applied sciences

MDPI

Article

Simulation Power vs. Immersive Capabilities: Enhanced Understanding and Interaction with Digital Twin of a Mechatronic System

Constantin-Catalin Dosoftei

Department of Automatic Control and Applied Informatics, "Gheorghe Asachi" Technical University of Iasi, 700050 Iasi, Romania; constantin-catalin.dosoftei@academic.tuiasi.ro

Abstract: Automation Studio, a specialised simulation software, offers virtual commissioning capabilities and robust tools for modelling the behaviour and performance of a pneumatic robot controlled by a PLC. Conversely, Unity is a versatile platform primarily used for creating high-quality 3D games and interactive simulations, providing immersive experiences with DTs through mixed-reality environments. This paper provides a study that compares and contrasts the simulation power of Automation Studio and the immersive capabilities of Unity in the context of developing digital twins for a mechatronic system. This research explores how these complementary approaches enhance the development, validation, understanding, and interaction with digital twins. By examining both platforms in this context, the article provides valuable insights for engineers, developers, and researchers looking to create digital twins for mechatronic systems, but not only. This study demonstrates the potential of leveraging the combined power of simulation and immersive capabilities to improve the interaction between the real robotic arm manipulator cylindrical type and its digital twin in different scenarios, using an OPC approach for mirroring. The combination of Automation Studio and Unity provides a powerful platform for applied science education in the field of the digital twin of mechatronic systems.

Keywords: digital twin; mechatronic system; virtual commissioning; mixed reality; cyber-physical system

Citation: Dosoftei, C.-C. Simulation Power vs. Immersive Capabilities: Enhanced Understanding and Interaction with Digital Twin of a Mechatronic System. *Appl. Sci.* **2023**, *13*, 6463. https://doi.org/10.3390/app13116463

Academic Editors: Francesco Longo, Antonio Padovano, Vittorio Solina and Giovanni Mirabelli

Received: 30 April 2023
Revised: 20 May 2023
Accepted: 22 May 2023
Published: 25 May 2023

1. Introduction

Industry 4.0 is a new reality of the modern economy because innovation and technological development play important roles in all organisations. As the first industrial revolution improved the operation of manufacturing industries, the second one introduced electricity into the industry, the third automated the uniform tasks of line workers, and the fourth aimed at improving information management and decision making. Cyber-physical systems (CPS), the Internet of Things (IoT), the Internet of Service (IoS), and smart factories are the four main pillars of Industry 4.0 [1]. These components work together to create highly automated and efficient production environments. Seven design principles further define Industry 4.0: interoperability, virtualisation, decentralisation, real-time capability, service orientation, modularity, cost reduction, and efficiency [2]. All components of the industrial network have a flawless connection thanks to interoperability. Decentralisation spreads decision-making and control, while virtualisation makes it possible to represent real assets digitally. Rapid adaptation to shifting circumstances is made possible by real-time capabilities, whereas service orientation emphasises the integration of modular, reusable services. Developing systems with easily interchangeable components is called modularity. Cost reduction and efficiency are focused on optimising industrial operations to increase productivity and profitability while minimising waste and resource consumption. Smart manufacturing systems can improve process efficiency, reduce downtime, and enhance supply chain management by leveraging evolutive technologies such as

IIoT, AI, big data analytics, additive manufacturing, distributed ledgers, cloud computing, and robotics. By integrating the latest technologies into their operations, organisations can reduce costs, increase productivity, and enhance sustainability, all while delivering high-quality products and services to customers. Businesses may develop highly automated, interconnected, and effective smart manufacturing systems by implementing Industry 4.0 ideas and technology. Organisations must undergo this change to be competitive in a world market that is always changing. Businesses may provide value to consumers and shareholders while fostering sustainable economic growth by utilising the most recent digital technology to enhance operations and achieve long-term success.

To define the term I4.0 only through one word, this is digitalisation. The concept of "digitalisation" alludes to our society's greater reliance on automation, networked, and web-enabled technology [3]. The main technology pushing digital transformation across all industries, and not only, is called digital twin (DT), representing a concept associated with cyber–physical integration. The DT concept came to life in 2002 at the University of Michigan when Prof. Grieves introduced, during a PLM course, the concept of virtual space that was equivalent to a physical product and with a bidirectional communication channel between these two spaces [4]. Presently, there are several different definitions of a digital twin [5–10]. Nonetheless, the essence of all of them stays with the initial concept of a mirrored spaces model with bridges of communication between the physical world and the virtual world, and the conceptual model is presented in Figure 1.

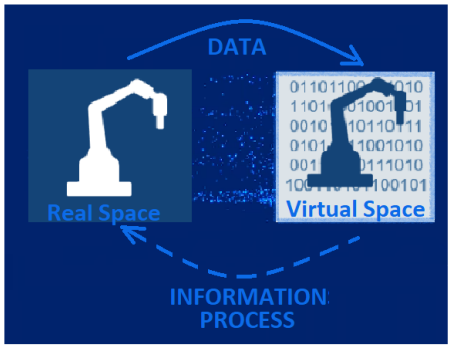


Figure 1. Conceptual model of the digital twin.

Despite the DT philosophy and model proposed in 2002, its application remained limited in the first decade for various reasons [11]. During this time, another concept associated with integrating CPS made significant progress, and CPS was hailed as a main pillar of Industry 4.0. Some perspectives view DT as a subset of CPS, [12]; however, notable differences exist. Introduced in 2006, CPS primarily functions as a scientific category, addressing complex systems that traditional IT terminology could not sufficiently describe. CPS represents a profound integration of computation, networking, and physical processes. Embedded computers and networks monitor and control the physical processes, with feedback loops where physical processes affect computations and vice versa. In contrast, a digital twin (DT) is a high-fidelity virtual model that mirrors a physical entity. A DT simulates the physical entity's behaviour in a virtual environment by utilising real-time data, allowing system performance monitoring, prediction, and optimisation.

To underscore this distinction, consider a traffic monitoring and control system. In the CPS context, this system involves surveillance cameras, traffic sensors, traffic lights, and embedded computers communicating with each other to adjust the traffic flow based on real-time road conditions. On the other hand, a DT for the same system would be a detailed virtual model of the traffic network, replicating the real-world situation in real-time based on data collected from sensors and cameras. While CPS and DT aim to integrate the physical and digital worlds, their functions and applications differ. CPS focuses on the real-time monitoring and control of physical systems using computational processes, while DT centres around virtual modelling and simulation. Even though CPS and DT share the goal

of bridging the gap between the physical and digital domains, their primary distinction lies in their categorisation. As a scientific category, CPS focuses on integrating computational and physical processes. Conversely, as an engineering category, DT emphasises using a digital copy of a physical system for real-time optimisation. This difference influences their respective roles and applications in industrial practices, with DTs being more prevalently used for enhancing precision and management.

Let us consider another example—a wind turbine system to illustrate these differences. In the CPS context, integrated sensors and controllers within the wind turbine system continuously measure variables such as wind speed, blade position, and power output. These data are then processed to adjust the turbine's blade position and other operational parameters to optimise the real-time power output. On the other hand, a DT of the same wind turbine would be a virtual model that uses real-time input data from the CPS to simulate the turbine's operation in a virtual environment. Engineers can use this model to anticipate and plan maintenance, assess the impact of different weather scenarios on turbine performance, and test new control strategies or equipment configurations in a safe and controlled environment.

These examples illustrate how CPS and DT complement each other to maximise the efficiency and reliability of modern industrial systems. While both technologies allow for greater integration between the physical and digital worlds, they approach this task from different perspectives and with various tools, proving their immense value in the era of Industry 4.0. manufacturing; their symbiosis promises a leap forward in the efficiency, flexibility, and intelligence of production systems.

1.1. DT Applications and Benefits

DTs are more commonly used in applied sciences due to their practical applications and direct implementation in various industries [8,13]:

- Manufacturing—used to improve production efficiency, reduce downtime, optimise the flow of materials and resources, and enhance product quality;
- Aerospace—utilised for monitoring, weight measuring, the exact specification of climatic factors, flying time measurement, and fault diagnosis in aircraft;
- Automotive—used to simulate the behaviour of various components and systems, allowing for the early detection of potential problems and reducing the need for physical prototyping. Nowadays, it can be used to develop autonomous vehicles and support the development of new technologies such as electric powertrains and advanced driver assistance systems;
- Agriculture—enables farmers to create virtual representations of their crops, allowing for the real-time monitoring and optimisation of growth and yield;
- Healthcare—utilised to model and simulate the human body. Doctors can test various therapies and forecast outcomes before actually providing them by creating a virtual clone of the patient;
- Construction—allows architects and engineers to create virtual representations of buildings, which can be used to identify potential problems and optimise performance before construction even begins;
- Energy—used to simulate and optimise power generation and distribution systems;
- Education—allows for the creation of holographic replicas of actual things and systems, resulting in a more engaging and participatory learning environment for actual native digital students.

DTs have experienced significant growth in recent years, both conceptually and practically. As DTs evolve, they are expected to create a "cyber-physical continuum" that blurs the distinction between the virtual and the real [14,15]. With the introduction of autonomous decision-making, the applications of DTs will only continue to grow, creating new opportunities for innovation and progress.

DT technology has many advantages to offer enterprises, including increased reliability, productivity, reduced risk, lower maintenance costs, faster production, and better

customer service. DT also creates new business opportunities and provides insights into product performance throughout its lifecycle. When looking at the DT benefits, it is possible to see that this technology positively impacts several aspects of the manufacturing industry and the supply chain as a whole. Some of the benefits of introducing digital twinning technologies within the company include improved and the research, design, and development of new products and services, greater process efficiency and productivity, reduced waste, increased supply chain visibility, and more efficient unit-level traceability, reduced risk of counterfeit and adulterations, and increased product safety [16]. Access to more data about a product's journey to the end user and customer information is also a significant benefit of DT technology.

1.2. DT Adoption in Romania

The range of DTs' applicability has expanded from the product design stage to the production, operation, and service stages, garnering considerable interest from academics and businesses alike.

DT technology is in the early stages of adoption in Romania, but its potential applications are being explored in various industries. Examples include implementing DT-based smart city components in a few cities; DT applications in automotive manufacturing started from robotic applications and utility companies, especially from the energy and healthcare sectors. The adoption of DT technology in Romania is being facilitated by the participation of representatives from major universities and the uptake of DT research themes from variousindustries [17–20]. In addition, university involvement in digital transformation hubs promotes the widespread adoption of DTs in the Romanian environment [21,22]. This engagement by the academic community and collaboration with digital transformation initiatives create opportunities for the wider adoption of DTs in Romania. On the other hand, the development of the IT industry in Romania has created the conditions for the opening of subsidiaries of multinational companies involved in software development and the implementation of the DT concept in different sectors, as well as local companies that integrate DT in special near-robot applications from the manufacturing industry.

1.3. Synergy between DT, Virtual Commissioning, and Mixed Reality

Digital twin is a key technology for improved industrial performance, as it integrates or merges with many other technologies from the Industry 4.0 paradigm. These include IIoT, big data and analytics, artificial intelligence, cloud computing, virtual commissioning, modelling and simulation, extended reality, and autonomous robots, as depicted in Figure 2. The particular technology employed in digital twin technology varies based on the nature of the application and its specifications.

The research presents a systematic approach combining digital twin technology with VC to optimise the design and operation of a cylindrical robot while incorporating mixed reality for seamless interaction with the real robot and its hologram. This synergy facilitates real-time adjustments, enhancing the system's overall performance and minimising the need for physical prototyping. Furthermore, this integrated solution streamlines the development process, enabling students and engineers to identify potential issues, validate control algorithms, and test various scenarios in a secure virtual space before the actual implementation. Additionally, this synergy emphasises a multidisciplinary approach encompassing the robot's design, functionality, and performance, addressing challenges that may arise in diverse working scenarios.

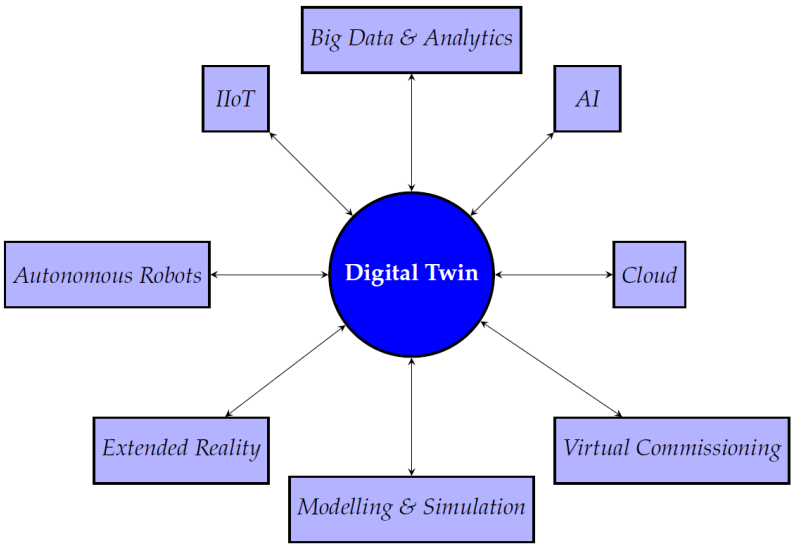


Figure 2. Synergy between DT and other components of Industry 4.0.

This paper's organisation is as follows: Section 2 covers DT and VC applications in robotics, software tools for DT implementation, and the role of XR technologies, particularly MR, in data accessibility and analysis. Section 3 overviews the development workflow for a 3DoF pneumatic cylindrical robot's DT application. Section 4 presents the implementation and validation of a digital twin in AS environment. Section 5 presents the implementation of the same robot in Unity and facilities a holographic approach. Section 6 presents the comparative advantages of both implementations. Section 7 comes with a few conclusions and directions for future research.

2. Related Work

DTs can improve design, production, operation, maintenance, training, and end-of-life administration over the whole lifecycle of a component or system, and [23] provides a comprehensive list of the applications of the digital twin classification throughout the lifecycle. This paper demonstrates that most digital twin applications and use cases belong to "development" or "operations" categories. The second is the main application area, and the function of the digital twin for both cycles received almost no consideration to present DevOps. Considering this gap, the current work is positioned precisely in this area because it includes the VC area after the creation of the virtual model of the robot. Applications in mechatronic systems represent an important category in scientific papers which present VC and DTs. Combining mechatronic components, control software, and digital technologies such as VC and DTs can help enterprises optimise their operations [24–26].

The application of DTs and VC in robotics provides significant benefits in terms of cost savings, improved performance, and streamlined development processes. These advantages have led to the rapid adoption of DTs and VC technologies in the robotics field, making it a top area of interest for both industries and researchers [27–30].

Regarding software platforms and tools available for DT implementation in different architectures, industry and academia have different choices due to the distinct needs and goals [31]. While academia favours more adaptable and configurable tools to serve a wide range of research topics (Matlab, LabView, Automation Studio, Ciros, Ansys, Unity, V-REP, Gazebo), the industry often looks for comprehensive, scalable solutions that can be quickly integrated into existing systems (Siemens PLM Software, FastSuite-2, Dassault's 3D experience, PTC's ThingWorx, AVEVA's digital twin, GE's Predix, RoboDK, CoppeliaSim, VisualComponents, Emulate3D) [1,10,23,28,32].

The huge volume of data around DTs can be difficult for non-expert users to evaluate and analyse. In order to increase the accessibility and usability of the data produced by DTs,

XR technologies are used to bridge the gap between the virtual and real worlds. The potential of combining the strengths of DT and XR (specifically MR, [33]) to create more efficient and effective approaches for various applications, such as design, operation, maintenance, and training, was recognised by industries and researchers, which led to the beginning of the strong connection between these technologies. The benefits of leveraging mixed reality as an effective front–end for digital twin models in a more intuitive, immersive, and interactive manner are presented in [34–38].

In the specialised literature, there appears to be a lack of research exploring the combined approach of implementing digital twins using both Automation Studio and Unity.

3. Materials and Methods

The development workflow of the case study DT application for a 3DoF cylindrical robot type (or RPP robot, which contains three linkages and three joints—one revolute joint that rotates about the base, and two prismatic joints), as presented in Figure 3, is based on three main elements: (1) 3D asset creation; (2) DT and VC application development—the ecosystem around AS; and (3) DT and MR application development—the ecosystem around Unity. The interactions between architectural components, physical and cyber layers, are a part of the DT approach and are realised through the OPC approach, which is usually used in horizontal and vertical communication automation processes because it provides semantic interoperability for all layers.

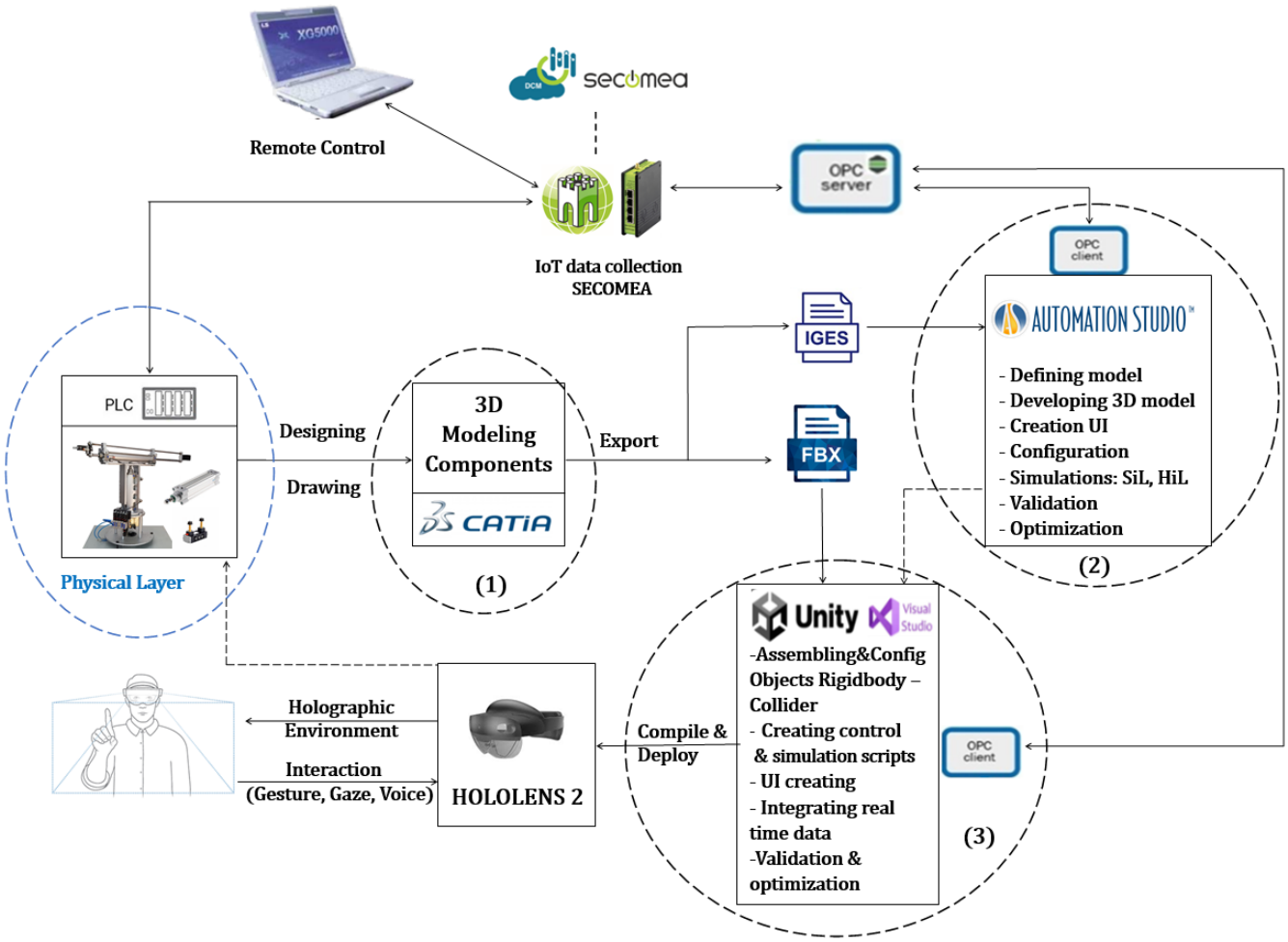
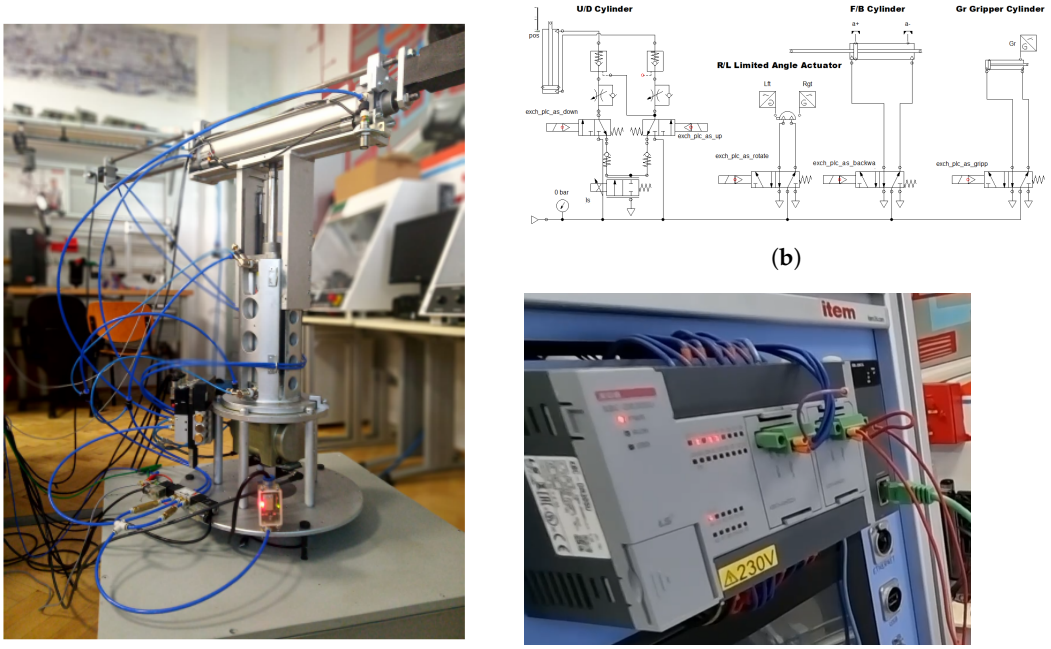


Figure 3. The development workflow of the project.

3DoF Pneumatic Cylindrical Robot

For activities requiring repetitive, accurate motions in a constrained environment, such as pick-and-place operations, assembly tasks, or simple welding tasks, a 3DoF pneumatic cylindrical robot is well suited. The RPP configuration implies that the robot has one revolute joint followed by two prismatic joints. In a pneumatic cylindrical robot, all movements are executed through pneumatic actuators. These actuators convert compressed air into mechanical energy, providing a fast and responsive actuation method. The robot's joints are powered by these actuators, allowing it to perform various tasks within its cylindrical workspace. This robot, as depicted in Figure 4a, is an educational setup from Tech-Con Lab—Department of Automatic Control and Applied Informatics from TUIASI—and used by students to learn about the principles and operation of pneumatic systems, control algorithms, and robotic kinematics.



(**a**) (**b**) (**c**)

Figure 4. The real RPP robot configuration: (**a**) the real 3DoF pneumatic robot; (**b**) the pneumatic circuit diagram of the robot; and (**c**) the PLC of the robot.

The following pneumatic actuators power the robot's joints:

1. Revolute joint (R): a pneumatic rotary actuator—called L/R—(model ARP 055) is used to execute the rotation of the base joint. It allows the robot arm to rotate horizontally at a maximum of 90°, panning within the cylindrical workspace;
2. Prismatic joint 1 (P1): a pneumatic linear actuator used to control the vertical motion of the robot arm—is called U/D. This actuator extends or retracts to raise or lower the arm within the workspace. The model is 41M2P080A0200, a magnetic double action cylinder with Ø80 mm and 200 mm strokes.
3. Prismatic joint 2 (P2): another pneumatic cylinder used to control the robot arm's radial motion is F/B. This actuator extends or retracts to adjust the robot's reach within the cylindrical space. The model is 60M6L063A040, a magnetic bidirectional cylinder with Ø63 mm and 400 mm stroke. It is a bidirectional cylinder because one end contains EOAT—180° angular gripper model CGSN-25 and the other has a counterweight to ensure smooth movement.

The control device for the robot is implemented with a PLC model XBC-DR30SU, which is a compact PLC offering a variety of input and output (I/O) options and communication capabilities. The PLC receives input signals from various sensors (inductive limit

switches for L/R pneumatic rotary actuator and gripper, ultrasonic sensors for the U/D cylinder and magnetic sensors for the F/B cylinder). The PLC sends output signals to the solenoid valves with electric commands and the external servo pilot, causing the robot's joints to move and perform the desired tasks. The rotary actuator uses a bistable valve 5/3 with a central position closed model 338D-015-02 with coils voltage of 24 V DC—which receives the command directly from the PLC output. For the F/B cylinder and gripper valves, monostable solenoid valves 5/2 model 358-015 are used. Control in position for the U/D cylinder is necessary to use a valve 2X3/2 NC model 338D-015-02 and a proportional solenoid valve 2/2 way NC model 21A2KCV15-10 connected to a controller +Smart.

4. Implementation and Validation DT in AS Ecosystem

One of the challenges in using modelling and simulation software, particularly when employing high-level modelling approaches with predefined elements from internal libraries, is the limitation imposed on customisation and flexibility. It is the case with industry applications for building DT models. This becomes evident when a low-level modelling approach is required, which involves building new components or mechatronic systems using CAD data and focusing on assembly and functional interactions between various elements. This is the case for the project from this paper developed in AS (a physics-based modelling and simulation tool with workshops such as pneumatic, hydraulic, electrical, and control), where the analytical and dynamic behaviour of the entire model can provide significant added value for a pneumatic mechatronic system. The process follows a V-diagram, as shown in Figure 5, starting from requirements powerfully linked with the tasks of the pneumatic robot from physical space up to the validation and optimisation of DT through the VC approach.

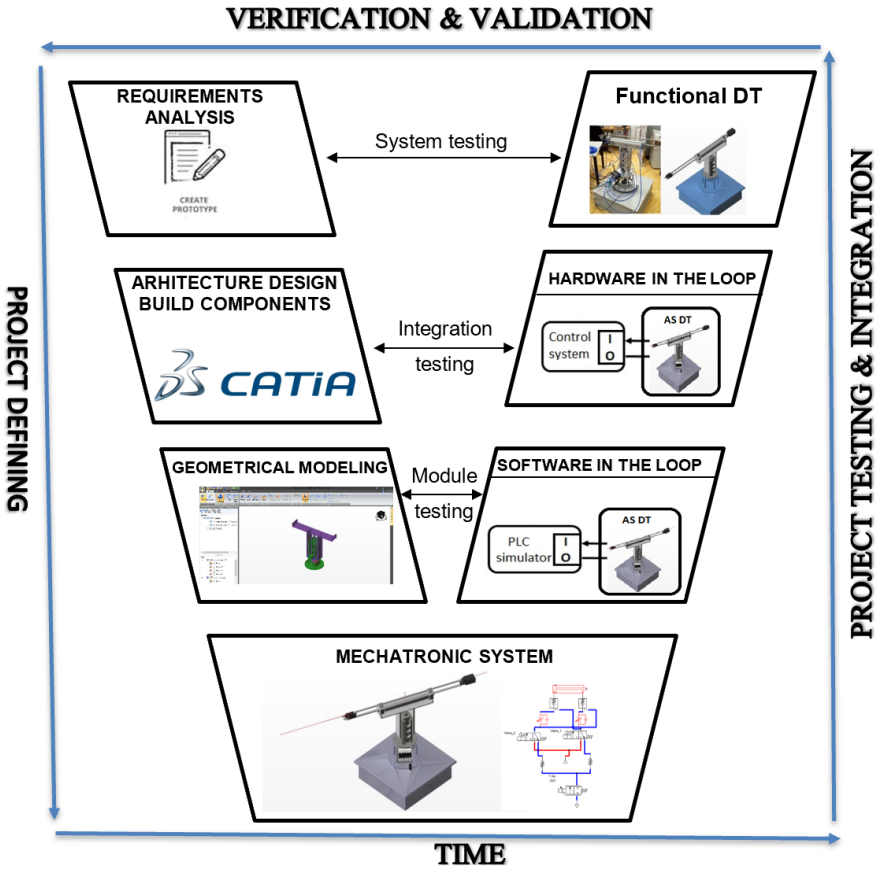
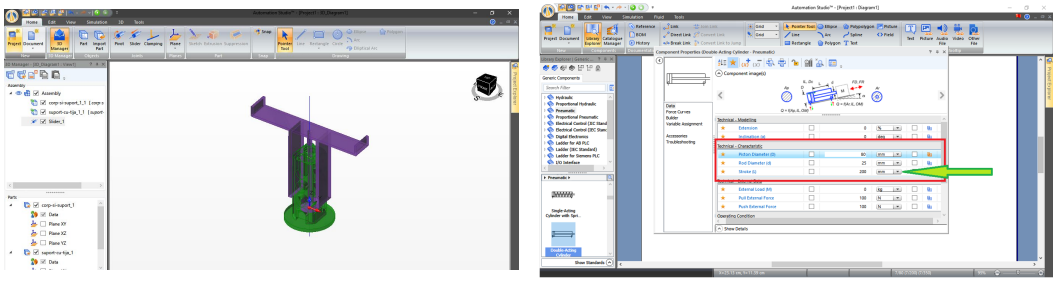


Figure 5. V-model of implementation in AS—reinterpreted from [1].

The 3D model is changed into a simulation model as the first step towards obtaining a digital twin. To accomplish this, 3D models (with an .igs extension) must be imported

into the AS—3D workshop. After, it is necessary to define the required rigid and collision bodies and create the type of displacement for each joint—Figure 6a.



(**a**) (**b**)

Figure 6. The implementation of first prismatic cylinder (U/D) in AS for two approaches: (**a**) the screenshot from the 3D editor workshop where the constraints and the displacement were defined; and (**b**) the screenshot from the pneumatic workshop to customise parameters similar to a real cylinder.

Virtual commissioning can be performed using software-in-the-loop simulation (SIL) or hardware-in-the-loop simulation (HIL). The role of a PLC in the digital twin model is two-fold: it provides real-time data to the digital model and allows for the virtual testing and optimisation of control system logic, notably streamlining the process of virtual commissioning. The PLC simulator from AS is used in the SIL simulation to control the virtual robot. The PLC program is checked and runs in a real PLC linked to the virtual robot developed in AS, which runs on the PC during the HIL simulation. Because the model of PLC is not included in the PLC library from AS, it is necessary to work with a generic PLC that closely matches the specifications of a real PLC from an I/O point of view. The communication configuration between the real PLC and the AS environment uses an OPC server to establish the connection, as presented in Figure 3. Setting up the OPC server software on the computer required configuring communication settings (PLC IP address, communication protocol, and port number) and adding the desired PLC tags/variables to the OPC server.

The realistic behaviour of a DT developed in AS can be attributed to the comprehensive pneumatic workshop offered within the software. This workshop facilitates the creation and simulation of complex pneumatic systems by providing an extensive library of pre-built components, accurate physics modelling, parameter customisation, and the ability to integrate with other domains such as electrical, hydraulic, and control systems.

Detailed models of pneumatic systems can be created using the pneumatic workshop to accurately represent the real-world behaviour. The customisation of various parameters ensures that the digital twin closely matches the characteristics of the physical components. At the same time, powerful simulation and analysis tools enable the visualisation, issue detection, and optimisation of the system's performance.

To have increased this project's complexity in the implementation of the virtual commissioning, the cylinder that lifts the robot chassis (U/D cylinder) was controlled in position because a pneumatic cylinder in a specific position is less usual than the simple extension and retraction control. It is assumed that the pressure drops across the piston are small compared with the supply pressure and may be neglected. Thus, the input for the block diagram of the cylinder is the airflow, and the output is the cylinder stroke, Figure 7. Equation (1) presents the relation between the volumetric flow rate ($q(t)$), piston surface (A), and the rod's velocity $\dot{x}(t)$:

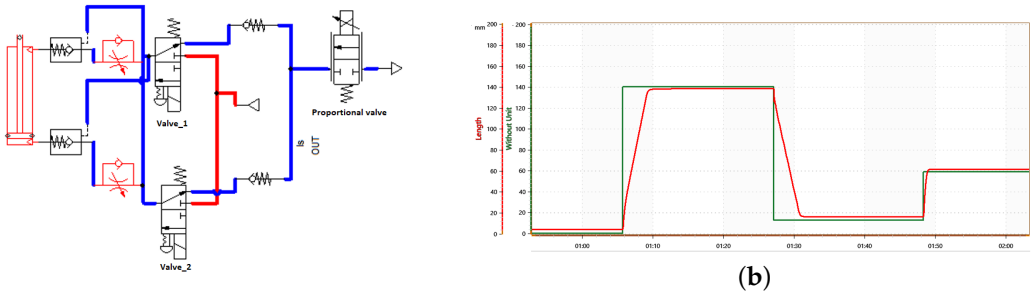
$$q(t) = A\dot{x}(t), \tag{1}$$

from where the cylinder position results from Equation (2):

$$x(t) = \frac{1}{A}\int q(t)\,dt, \tag{2}$$

In AS, a servo pneumatic system was implemented for the physical behaviour of the U/D cylinder, with the main goal of accurately controlling the position of the cylinder's rod. The proportional valve is responsible for managing the airflow exhausted from the cylinder. It allows one to make precise adjustments to the cylinder's motion by varying the airflow according to the control signal from the feedback controller. The process begins when the controller receives a desired setpoint for the position of the cylinder's rod. The ultrasonic position sensor continuously measures the actual position of the rod and sends this information back to the controller.

The controller calculates the error between the desired setpoint and the actual position, using this information to generate a control signal to adjust the proportional valve. As the proportional valve adjusts the airflow exhausted from the cylinder based on the control signal, the cylinder's rod moves towards the desired position. The position sensor keeps providing feedback to the controller, ensuring that the system maintains a closed-loop control configuration. This enables the servo pneumatic system to combine the benefits of a pneumatic, such as high force and robust components, with the precision and accuracy of a closed-loop control system.



(**a**) (**b**)

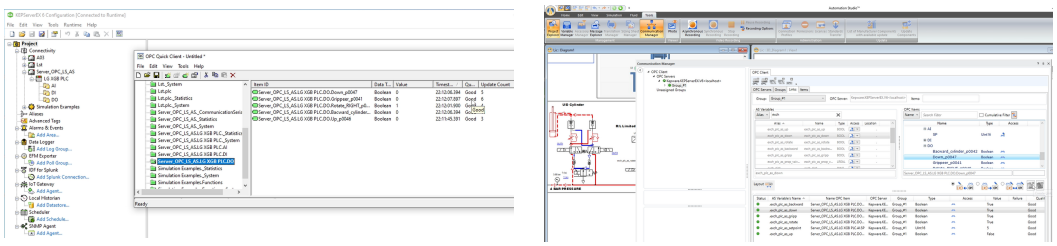
Figure 7. The cylinder (U/D) for two approaches: (**a**) the pneumatic diagram for positioning cylinder in AS; and (**b**) displacement of cylinder for different position references.

The hardware system utilised for conducting the HiL simulation consists of several components, presented in Figure 3. A host computer is included, which runs the DT of the 3DoF robot in an AS environment which the real PLC conducts. The mechanism behind the communication between the mechatronic digital twin and the real PLC is an OPC server. OPC is a popular industrial communication protocol that permits data transmission and compatibility between automation devices and software programs. In this setup, the OPC server (KEPServerEX) links the digital twin and the real PLC in this configuration. Regardless of the implementation specifics or communication protocols the two components use, it offers a common interface that enables seamless communication. KEPServerEX facilitates M2M communications by enabling connections between data values from PLC and AS [39]. The choice of communication protocol is decisive to the successful operation of a DT. The chosen protocol must be well suited to the system's specific requirements, including the network infrastructure, PLC compatibility, real-time performance requirements, etc. Given its robustness and real-time performance, the OPC is an excellent choice for many DT applications, especially mechatronic applications.

Employing an OPC server establishes a vital communication bridge between the digital twin of the mechatronic system and the real-world PLC. Firstly, the OPC server builds a connection with the PLC, enabling a two-way communication pathway. This pathway facilitates the transmission of various data types, encompassing status indicators, sensor-derived measurements, and control commands. The OPC server adeptly converts data originating from the PLC into a format compatible with and readily interpretable by the digital twin. This conversion mechanism accommodates the diversity of data formats and protocols potentially employed by different PLCs, thereby establishing a universal communication language. Subsequently, the OPC server orchestrates the data transfer to the digital twin. Through this exchange, the digital twin becomes a mirror, accurately

reflecting the state of the real-world mechatronic system. Furthermore, the OPC server also relays control directives from the digital twin to the PLC, effectively empowering the digital twin to guide the operational state of the real system. The OPC server diligently ensures rapid data delivery to the digital twin, thereby maintaining congruence between the digital twin and the real system. Hence, the OPC server's role is pivotal in simplifying and streamlining the communication and data translation processes between the digital twin and the PLC. By adhering to the OPC standard, the server guarantees seamless interfacing capabilities with any compliant system. This results in the more efficient, flexible, and robust development and utilisation of digital twins across diverse mechatronic systems.

In the project, communication with the PLC model XBC-DR30SU occurs via an Ethernet communication module, specifically the EMTA configured as a Modbus TCP server. The process involves creating a generic channel (because the model of PLC is not in the standard library of KEPServerEX and adding the device connected through its IP address. Initially, all inputs and outputs are defined in the server (Figure 8a, and subsequently, they are set up in the client within AS. AS has the capability to connect to an OPC Server, which simplifies the process of pairing OPC tags with sensors and solenoid valves from the virtual model (Figure 8b). The DT can effectively communicate with the real-world PLC by establishing this connection. This simulation makes it easy to understand the impact of different parameters (flows, pressures from pneumatic circuit) in the behaviour robot to make the finest adjustments to synchronise the robot 1:1 with its DT. In a pneumatic system, the flow rate significantly affects the speed at which the pneumatic cylinder operates. A higher flow rate means more air can fill the cylinder per unit of time, resulting in faster cylinder actuation and vice versa.



(**a**) (**b**)

Figure 8. Configuration of the OPC mechanism: (**a**) the screenshot from the KEPServerEX—defining the OPC server; and (**b**) the screenshot from the AS—the OPC client linked with the variable from DT.

To ensure that the DT of the pneumatic robot accurately replicates the real-world system, its behaviour must be validated. The validation process entails a series of steps that compare the DT's performance with that of the actual system, ensuring that the DT effectively represents the real-world system's behaviour. In the industrial domain, the accuracy of a DT is of paramount importance, often even more so than in academia. This is because industries rely on DTs to make informed decisions regarding optimisation, control strategies, predictive maintenance, training and safety, and last but not least, customisation and personalisation. All these things have significant financial, operational, and safety implications.

A Systematic Approach Validation Process for DT

A systematic approach validation process for DT involves several interconnected stages that work together to improve the DT's accuracy and reliability:

- **Establish performance metrics**—it is important to establish key performance indicators (KPIs) for the modelled system and prioritise them based on their significance for the specific application. This prioritisation helps focus the validation process on the most critical aspects of the system's performance.
- **Set-up working scenarios**—this step involves defining the working scenarios. A bottom–up approach can be employed, starting with the individual components

of the robot and progressing towards the entire system. This method ensures a comprehensive understanding of the system at each level of complexity.

- **Running scenarios for the pair: the robot and its DT**—the defined scenarios are run for both the robot and its DT, and the performance metrics are recorded. This step provides valuable data about the behaviour of the DT under different conditions, which can be compared with the real-world system performance.
- **Comparative analysis between the real robot's behaviour and that of the DT**—by examining the differences in performance metrics, areas of discrepancy and improvement can be identified. The DT can be considered validated if the discrepancies are within acceptable tolerances. Otherwise, it is necessary to take the next step.
- **Calibration and adjustments of the DT**—adjust the parameters and settings of the DT to match the real-world system. This may include tuning the pneumatic actuator's characteristics, control logic, and environmental conditions. By performing multiple iterations, the parameters are adjusted until the DT accurately represents the real system's behaviour.
- **Resumption of tests from the third step**—once the DT has been recalibrated, the working scenarios are run again with the updated values. This step validates the changes made during recalibration and ensures that the DT more closely represents the real system.
- **Iterate and improve**—in this last stage, the validation of the DT is understood, as is continuously updating the DT with new data from the real system and adjusting its parameters accordingly, ensuring that the DT remains a reliable and effective representation of the real system. This iterative process is vital for the DT to remain a valuable tool for the design, simulation, optimisation, and control strategy testing tasks.

Employing this systematic approach streamlines the validation process for the DT pneumatic robot, ensuring its accurate representation of the real-world system and enhancing its efficacy in design, simulation, and optimisation tasks. Although the validation model has been specifically applied to a pneumatic robot, it can be generalised to accommodate other mechatronic systems with the DTs created in diverse software platforms that encompass physical properties while allowing access to the analytical and dynamic aspects. This versatility expands the scope of the validation process, making it applicable to a diverse array of mechatronic systems featuring DTs.

The sequence corresponding to the graph depicted in Figure 9 was carried out at the macro level for both the robot and its DT.

The robot returns to its homing point when the user presses the "Start" button. In the home pose configuration, the robot has the gripper closed, rotated to the left, completely backward, and has the cylinder responsible for lifting the robot chassis positioned at 10 mm. In the following state of the graph, the robot transitions from its home pose to a new state in which the cylinder responsible for lifting the robot chassis (U/D cylinder) raises it to its maximum height of 200 mm. This movement represents a change in the robot's position and a step towards executing the defined sequence. After reaching the maximum height, the next step in the sequence involves the bilateral F/B cylinder moving forwards. This action represents a further change in the robot's position and continues the execution of the defined sequence.

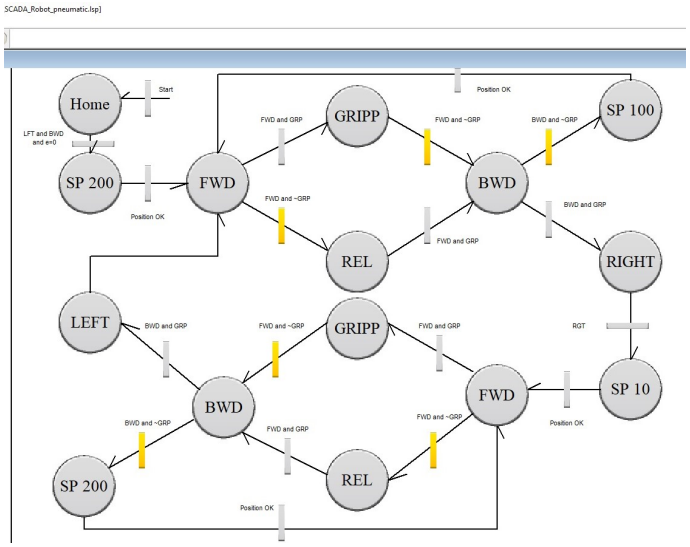
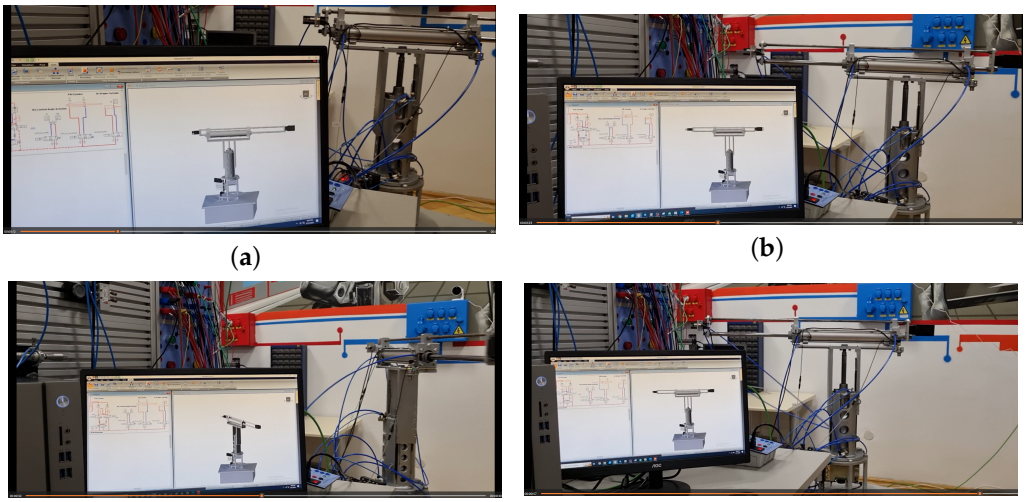


Figure 9. Working scenario implemented for testing.

Figure 10 captures snapshots at various moments during the execution of the working scenario, with the real robot in the background and its digital twin displayed on the monitor in the foreground. These snapshots provide a visual comparison of the robot's behaviour and the digital twin's performance throughout the scenario, further highlighting the accuracy and effectiveness of the digital twin in replicating the real system.



(**a**) (**b**) (**c**) (**d**)

Figure 10. Snapshots at various moments during the execution of the working scenario. (**a**) t = 12 s; (**b**) t = 23 s; (**c**) t = 32 s; and (**d**) t = 57 s.

A rough analysis can be conducted by observing the movements of the real robot and its digital twin in parallel. This qualitative comparison identifies any major discrepancies in motion and behaviour, providing an initial assessment of the DT's accuracy in replicating the real system. This approach is particularly feasible for cylinders with a full stroke, as their motion is more easily observed and compared between the real robot and the DT. In the first three captures from Figure 10, good alignment and synchronisation between the real robot and its digital twin can be observed. However, in the last capture at t = 57 s, it can be noticed that the cylinder in the simulation is lagging behind the real one. This discrepancy indicates that there may be some differences in the parameters or timing between the digital twin and the real system that need to be addressed in order to improve the accuracy of the DT's representation. The first action to be taken is to modify the air flow rate entering the

cylinder, as it is known that the cylinder's speed primarily depends on the airflow rate. By adjusting the airflow rate in the DT's parameters, the simulation can more accurately represent the real system's behaviour, leading to better synchronisation and alignment between the real robot and its DT. This adjustment should be made based on the observed discrepancy and the known relationship between the air flow rate and cylinder speed to minimise the motion difference between the real system and its digital twin. AS provides access to the internal parameters of the flow control valves, as shown in Figure 11. This feature allows users to easily adjust the airflow rate and the other characteristics of the components within the pneumatic system, ensuring that the digital twin's behaviour more closely matches that of the real robot. By fine-tuning these parameters, the user can adjust the speed cylinder.

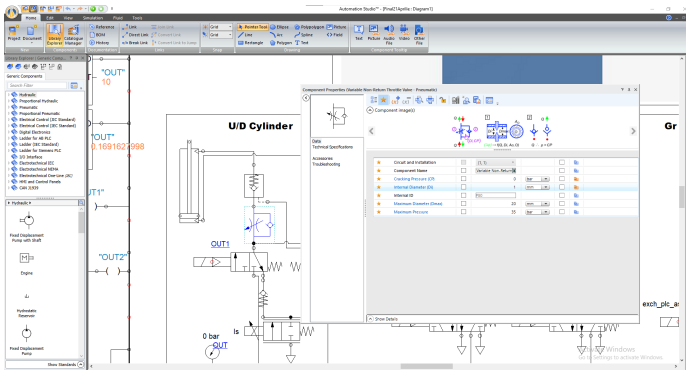


Figure 11. The screenshot from AS—unidirectional flow regulator properties.

Considering the proposed KPIs, a more detailed and quantitative analysis follows to ensure the accurate alignment of the DT with real-world robot performance. During the scenario execution, data are extracted into a graphical format to analyse the real robot's behaviour and DT. This visualisation allows for a more comprehensive understanding of their performance, allowing the comparison and identification of any discrepancies between the two systems.

In Figure 12, the behaviour of the U/D cylinder is represented for both the actual robot (derived from monitoring the PLC programming software for the values received from the ultrasonic distance sensor) and its DT (taken from the position graph specific to the cylinder in AS) for various reference values imposed in the working scenario. This comparison directly assesses the DT's accuracy in replicating the robot's behaviour. It helps identify potential areas for improvement and calibration to ensure a more faithful representation of the natural system.

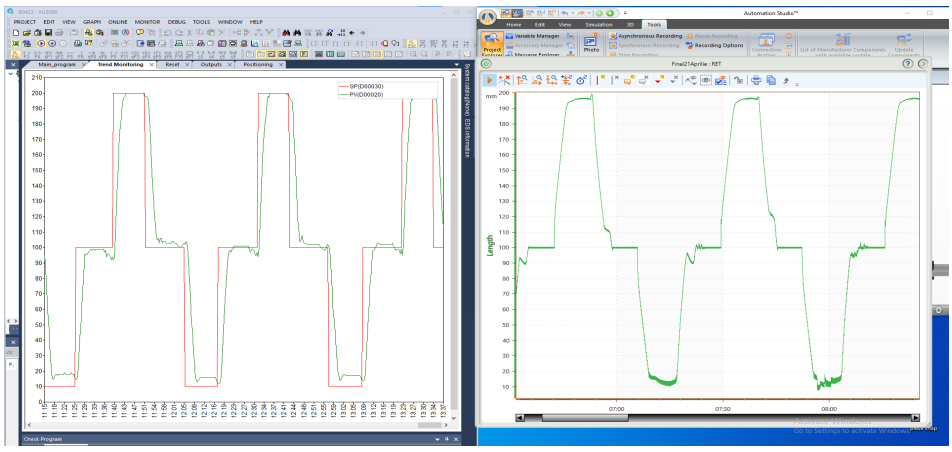


Figure 12. The real and mirrored behaviour of the U/D cylinder.

The refinement and analysis of the two graphs can be conducted within a dedicated computational environment tailored for data processing and visualisation. Widely utilised tools for this purpose include MATLAB or Python and spreadsheet applications such as Microsoft Excel or Google Sheets. Importing the data from both graphs into the selected software allows a comprehensive examination, comparison, and visualisation of the robot's behaviour and DT.

Communication delay, also known as latency, between the DT and the real system is an important aspect affecting the system's performance. Low latency can be a key issue for achieving real-time performance in a HiL simulation. High latency, on the other hand, may lead to a time lag between the execution of control actions on the DT and their implementation on the existing system, potentially affecting the overall system performance and accuracy. The performance of the host computer where the DT from AS is implemented can significantly influence the communication delay or latency. A computer with high performance will be able to run the DT more smoothly, leading to less delay and better real-time performance.

Upon completing the entire process, an operational DT is obtained. To preserve its effectiveness and reliability, continuous updates are imperative. By systematically updating the DT with new data from the real system and adjusting its parameters as needed, the DT remains a precise and trustworthy representation of the real system. This ongoing process of iteration and improvement is crucial for the DT to serve as a valuable instrument while adapting to alterations and evolving demands in the real-world system.

5. Implementation and Interaction with DT in Unity Ecosystem

By leveraging the advantages of the DT implementation using the features provided by AS, one can enhance their understanding and expertise in the system's operation. Transitioning to the Unity platform enables an MR experience to seamlessly integrate the real robot and its digital counterpart. It offers an immersive environment that further strengthens the comprehension of the system's behaviour and functionality. The immersive environment allows the development of unique and personalised experiences with the DT, which involve and engage the user at a deeper level than in the case of traditional methods.

Unity is a powerful game engine and development platform that is primarily used for creating high-quality 3D games and interactive simulations. While Unity is not designed for creating DTs, its 3D modelling and interactivity capabilities can certainly be leveraged for that purpose. DTs are virtual replicas of physical systems. Unity can provide a powerful tool for creating and exploring those replicas in a highly immersive and interactive way, where the external environment can influence DT behaviour. By creating a DT influenced by external factors, it is possible to understand better how the system operates under different conditions and make more informed decisions about its design and operation.

The DT of the pneumatic robot was constructed by importing components from a CAD model with the *.FBX* format and then configuring its rotational or translational movements within the Unity engine. In this process, the CAD model was likely converted into a series of meshes and materials that were imported as `GameObjects` in Unity. Such `GameObjects` were then assembled and configured to create the DT of the robot, including its movements and interactions with the environment. The game engine comprises numerous working contexts, or "scenes"—as shown in Figure 13—where the DT is positioned and configured according to the intended simulation-related parameters. The `GameObjects`, which has been put on the robot, can later change a wide range of properties, including the location, orientation, size, weight, material, and animations. The `GameObjects` is then animated using *C#* scripting. *C#* provides a clear, powerful, and robust environment to develop the required codes. Nonetheless, it also has the advantage that most VR, AR, and MR hardware-related packages are already available in this programming language.

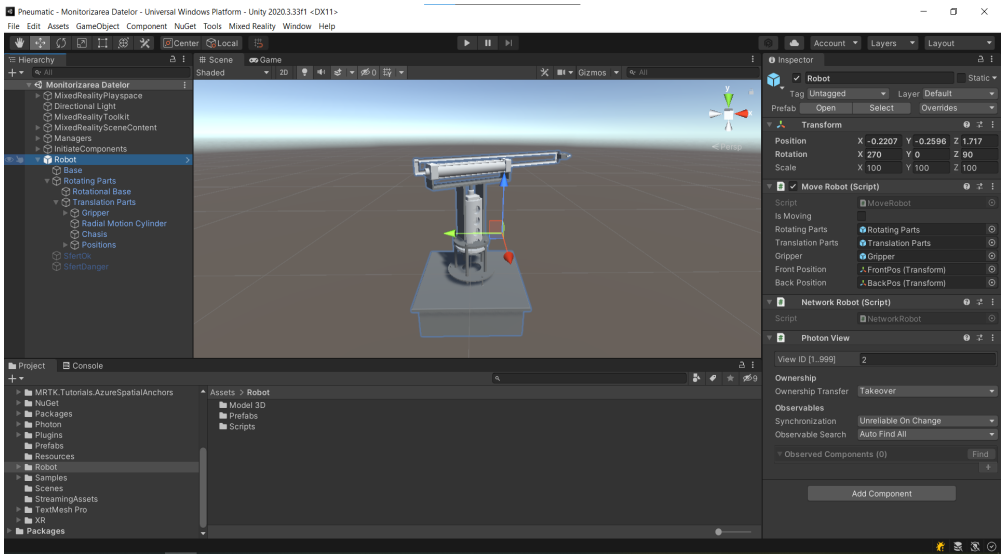
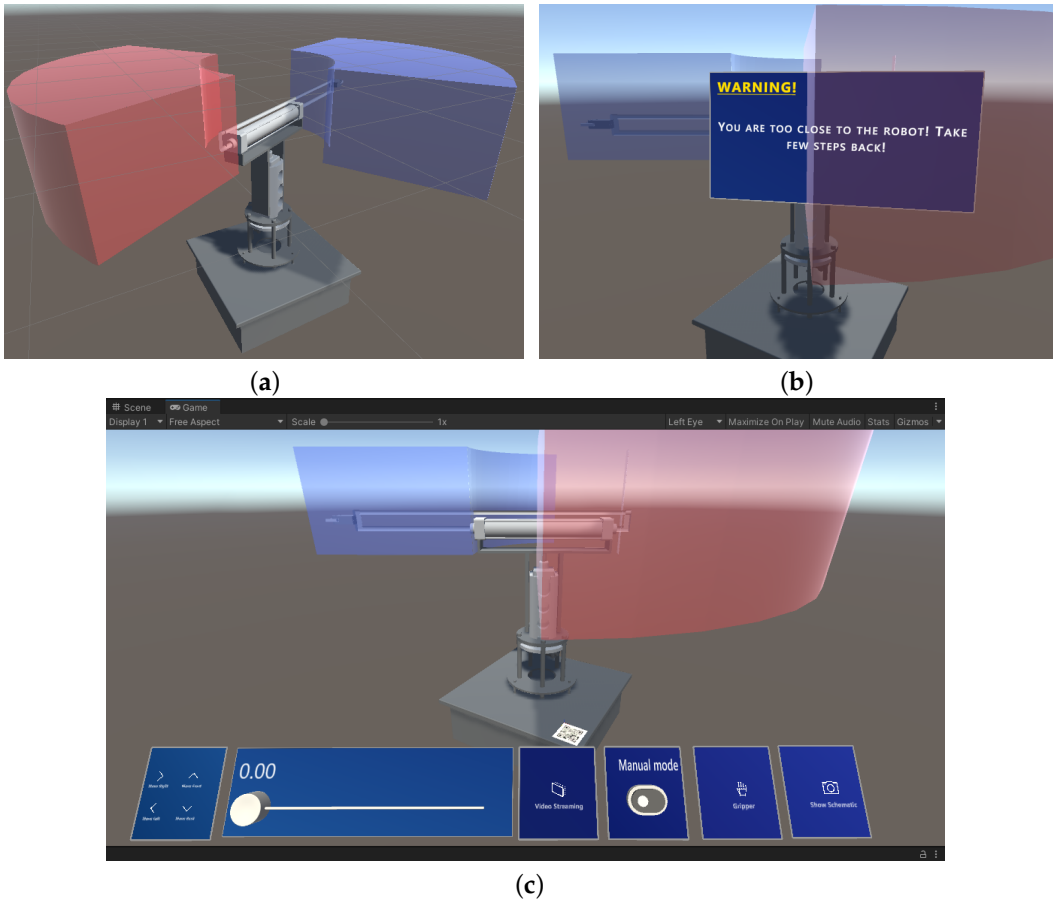


Figure 13. The screenshot from the scene in Unity 3D and the proprieties of the robot's DT.

In mixed-reality experiences involving real-world robots, safety is a critical consideration, especially in the case of non-collaborative robots. The DT representation of the robot and its environment can be used to create virtual boundaries and limits corresponding to the robot's physical workspace and safety zone, Figure 14a. These virtual boundaries and limits were programmed based on the movement of the real-world robot, ensuring that they accurately represent the current workspace and safety zone. The working space for the cylindrical robot is a hollow quarter-cylinder with the height determined by the stroke of the U/D cylinder. The difference between radii equals the stroke of the second prismatic cylinder (F/B). The user is alerted when they are about to enter a potentially dangerous zone. This is performed by displaying a warning message through haptic feedback, Figure 14b. At the same time, when the user enters these dangerous areas, the real-world robot is programmed to stop its operation until the user automatically exits the area. Figure 14c depicts the menu that accompanies the robot in MR.

5.1. Defining Moving

The robot's DT can perform various movements such as the real robot using the `RoboMove` *C#* script. This script defines the behaviour of a robotic arm with rotating and translating parts in Unity. It includes several methods for initiating movements in different directions, moving the arm's gripper to specific positions, and updating the arm's position in the scene. The script includes an enumeration called `Position`, which represents the two possible states for the arm's rotating part: `Right` and `Home`. The class `MoveRobot` has several public methods such as `StartMovementFront()`, `StartMovementBack()`, `StartMovementRight()`, `StartMovementLeft()`, `TranslateUpDown()`,`MoveFront()`, `MoveBack()`,`MoveRight()`, and `MoveLeft()`. The class also has a public variable, *isMoving*, a flag that prevents the robot from starting another movement before the current movement is completed. The script defines the actuator ranges, including the maximum and minimum radii, angle, and height, which control the arm's movements. All these values are in link with the values of strokes of pneumatic actuators. The `Start()` method initialises variables and sets the arm's initial position to "Home"—similarly to the position defined for the robot in the scenario from the HiL simulation from AS. Because of the rotation and translation moving out of the *F*/*B* cylinder, it is necessary to define the equivalent end-of-stroke sensors (for this, the *Position* flag is necessarily near *Home*). The `Update()` method checks for movement initiation flags and calls the appropriate movement method based on which flag is set.



(**a**) (**b**)

(**c**)

Figure 14. The screenshot of the MR application in HoloLens 2: (**a**) the working space—a hollow quarter-cylinder (blue colour space)—and the predefined safety zone linked to the F/B cylinder which has a bilateral construction (red colour space); (**b**) intrusion incident into the safety area; and (**c**) holographic menu that accompanies the robot in the scene: push buttons—MoveRight/MoveLeft/MoveFront/MoveBack/VideoStreaming/Manualmode/Gripper/ShowSchematic and a slider for lifting the robot chassis.

In Listing 1, the constant *desiredDuration* corresponds to the time required by the real robot for the L/R actuator to complete its entire travel. This is closely related to the flow rate and pressure values at the air inlet port of the real actuator. The mathematical function *Mathf.SmoothStep* models cylinder cushioning, which is needed to lower the speed of the cylinder before it reaches the end cap. The function in Unity is used to interpolate between two values smoothly, creating a natural transition ideal for animations or object movements. Based on a cubic formula, this function generates a smoother and more natural result than simple linear interpolation.

5.2. Pose of the DT in Virtual Space

In the context of real-world interaction in the mixed-reality application for the non-collaborative robot, the dimensions of the DT must match the dimensions of the real-world robot so that the safety zones and warnings are consistent for users. Therefore, the DT must be accurately configured with the exact dimensions and precisely anchored to the position and orientation of the real robot.

If the CAD files are incomplete or inaccurate, the DT may not be a faithful replica of the real-world object. The first option is to modify the file in CAD software and reimport it into Unity. However, Unity provides tools for accurately measuring distances, angles, and other properties of `GameObjects` that can help adjust the DT. If it is accurately configured with the same dimensions, the next step is to anchor the hologram to the space so that there is

a perfect overlap between the robot and the hologram. Since HoloLens 2 comes with the native ability to read QR codes, the positioning and orientation of the hologram will use the position of the QR code on the real robot, which is fixed to the robot pedestal. Using the add-on *Adjust Pivot* in Unity positions the reference point of the robot model at the position of the QR code, and this allows the position of the QR code to be used as a reference point for the position and orientation of the robot in the Unity scene. Doing so ensures the accurate overlapping of virtual objects relative to real objects in the environment.

Listing 1. MoveRight method from RobotMove class.

```
public void MoveRight()
//MoveRight method
{
    if (currentPositon != Position.Right)
    {
        isMoving = true;
        elapsedTime += Time.deltaTime;
        var percentageComplete = elapsedTime / desiredDuration;

        rotatingParts.transform.localRotation = Quaternion.Lerp(Quaternion.
            Euler(rotatingParts.transform.localEulerAngles.x, rotatingParts.
            transform.localEulerAngles.y, rotatingParts.transform.
            localEulerAngles.z),
            target, Mathf.SmoothStep(0, 1, percentageComplete));
        if (rotatingParts.transform.localRotation.eulerAngles == target.
            eulerAngles)
        {
            elapsedTime = 0;
            currentPositon = Position.Right;
            startRight = false;
            isMoving = false;
            Debug.Log(currentPositon);
        }
    }
    else
    {
        startRight = false;
    }
}
```

The implementation of this functionality in the actual project, which starts from scanning the QR code, identifying the robot's DT and positioning the robot's hologram was achieved through several scripts grouped in the QRTracking namespace. The `QRCodesManager.cs` script initialises and manages the QR code tracking system, which uses the Microsoft.MixedReality.QR library to detect and track the QR codes in the scene. The `QRCodesVisualizer.cs` script visualises the detected QR codes in the scene, using a specified prefab object as a template for each QR code object. It subscribes to various events raised by the `QRCodesManager` script, such as `QRCodeAdded`, `QRCodeUpdated`, and `QRCodeRemoved`, which allow it to add, update, and remove the QR code objects in response to changes in the tracking data. The `QRCode.cs` script is attached to each QR code object and is responsible for displaying the relevant information about the QR code, such as its ID, text, physical size, and timestamp. It also handles user input, such as clicking on the QR code object, and launches the associated URI if it is valid. The script also contains a code for positioning a virtual pneumatic object (the DT of the robot) in the scene, which represents the position and orientation of the QR code relative to the real-world robot—as shown in Figure 15.

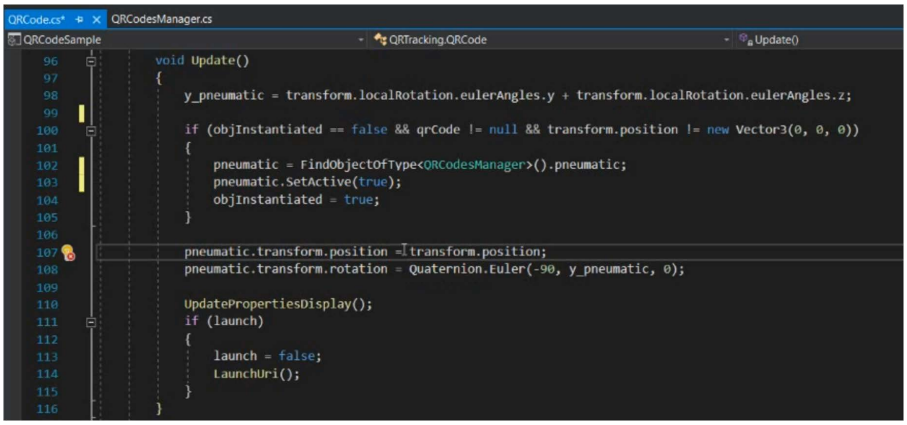


Figure 15. The `Update()` method is responsible for matching the position and orientation of the DT robot with the QR code object.

5.3. A Virtual Space for Multiple Users to Interact with Robot

Enabling multiple users to simultaneously view and interact with a robot in a shared virtual environment can effectively showcase its capabilities and facilitate collaboration, especially in training or educational contexts where the use of a DT is required. This was made possible using Photon Unity Networking (PUN), a Unity plugin that enables network synchronisation and communication for multiplayer games and applications [40]. By broadcasting data from the PUN server, the robot's and users' DT are synchronised across all connected devices. Any user input or interaction with the robot triggers an RPC event callback on all associated devices via the PUN server. This ensures that the robot's state is synchronised across all participants in the shared environment.

The process of integrating the Photon SDK for Unity in a DT project to establish a shared virtual environment is depicted in Figure 16. The procedure involves importing the SDK, configuring the server settings, creating a Singleton pattern-based NetworkManager to facilitate connections to the Photon server, adding a lobby to manage room creation and joining, instantiating a `Robot` prefab equipped with pertinent components to enable user interactions, devising a `Robot` prefab script to manage object parenting and ownership transfer.

The implementation of the right side of the diagram in Figure 16 is shown in Figure 17.

The `MoveRobot` script discussed in Section 5.1 allows for the precise and controlled movement of a robotic arm in a Unity scene, with the ability to synchronise movement across multiple instances of the game object using Photon networking.

5.4. The System for Remote Support

By incorporating shared DT technologies, teams can collaborate remotely to solve challenging problems in real-time without being nearby. This is feasible in both industry and academia. Leveraging the features of shared DT technology, a video-streaming concept incorporating distinct roles for technical and non-technical individuals was designed to further improve collaboration and learning experiences. In situations where non-technical individuals need to execute intricate tasks, real-time expert guidance via videostreaming can be invaluable. The expert (professor) can observe the non-technical (student) individual's actions and provide immediate feedback, ensuring that the task is performed correctly and safely.

Figure 18a presents the user interface menu, where users are prompted to select their role as either "technician" or "non-technician." This simple, intuitive menu is designed to quickly identify and categorise users based on their technical expertise, enabling a more streamlined and targeted communication experience within the system. By making this selection, users are guided to the appropriate functionality tailored to their role, ensuring that the subsequent interactions are suitable for their level of knowledge and experience.

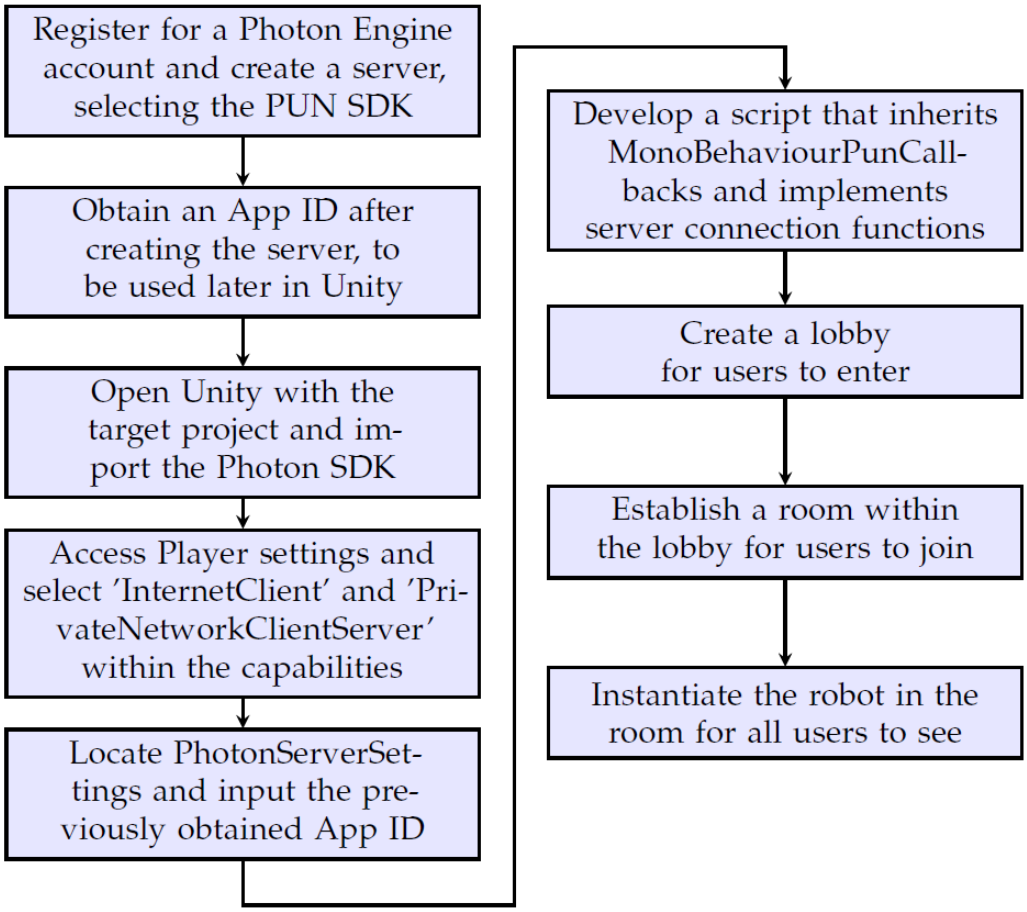


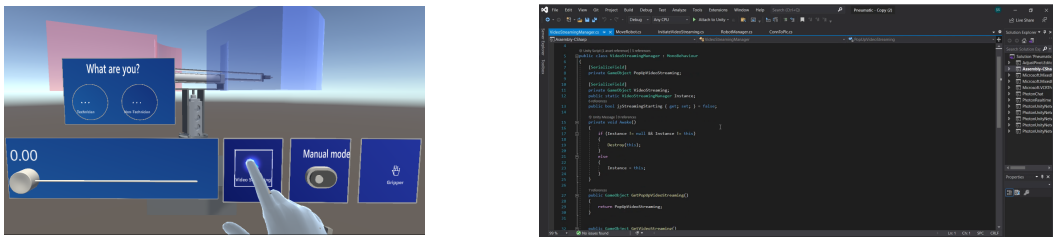
Figure 16. Procedure for developing an application with PUN.

`MixedReality-WebRTC` was used for the implementation with a DSS server made in NodeJS. The server allows the communication of packets between glasses. Two instances with different IDs must be created for the network to be able to detect them. Communication can only be achieved if the two instances are connected to the server. Creating a manager with a script (in Figure 18b) to return the pop-up with the choice of role and the window where the received image will be. This manager will have a single instance; thus, it can access the objects returning in other scripts. The system establishes an audio and video connection between technicians and non-technicians, allowing for the transmission of video and sound feeds between both parties. In the implemented set-up, the technician can view the non-technician's perspective via video, while the non-technician can hear the technician's instructions through audio. This one-way video and two-way audio setup ensure privacy for the technician (which may be the case when the technician uses a laptop, not HMD) while still enabling them to offer valuable assistance.

Figure 19 shows sequences from an experiment in which two operators interact with the robot's hologram synchronised with the real robot. The film is integrated into the PIP technique so that it is possible to see two perspectives: the real one and the perspective seen by the operator who chose their role as a technician. The projection capture on the technician's HoloLens uses the facilities offered by mixed-reality Capture—Microsoft.

```
Assembly-CSharp                                          NetworkManager
1  using Photon.Pun;
2  using Photon.Realtime;
3  using System.Collections;
4  using System.Collections.Generic;
5  using UnityEngine;
6
   Unity Script (1 asset reference) | 1 reference
7  public class NetworkManager : MonoBehaviourPunCallbacks
8  {
9      private TypedLobby customLobby = new TypedLobby("robotLobby", LobbyType.Default);
10
11     private Dictionary<string, RoomInfo> cachedRoomList = new Dictionary<string, RoomInfo>();
12
13     private bool isFirst = false;
14
15     public static NetworkManager Instance;
       Unity Message | 0 references
16     private void Awake()
17     {
18         if (Instance != null && Instance != this)
19         {
20             Destroy(this);
21         }
22         else
23         {
24             Instance = this;
25         }
26     }
27
       Unity Message | 0 references
28     private void Start()
29     {
30         PhotonNetwork.ConnectUsingSettings();
31     }
       3 references
32     public override void OnConnectedToMaster()
33     {
```

Figure 17. C# script `NetworkManager`—implementing a lobby defining a dictionary for all rooms from the lobby and connection of users through AppID.



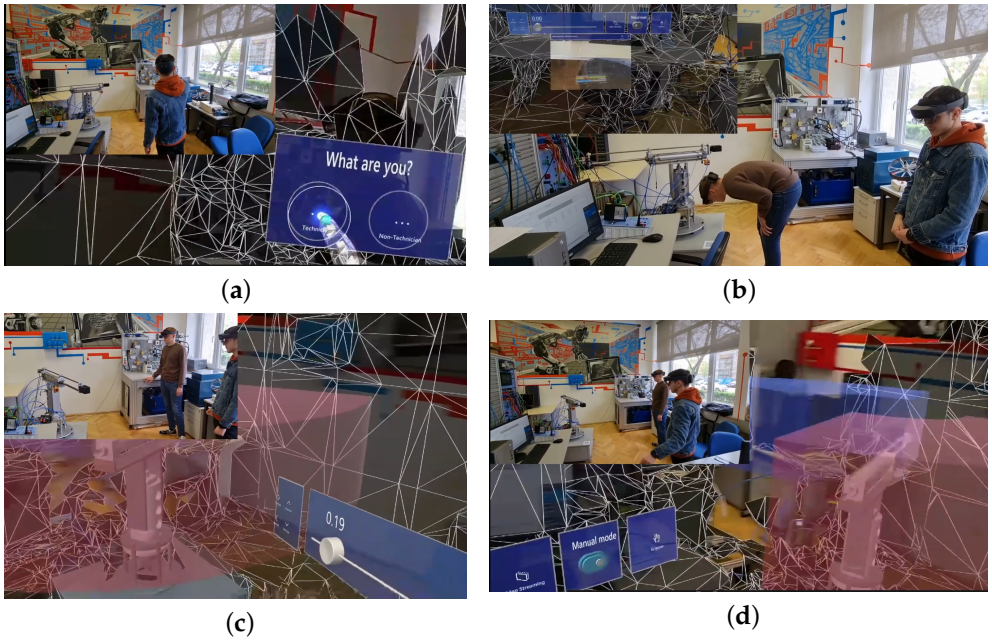
(**a**) (**b**)

Figure 18. Remote assistance platform that enables the technician and non-technician to communicate effectively: (**a**) the choosing of the role; and (**b**) the implementation by the manager.

5.5. PLC Integration in Unity Ecosystem

In order to connect the DT of the robot to the real PLC, a remote connection was created using the `GateManager` facility from Secomea to link the laptop with the PLC. The laptop was configured to redirect any requests received on a port 502 to the PLC. This enabled the direct transmission of information from the HMD to the PLC registers through the laptop. `NModBus` was used for communication with the PLC.

To establish a connection with the PLC, the method `StartConnection()` in the `ConnToPlc` script was marked with [PunRPC], indicating that it is an RPC method that can be called by all users. Similarly, all methods marked with [PunRPC] can be forced to be called by all users. The function `CloseConnection()` in the `ConnToPlc` script was used to close the connection.



(**a**) (**b**) (**c**) (**d**)

Figure 19. Snapshots at various moments during the execution of the working scenario from MR application in HoloLens2 and real actions: (**a**) choosing the technician role; (**b**) scanning the QR code by a non-technician under the careful observation of the technician through the video streaming facility; (**c**) the positioning of the cylinder that raises the robot chassis to 19% of the stroke using the slider; and (**d**) the technician takes control. It can be seen that both safety zones on the hologram are superimposed in the space above the real robot.

When the manual button is pressed, the `Toggle()` function in the `ConnToPlc` script is called. If the connection is successful, the button turns blue, but there is a chance it will turn blue even if the connection is not successful, and the button will need to be pressed again. The function SetToggleToOthers visually sets the button as pressed for the other participants.

After the connection has been established, the position of the real robot is queried from the PLC so that the hologram of the robot can be positioned accordingly. The `Update()` function in the `RobotManager` script is called at each frame to check whether the robot is in the scene. If the robot appears, it is assigned to a variable, and if the manual button has been pressed and activated, the real robot's position is queried and synchronised with the hologram, and the synchronisation flag is set to true. All commands are sent to all other participants in the room, so their hologram robot should move accordingly.

To move the robot in manual mode, when it pushes the `MoveRight` button, the photon calls the `OnMethodCalledRight()` method, which calls another method `MoveRight()` from `ConnToPlc`. Through the `PhotonNetwork` mechanism, the `OnMethodCalledRight()` method sets the parameter to "true" for all participants, which checks whether the action comes in manual mode and if the movement action comes from pressing the button. If these conditions are met, the PLC receives a signal, and the real robot executes the movement to the right as shown in Figure 20.

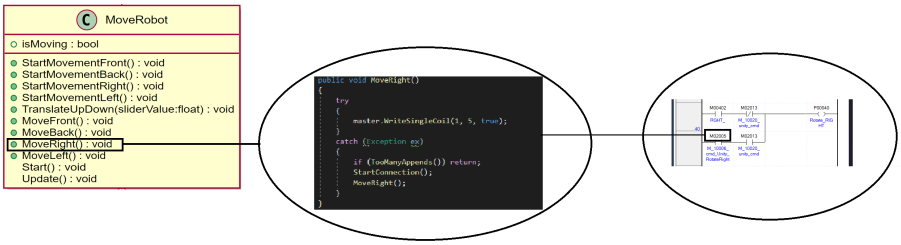


Figure 20. The mechanism for the right robot movement began with Unity—with the textttMoveRobot class, which includes the `MoveRight()` method, and from here, set the M2005 Modbus address as responsible for activating the output of the PLC that energises the solenoid valve coil via the *ModbusBitWrite* bridge.

5.6. The PlantUML Diagram

Figure 21 visualises the system's structure and behaviour, enabling a deeper understanding of the DT's robot project in the Unity development environment. The relationship between *MonoBehaviour*, *MonoBehaviourPun*, and the other classes in your *PlantUML* code is based on inheritance. Custom classes inherit from *MonoBehaviour* or *MonoBehaviourPun* to access the respective features provided by the Unity engine and Photon Unity Networking library, enabling them to interact with the Unity scene and synchronise networked *GameObjects* in a multiplayer setting.

While only some classes have been examined in detail in the article thus far, this overview provides a general understanding of each class's roles in the development and interaction with the robot's DT from this implementation. The diagram effectively illustrates how the classes interact with one another, either through inheritance or other relationships, and the methods and properties associated with each class.

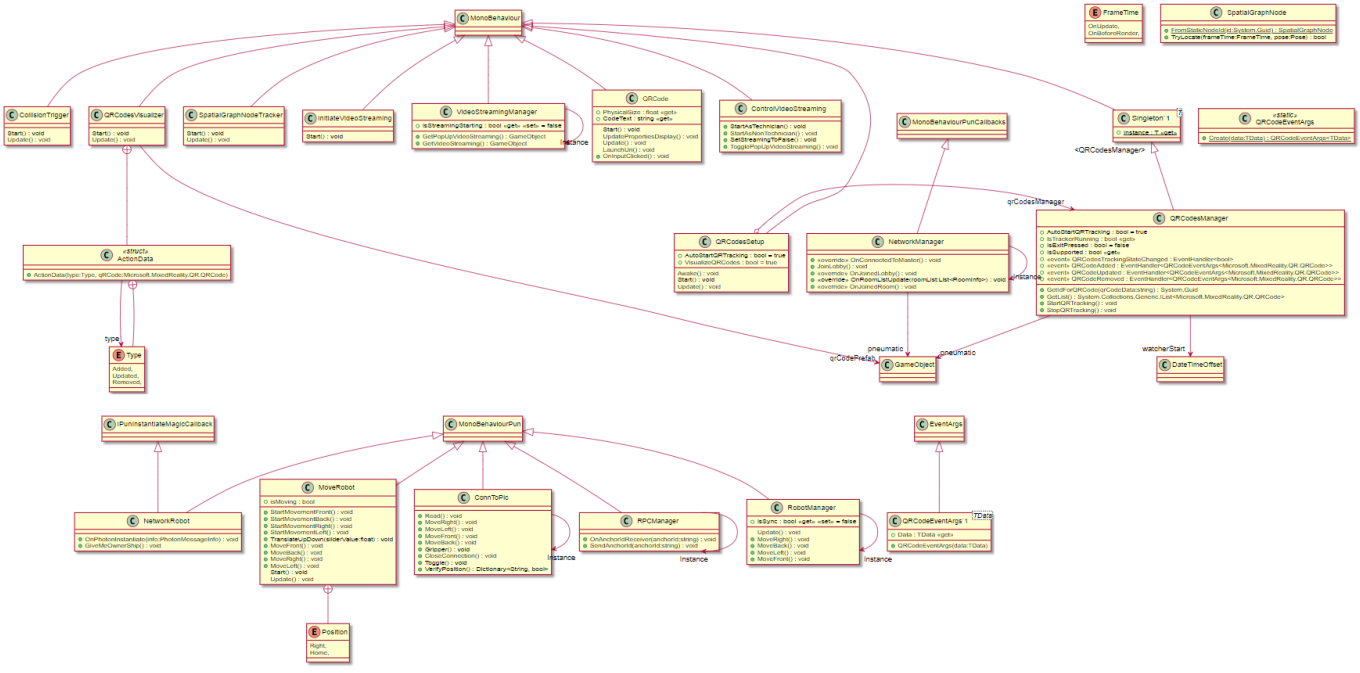


Figure 21. The PlantUML diagram of the pneumatic robot project in the Unity development environment.

5.7. The Challenges of Using Unity in DT

While the Unity Engine boasts an excellent platform for visualising and interacting with digital twins (DTs) using a variety of HMDs such as Microsoft's HoloLens 2, it presents some challenges. Primarily designed as a game engine, Unity might not natively support complex engineering simulations, necessitating the integration of external libraries or tools. This increases the complexity of the DT development process within Unity.

Moreover, Unity's steep learning curve may pose a barrier, especially for those without a computer programming background, as scripting in languages such as C# are often a requirement for Unity applications.

Unity can also encounter performance limitations when dealing with highly detailed DTs or data-heavy real-time systems. The task of rendering sophisticated 3D models or conducting extensive real-time simulations might strain Unity's capabilities, particularly when hardware resources are limited. Additionally, Unity may face compatibility issues when interfacing with other industry-standard hardware or software tools. For instance, establishing real-time communication between Unity and specific industrial controllers, such as PLCs, could require additional middleware or customised development.

Furthermore, managing large volumes of real-time data, a cornerstone in DT operations, may necessitate a sophisticated data processing system to guarantee the prompt and accurate updating of the DT.

Nevertheless, despite these challenges, the Unity community continuously evolves and develops solutions. This continuous advancement helps to uphold Unity's position as a valuable tool in the ever-growing field of DTs development.

6. Discussion

The development of DTs for mechatronic systems can be approached differently depending on the software used. AS and Unity each have unique strengths and challenges, which this article aims to compare in creating DTs for the same mechatronic system.

In AS, the operating characteristics of pneumatic equipment, such as actuators, valves, and sensors, are mainly predefined. These predefined elements make it easier for students, engineers and developers to create realistic simulations of pneumatic systems, as they can directly use the built-in components and characteristics. AS is specifically designed for simulating, designing, and validating complex automation systems, streamlining the modelling process in a particular 2D environment.

On the other hand, Unity is a general-purpose game engine and development platform, which means that it is not tailored for simulating pneumatic systems out of the box. In order to create a DT in Unity, developers need to implement various mathematical functions, physics simulations, and other techniques to accurately model the behaviour of pneumatic components. This inconvenience can be more challenging and time-consuming, as developers have to create these models from scratch. Despite the challenges, Unity's flexibility and extensibility can be advantageous in some situations. For instance, developers can create custom visualisations, integrate the DT with other systems, or even deploy the simulation to various platforms, including virtual and augmented reality environments. This level of versatility allows for more creative and innovative applications, although it may come at the cost of a steeper learning curve and increased development time.

One potential approach to overcome these challenges is to leverage the knowledge gained from working with AS. By initially using AS, the persons involved can understand the underlying phenomena and the behaviour of pneumatic components. This knowledge can be transferred to the development process in Unity. Once the developers in Unity have a clear grasp of the requirements and the desired dynamic behaviour of the system, they can more effectively search for and implement various techniques and tricks to achieve realistic simulations.

This sequential approach can help bridge the gap between the specialised knowledge of pneumatic systems and the flexible development environment offered by Unity. By leveraging the expertise from both domains, creating high realistic and accurate DTs is possible while benefiting from Unity's versatility and creative potential.

The DT model's complexity affects the latency, so more intricate and detailed models necessitate more computational resources and processing time, increasing the latency. On the other hand, simpler models, although they might reduce the latency, may also compromise the accuracy of the DT's representation of the physical system. Experiments from this project with the implementation of DT in AS and Unity showed differences in

latency. Specifically, the DT implemented in AS exhibited more delay than Unity. This observation can be attributed to the different computational demands of these two environments, emphasising the simulation environment's role in influencing latency. A relevant trade-off arises in this context: the balance between the fidelity of the model representation and real-time performance. Reducing the complexity of the model can decrease the latency, improving the real-time representation. However, this simplification can also reduce the accuracy of the digital twin. Therefore, it is important to consider this trade-off depending on the specific requirements and goals of the project. the impact of latency on the DT's performance reinforces the need for a balanced approach in DT development. Decisions regarding the model complexity, choice of simulation environment, and computational resources should consider both the desired accuracy and real-time performance of the DT.

In summary, the choice between AS and Unity for developing the DTs of pneumatic systems depends on the specific requirements and goals of the project. The AS platform offers a more streamlined and specialised approach, while Unity provides greater flexibility and versatility, albeit with a potentially higher degree of complexity. The comparison is a valuable resource for engineers, developers, and researchers looking to create DTs for mechatronic systems, helping them make informed decisions about the best tools and techniques for their projects.

The advantage of using a hologram projected by a head-mounted display (HMD) makes implementing a DT in Unity well worth it, especially when the DT has a solid connection to the actual dynamic behaviour of the process. By employing HMDs such as the HoloLens 2, students and professionals can interact with the DT in a highly immersive and intuitive manner, providing them with a unique learning experience. This immersive experience can enhance the understanding and retention of complex concepts related to mechatronic systems, as users can directly visualise and manipulate the DT in a three-dimensional space. Furthermore, holograms can foster collaboration among students and professionals, as they can share their virtual environment and work together to understand and improve the DT's behaviour.

7. Conclusions

The article highlights the differences and potential advantages of developing DTs in AS and Unity, showcasing the challenges and opportunities inherent in each development environment. The approach is multidisciplinary (automation and computer science, mechanical engineering, electrical engineering, and human–computer interaction), combining knowledge and methods from various fields to develop the efficient DT of the 3DoF pneumatic robot. By integrating expertise from both AS and Unity, this paper showcases the potential of interdisciplinary collaboration in addressing the complex challenges and achieving innovative solutions in DT development. The paper effectively amalgamates the best of both worlds by creating an initial, accurate DT in AS and then expanding and enhancing it in Unity. AS was used for accuracy, clarity, and reliability in simulating pneumatic systems and was exploited by Unity for its versatility, creativity, and capacity to deliver an immersive, interactive experience. This combined approach results in a DT that accurately represents its real-world counterpart, offers improved user engagement, and has broader applicability.

One future research direction is to investigate the possibility of creating hybrid frameworks that combine the strengths of both AS and Unity. Such frameworks allow for the streamlined development of DT with a focus on pneumatic/hydraulic systems while also providing the flexibility and versatility of Unity. This approach could lead to more efficient development processes and novel applications.

The experience gained from developing a digital twin in both Automation Studio and Unity can be highly valuable for students and professionals alike. By working with these two platforms, learners can comprehensively understand the modelling and simulation process and the challenges and solutions unique to each environment. This immersive

experience can enhance the understanding and retention of complex concepts related to mechatronic systems, as users (especially students who belong to the category of digital natives) can directly visualise and manipulate the digital twin in a three-dimensional space.

An accurate DT enables industries to minimise costly downtime, reduce waste, improve efficiency, and ensure product quality. Moreover, it allows for the early identification and mitigation of potential issues, resulting in increased reliability and reduced maintenance costs. In safety-critical applications, an accurate DT can be vital in preventing accidents and ensuring the well-being of workers and the environment.

Funding: The research leading to these results has received funding from the European Union's HORIZON-CL4-2022-TWIN-TRANSITION-01-01 research and innovation actions under grant agreement R3GROUP Project GA no. 101091869, and by a research grant of the TUIASI, project number GI-TD-DigitAll-7/2022.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

TUIASI	Technical University "Gheorghe Asachi" from Iasi
DT	Digital Twin
CPS	Cyber-Physical Systems
I4.0	Fourth Industrial Revolution, Industry 4.0
IT	Information Technology
IoT	Internet of Things
IIoT	Industrial Internet of Things
AI	Artificial Intelligence
DevOps	Development and Operational Cycles
XR	Extended Reality
MR	Mixed Reality
VC	Virtual Commissioning
PLC	Programmable Logic Controller
RPP	Revolute Prismatic Prismatic
OPC	Open Protocol Communication
DoF	Degree of Freedom
AS	Automation Studio conceived by Famic Technologies Inc.
EOAT	End of Arm Tooling
M2M	Machine-to-Machine
SiL	Software in the Loop Simulation
HiL	Hardware in the Loop Simulation
PUN	Photon Unity Networking
RPC	Remote Procedure Call
HMD	Head-Mounted Display
HL	HoloLens
URI	Uniform Resource Identifier

References

1. Dosoftei, C.C.; Lupu, A.; Pascal, C.M. A new approach to create a realistic virtual model of a cylindrical robot using Automation Studio. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *591*, 012078. [CrossRef]
2. Liagkou, V.; Stylios, C.; Pappa, L.; Petunin, A. Challenges and Opportunities in Industry 4.0 for Mechatronics, Artificial Intelligence and Cybernetics. *Electronics* **2021**, *10*, 2001. [CrossRef]
3. Gerber, A. What Does Industry 4.0 Mean? The Definition of Digitization. Available online: https://www.wfb-bremen.de/en/page/bremen-invest/what-does-industry-40-mean-short-definition (accessed on 10 December 2022).

4. Grieves, M.; Vickers, J. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*; Springer International Publishing: Cham, Swizerland, 2017; pp. 85–113. [CrossRef]
5. Park, J.S.; Lee, D.G.; Jimenez, J.A.; Lee, S.J.; Kim, J.W. Human-Focused Digital Twin Applications for Occupational Safety and Health in Workplaces: A Brief Survey and Research Directions. *Appl. Sci.* **2023**, *13*, 4598. [CrossRef]
6. Madni, A.M.; Madni, C.C.; Lucero, S.D. Leveraging Digital Twin Technology in Model-Based Systems Engineering. *Systems* **2019**, *7*, 7. [CrossRef]
7. Steindl, G.; Stagl, M.; Kasper, L.; Kastner, W.; Hofmann, R. Generic Digital Twin Architecture for Industrial Energy Systems. *Appl. Sci.* **2020**, *10*, 8903. [CrossRef]
8. Tao, F.; Qi, Q.; Wang, L.; Nee, A. Digital Twins and Cyber-Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. *Engineering* **2019**, *5*, 653–661. [CrossRef]
9. Cortés, D.; Ramírez, J.; Villagómez, L.; Batres, R.; Vasquez-Lopez, V.; Molina, A. Digital Pyramid: An approach to relate industrial automation and digital twin concepts. In Proceedings of the International Conference on Engineering, Technology and Innovation (ICE/ITMC), Cardiff, UK, 15–17 June 2020; pp. 1–7. [CrossRef]
10. Attaran, M.; Celik, B.G. Digital Twin: Benefits, use cases, challenges, and opportunities. *Decis. Anal. J.* **2023**, *6*, 100165. [CrossRef]
11. Wang, Z. Digital Twin Technology. In *Industry 4.0-Impact on Intelligent Logistics and Manufacturing*; IntechOpen: London, UK, 2020; Chapter 7. [CrossRef]
12. Lee, J.; Azamfar, M.; Singh, J.; Siahpour, S.; Siahpour, S. Integration of digital twin and deep learning in cyber-physical systems: Towards smart manufacturing. *Int. J. Precis. Eng. Manuf. Smart Technol.* **2020**, *2*, 34–36. [CrossRef]
13. Hu, W.; Hu, W.; Zhang, T.; Deng, X.; Liu, Z.; Liu, Z.; Tan, J.; Tan, J. Digital twin: A state-of-the-art review of its enabling technologies, applications and challenges. *J. Intell. Manuf. Spec. Equip.* **2021**, *2*, 1–34. [CrossRef]
14. Gamble, J. Digital Twins: Bridging the Physical and Virtual Worlds. Available online: https://www.ericsson.com/en/about-us/new-world-of-possibilities/imagine-possible-perspectives/digital-twins/ (accessed on 8 February 2023).
15. Saracco, R. Digital Twins' Future. Available online: https://cmte.ieee.org/futuredirections/2021/01/26/digital-twins-future/ (accessed on 2 April 2023).
16. Saravanan, S.K.; Muthusenthil, B.; Gurusubramani, S. A Review of Digital Twin Leveraging Technology, Concepts, Tools and Industrial Applications. In Proceedings of the 1st International Conference on Computational Science and Technology (ICCST), Chennai, India, 9–10 November 2022; pp. 742–748. [CrossRef]
17. Constantinescu, C.; Popescu, D.; Todorovic, O.; Virlan, O.; Tinca, V. Methodology of Realising the Digital Twins of Exoskeleton-Centred Workplaces. 2018. Available online: https://atna-mam.utcluj.ro/index.php/Acta/article/view/1026/955 (accessed on 10 February 2023)
18. Mincă, E.; Filipescu, A.; Cernega, D.; Șolea, R.; Filipescu, A.; Ionescu, D.; Simion, G. Digital Twin for a Multifunctional Technology of Flexible Assembly on a Mechatronics Line with Integrated Robotic Systems and Mobile Visual Sensor—Challenges towards Industry 5.0. *Sensors* **2022**, *22*, 8153. [CrossRef] [PubMed]
19. Moiceanu, G.; Paraschiv, G. Digital Twin and Smart Manufacturing in Industries: A Bibliometric Analysis with a Focus on Industry 4.0. *Sensors* **2022**, *22*, 1388. [CrossRef] [PubMed]
20. Dosoftei, C.C.; Lupu, A.; Mastacan, L. Real-Time Communication between Automation Studio and PLC Based on OPC Technology for Control 3-DoF Robot. In Proceedings of the 24th International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019; pp. 1493–1496. [CrossRef]
21. Innovation through Digital Transformation. Available online: https://digital-innovation.zone/en/home/ (accessed on 12 February 2023).
22. Romania-Digital Innovation Hub. Available online: https://projects2014-2020.interregeurope.eu/ruralsmes/news/news-article/12598/romania-digital-innovation-hub/ (accessed on 4 February 2023).
23. Ugarte Querejeta, M.; Illarramendi Rezabal, M.; Unamuno, G.; Bellanco, J.L.; Ugalde, E.; Valor Valor, A. Implementation of a holistic digital twin solution for design prototyping and virtual commissioning. *IET Collab. Intell. Manuf.* **2022**, *4*, 326–335. [CrossRef]
24. Süß, S.; Strahilov, A.; Diedrich, C. Behaviour simulation for virtual commissioning using co-simulation. In Proceedings of the 20th Conference on Emerging Technologies & Factory Automation (ETFA), Luxembourg, 8–11 September 2015; pp. 1–8. [CrossRef]
25. Ugarte, M.; Etxeberria, L.; Unamuno, G.; Bellanco, J.L.; Ugalde, E. Implementation of Digital Twin-based Virtual Commissioning in Machine Tool Manufacturing. *Procedia Comput. Sci.* **2022**, *200*, 527–536. [CrossRef]
26. Kucak, D.; Juricic, V.; Juričić, V.; Dambic, G. Machine Learning in Education—A Survey of Current Research Trends. In Proceedings of the 29th Daaam International Symposium on Intelligent Manufacturing and Automation, Zadar, Croatia, 24–27 October 2018; pp. 406–410. [CrossRef]
27. Beilby, A. Top 4 Applications for Digital Twins. Available online: https://virtualcommissioning.com/top-4-applications-for-digital-twins/ (accessed on 15 April 2023).
28. Ryalat, M.; ElMoaqet, H.; AlFaouri, M. Design of a Smart Factory Based on Cyber-Physical Systems and Internet of Things towards Industry 4.0. *Appl. Sci.* **2023**, *13*, 2156. [CrossRef]

29. Apv, R. Virtual commissioning and digital twin of yaskawa motoman robotic cells based on the industry 4.0 context. *Int. Robot. Autom. J.* **2021**, *7*, 35–38. [CrossRef]
30. Dumitrascu, A.; Nae, L.; Predincea, N. Virtual Commissioning as a Final Step in Digital Validation of the Robotic Manufacturing Systems. *Proc. Manuf. Syst.* **2014**, *9*, 215–220.
31. Liu, Y.K.; Ong, S.K.; Nee, A.Y.C. State-of-the-art survey on digital twin implementations. *Adv. Manuf.* **2022**, *10*, 1–23. [CrossRef]
32. Zamfirescu, I.; Pascal, C. Modelling and simulation of an omnidirectional mobile platform with robotic arm in CoppeliaSim. In Proceedings of the 24th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 8–10 October 2020; pp. 667–672. [CrossRef]
33. Mixed Reality Documentation. Available online: https://learn.microsoft.com/en-us/windows/mixed-reality/ (accessed on 10 February 2023).
34. Künz, A.; Rosmann, S.; Loria, E.; Pirker, J. The Potential of Augmented Reality for Digital Twins: A Literature Review. In Proceedings of the Conference on Virtual Reality and 3D User Interfaces (VR), Christchurch, New Zealand, 12–16 March 2022; pp. 389–398. [CrossRef]
35. Kim, H.I.; Kim, T.; Song, E.; Oh, S.Y.; Kim, D.; Woo, W. Multi-scale Mixed Reality Collaboration for Digital Twin. In Proceedings of the International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), Bari, Italy, 4–8 October 2021; pp. 435–436. [CrossRef]
36. Orsolits, H.; Rauh, S.F.; Estrada, J.G. Using mixed reality based digital twins for robotics education. In Proceedings of the International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), Singapore, 17–21 October 2022; pp. 56–59. [CrossRef]
37. Valles, J.; Zhang, T.; McIntosh, P.; Pacilli, M.; Nataraja, R. Assessment of Core Surgical Skills Using a Mixed Reality Headset—The MoTOR Study. *J. Med. Syst.* **2022**, *46*, 102. [CrossRef] [PubMed]
38. Yang, C.; Tu, X.; Autiosalo, J.; Ala-Laurinaho, R.; Mattila, J.; Salminen, P.; Tammi, K. Extended Reality Application Framework for a Digital-Twin-Based Smart Crane. *Appl. Sci.* **2022**, *12*, 6030. [CrossRef]
39. KEPServerEX: One Data Source for your Industrial Automations. Available online: https://www.ptc.com/en/products/kepware (accessed on 19 February 2023).
40. The Ease-of-Use of Unity's Networking with the Performance & Reliability of Photon Realtime. Available online: https://www.photonengine.com/pun/ (accessed on 8 March 2023).