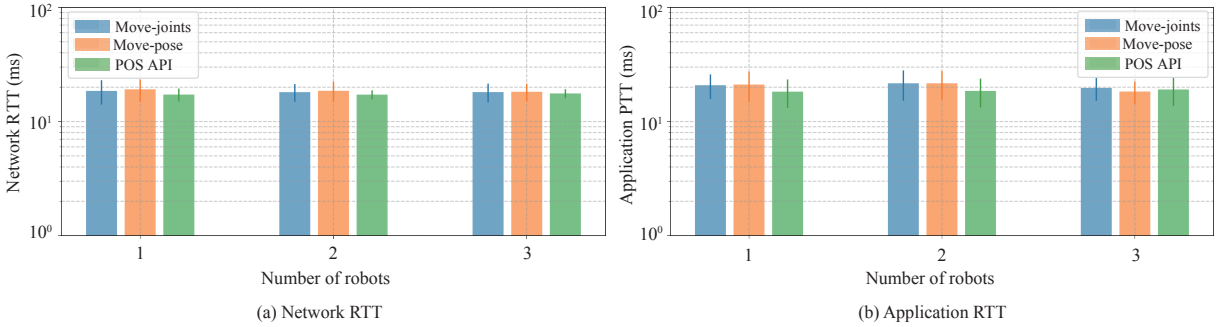


Figure 7 Network (a) and Application (b) level RTT (y axis) by varying the number of served simulated robots (x axis) and the command abstraction-level. "move joints" command is reported in blue; "move pose" command is reported in orange; "ROS API" command is reported in green.

milling), manufactured products inspection and packaging, as well as material handling (e.g., sorting and grouping, pick and place).

Hence, the global manufacturing sector can benefit primarily from the Edge-based Digital Twin solution for remote control of robotic arms presented in Section 3, which further encourages the deployment of advanced network-based manufacturing robotic solutions. The DTaaS design can in fact be easily readjusted regardless of the robotic system. Moreover, smart industrial processes can be refined by leveraging the accomplished experimental results and the drawn insights. Specifically, the presented novel trade-off between the robotic arm's workload and position control accuracy results in an informed choice of command abstraction-levels. Finally, the manufacturing processes will be able to benefit from a DTaaS made highly flexible and scalable thanks to agile and automated resource scaling and orchestration frameworks defined on the basis of the resource profile here presented.

6 Conclusion and future work

We presented the experimental evaluation and profile of an Edge-based Digital Twin solution designed for the remote control of robotic arms. The service was split into virtual network functions, deployed on a laboratory setup and offered to 6-axis Niryo One simulated robotic arms.

In an initial measurement campaign, the robotic arms leveraged Gigabit Ethernet cables to join the service, to investigate the interconnection between the resources occupied only by the service and the number of remotely controlled robots. Our results show that the most critical function of the Digital Twin as a Service is the inverse kinematics computation, followed by the movement trajectory plan. Both these functions are taken over by the Motion Planning VNF. We then analyzed the impact of the commands imposed on the robots on the service profile. Particularly, measurements proved that the exploitation of low abstraction level commands can lead to relevant computational resources savings, thus great performance benefits; however, additional safety mechanisms are required if the trajectories are not predictable. Finally, in a second measurement campaign, LTE shortage in accommodating real-time DT applications has been empirically proved.

For future research, the testbed is going to integrate enhanced radio communication technologies, e.g., 5G and Wi-Fi 6E, in order to meet the stringent real-time latency requirements involved by the operational DT. Moreover, in accordance with the results derived in this work, the need for an agile and automated network orchestration framework emerges to enhance resource usage efficiency and provide performance guarantees. We then intend to design and deploy an automated smart network service orchestrator able to ensure resource utilization and energy consumption optimization while avoiding running into service disruptions caused by shortage of allocated resources. In this regard, in compliance with the underlying and driving idea of 5GB and

6G, Artificial intelligence (AI) and Machine Learning (ML) may provide the key tools to achieve the aforementioned goals. ML, in particular, can support and leverage network slicing capabilities at best, in order to logically isolate resource pools dedicated to different industrial vertical applications.

Declaration of competing interest

We declare that we have no conflict of interest.

References

1 Brem A, Wolfram P. Research and development from the bottom up–introduction of terminologies for new product development in emerging markets. Journal of Innovation and Entrepreneurship, 2014, 3(1): 9
DOI: 10.1186/2192-5372-3-9

2 Gorecky D, Schmitt M, Loskyll M, Zühlke D. Human-machine-interaction in the industry 4.0 era. In: 2014 12th IEEE International Conference on Industrial Informatics (INDIN). 2014, 289–294

3 Tao F, Qi Q. New IT driven service-oriented smart manufacturing: framework and characteristics. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2019, 49(1): 81–91
DOI: 10.1109/tsmc.2017.2723764

4 Ghosh A K, Ullah A S, Kubo A. Hidden Markov model-based digital twin construction for futuristic manufacturing systems. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2019, 33(3): 317–331
DOI: 10.1017/S089006041900012X

5 Barricelli B R, Casiraghi E, Fogli D. A survey on digital twin: definitions, characteristics, applications, and design implications. IEEE Access, 2019, 7: 167653–167671
DOI: 10.1109/access.2019.2953499

6 Groshev M, Guimaraes C, De La Oliva A, Gazda R. Dissecting the impact of information and communication technologies on digital twins as a service. IEEE Access, 2021, 9: 102862–102876
DOI: 10.1109/access.2021.3098109

7 Leng J, Zhang H, Yan D, Liu Q, Chen X, Zhang D. Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. Journal of Ambient Intelligence and Humanized Computing, 2019, 10(3): 1155–1166
DOI: 10.1007/s12652-018-0881-5

8 Rosen R, von Wichert G, Lo G, Bettenhausen K D. About the importance of autonomy and digital twins for the future of manufacturing. IFAC-PapersOnLine, 2015, 48(3): 567–572
DOI: 10.1016/j.ifacol.2015.06.141

9 Bao J, Guo D, Li J, Zhang J. The modelling and operations for the digital twin in the context of manufacturing. Enterprise Information Systems, 2019, 13(4): 534–556
DOI: 10.1080/17517575.2018.1526324

10 Um J, Weyer S, Quint F. Plug-and-Simulate within Modular Assembly Line enabled by Digital Twins and the use of Automation ML. IFAC-PapersOnLine, 2017, 50(1): 15904–15909
DOI: 10.1016/j.ifacol.2017.08.2360

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

12 Rajkumar R, Lee I, Sha L, Stankovic J. Cyber-physical systems: the next computing revolution. In: Design Automation Conference. Anaheim, CA, USA, IEEE, 2010, 731–736

13 Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: vision and challenges. IEEE Internet of Things Journal, 2016, 3(5): 637–646
DOI: 10.1109/jiot.2016.2579198

14 Zeb S, Mahmood A, Hassan S A, Piran M J, Gidlund M, Guizani M. Industrial digital twins at the nexus of NextG wireless networks and computational intelligence: a survey. Journal of Network and Computer Applications, 2022, 200103309
DOI: 10.1016/j.jnca.2021.103309

15 Adamuz-Hinojosa O, Ordonez-Lucena J, Ameigeiras P, Ramos-Munoz J J, Lopez D, Folgueira J. Automated network service scaling in NFV: concepts, mechanisms and scaling workflow. IEEE Communications Magazine, 2018, 56(7): 162–169
DOI: 10.1109/mcom.2018.1701336

16 Willig A, Matheus K, Wolisz A. Wireless technology in industrial networks. Proceedings of the IEEE, 2005, 93(6): 1130–1151
DOI: 10.1109/jproc.2005.849717

17 Thomesse J P. Fieldbus technology in industrial automation. Proceedings of the IEEE, 2005, 93(6): 1073–1101
DOI: 10.1109/jproc.2005.849724

18 Xia K, Sacco C, Kirkpatrick M, Saidy C, Nguyen L, Kircaliali A, Harik R. A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence. Journal of Manufacturing Systems, 2021, 58: 210–230
DOI: 10.1016/j.jmsy.2020.06.012

19 Yi Y, Yan Y, Liu X, Ni Z, Feng J, Liu J. Digital twin-based smart assembly process design and application framework for complex products and its case study. Journal of Manufacturing Systems, 2021, 58: 94–107
DOI: 10.1016/j.jmsy.2020.04.013

20 Malik A A, Brem A. Digital twins for collaborative robots: a case study in human-robot interaction. Robotics and Computer-Integrated Manufacturing, 2021, 68: 102092
DOI: 10.1016/j.rcim.2020.102092

21 Hammad A, Ali S S, Eldien A S T. A novel implementation for FastSLAM 2.0 algorithm based on cloud robotics. In: 2017 13th International Computer Engineering Conference (ICENCO). Cairo, Egypt, IEEE, 2017, 184–189

22 Wang X V, Wang L, Mohammed A, Givehchi M. Ubiquitous manufacturing system based on Cloud: a robotics application. Robotics and Computer-Integrated Manufacturing, 2017, 45: 116–125
DOI: 10.1016/j.rcim.2016.01.007

23 Rahimi R, Shao C, Veeraraghavan M, Fumagalli A, Nicho J, Meyer J, Edwards S, Flannigan C, Evans P. An industrial robotics application with cloud computing and high-speed networking. In: 2017 First IEEE International Conference on Robotic Computing. Taichung, Taiwan, China, IEEE, 2017, 44–51

24 Huang H, Yang L, Wang Y, Xu X, Lu Y. Digital Twin-driven online anomaly detection for an automation system based on edge intelligence. Journal of Manufacturing Systems, 2021, 59: 138–150
DOI: 10.1016/j.jmsy.2021.02.010

25 Liu C, Le Roux L, Körner C, Tabaste O, Lacan F, Bigot S. Digital twin-enabled collaborative data management for metal additive manufacturing systems. Journal of Manufacturing Systems, 2022, 62: 857–874
DOI: 10.1016/j.jmsy.2020.05.010

26 Girletti L, Groshev M, Guimarães C, Bernardos C J, de la Oliva A. An intelligent edge-based digital twin for robotics. 2020 IEEE Globecom Workshops (GC Wkshps), 2020, 1–6

27 Kim S H, Nam E, Ha T I, Hwang S H, Lee J H, Park S H, Min B K. Robotic machining: a review of recent progress. International Journal of Precision Engineering and Manufacturing, 2019, 20(9): 1629–1642
DOI: 10.1007/s12541-019-00187-w

28 Gautam R, Gedam A, Zade A, Mahawadiwar A. Review on development of industrial robotic arm. International Research Journal of Engineering and Technology, 4(3), 2017, 1752–1755