DOI: 10.2507/34th.daaam.proceedings.xxx

ASSESSING UNITY'S APPLICABILITY FOR CREATING DIGITAL TWINS BY CREATING A MULTI-ROBOT DIGITAL TWIN

Elisa Bauer, Ali Aburaia, Horst Orsolits & Mohamed Aburaia

This Publication has to be referred as: Bauer, E[lisa]; Aburaia, A[li]; Orsolits, H[orst]; Aburaia, M[ohamed] (2023). Assessing Unity's applicability for creating digital twins by creating a multi-robot digital twin, Proceedings of the 34th DAAAM International Symposium, pp.xxxx-xxxx, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-xx-x, ISSN 1726-9679, Vienna, Austria
DOI: 10.2507/34th.daaam.proceedings.xxx

Abstract

A Digital Twin (DT) replicates physical entities virtually and accurately mirrors their behaviour and condition. The number of Scopus database publications mentioning "digital twin" in their abstract increased from 51 to 2381 between 2016 and 2020. With the growing demand for DTs, there is a need for software to create and run them. While Unity has often been used for DT creation in research, its suitability for this purpose remains unexplored. This thesis aims to investigate Unity's suitability by creating a DT of a factory. The DT includes Kuka and UR5 robots, as well as linear and rotary actuators. Communication between the DT and physical system is established using ROS. Tests were conducted to measure the ROS connection latency and the accuracy of robot movements. Latency was tested under different scenarios: regular operation, reduced GPU power, and additional CPU load. Although latency was generally low, additional CPU load had a significant influence on it. The accuracy and repeatability of robot movements were assessed using the ISO 9283:1998 standard as a guide. Compared to their physical counterparts, the robot's movements exhibited significantly worse accuracy and repeatability by orders of magnitude. The robots controlled using the ArticulationBody script showed large positioning errors.

Keywords: Digital Twin; ROS; Unity

1. Introduction

Industry 4.0 has become a transformative global influence, revolutionizing industries. Deloitte's 2019 survey revealed that 74% of executives anticipated its substantial impact, 88% of leaders in other industries and 92% of IT executives agreed on its influential role [1]. Using Cyber-Physical Systems and Artificial Intelligence, Industry 4.0's implementation enhances flexibility, resource efficiency, and product adaptation to individual consumer needs [2]. A key factor to achieving these benefits is the concept of Digital Twins (DT). A DT virtually replicates a physical product, machine or system to mimic their condition and behaviour as closely as possible through an array of simulations [3]. The goal of a DT is to mirror the behaviour and condition of a physical counterpart. To achieve this one DT can consist of multiple simulations [3]. The Unity engine includes multiple relevant simulation features, like the Nvidia PhysX physics engine and allowing customization via C# scripts. Unity Technologies further supports developers in the robotics field with additional packages geared towards robotics, including robot control and Robot Operating System (ROS) integration [4]. These features make Unity an attractive option to researchers. The result is that the Unity game

engine is already commonly employed in academic DT simulations. Many of these focus on visualization and did not evaluate the simulation accuracy of Unity in their studies [5–7].
The aim of this work is to analyse the accuracy of Unity's simulations regarding robot movement and the ROS connection.

2. Materials and methods

The physical and digital systems consist of three Kuka KR 6 R900 sixx robots, two UR5 robots mounted on linear rails, a Festo tripod EXPT-45, and an eXtended Transport System by Beckhoff in an oval configuration, featuring six movers. There are also various small linear and rotational actuators, see Figure 1. The DT was developed using Unity version 2021.3.8f1. All throughout development and testing a PC equipped with an 8-Core Ryzen 7 5800X CPU running at 3.8 GHz, an Nvidia Geforce RTX 3070 GPU, and 32GB of RAM was used.

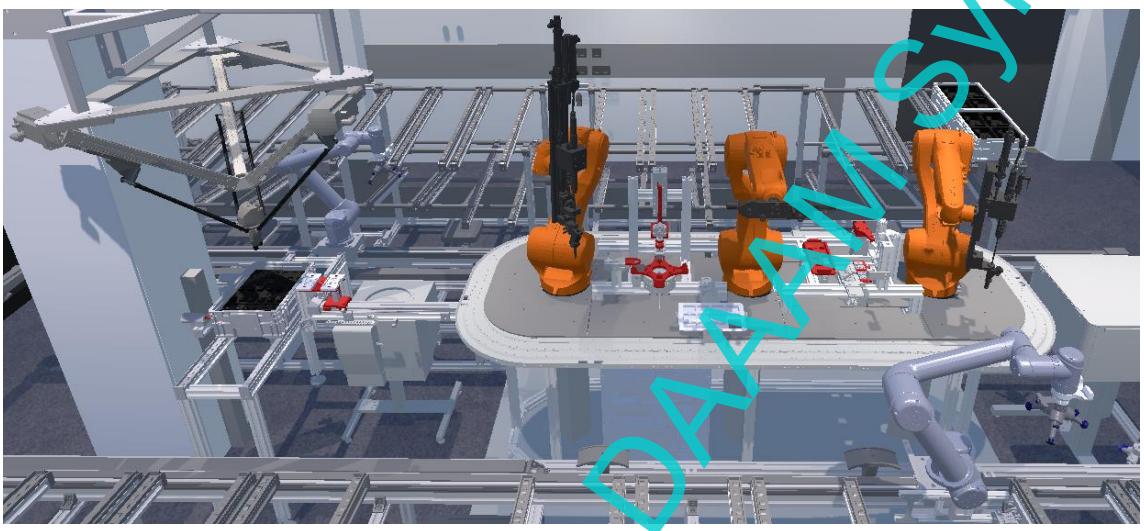

Figure 1 View of the DT in Unity including the Kuka, UR5 and Delta robots, the eXtended Transport System and, shown in red, the various smaller moving components.

In order to measure the movement accuracy, it is divided into two components, pose accuracy and timing accuracy. The timing is given by the data of the physical twin. The connection between the physical and DT is realised using ROS. For the timing the latency of the ROS connection is examined. The pose accuracy and repeatability of the robots is measured using the standard ISO 9283:1998 as a guide [8].

3. Practical realisation

To measure the variables mentioned above a DT was created. The models of the factory were available as SolidWorks files. Two methods were used to convert the models from CAD files to formats which can be imported into Unity. The first method, visualised in Figure 2, was used for the robots and actuators, except for the Delta robot. They were exported from SolidWorks as Unified Robotics Description Format (URDF) files using the URDF Exporter Addon. The URDF files include the links, joints and geometry, which were set in the Addons interface in SolidWorks. The URDF files were then imported into Unity using the URDF Importer package. The importer creates the robots as nested GameObjects with the ArticulationBody script attached to each link. The joint settings are automatically taken from the URDF file.

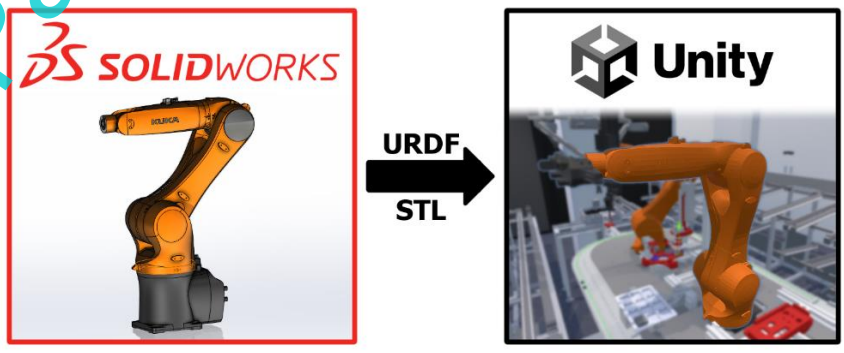

Figure 2 Visualisation of the software and file types to import the robots into Unity.

In the second method, shown in Figure 3, the models were exported from SolidWorks as Virtual Reality Modelling Language (VRML) files. The files were then imported into Blender and optimised, by:

- Combining the meshes into meaningful objects.
- Setting the origins.
- Cleaning up the geometry.
- Deleting unnecessary geometry.
- Reducing the vertex count.
- Removing excess materials.
- Assigning new materials.

Once the geometry was optimised the models were exported as FBX files, in this format the models were imported into Unity.

Figure 3 Visualisation of the software and file types to import other geometry into Unity.

The connection to the physical twin was established using the ROS TCP Endpoint node in combination with the ROS TCP Connector package in Unity. The physical twin publishes the data which is used to control the movement in the DT. The robots in the DT move according to the data immediately after receiving it. For the robots and actuators, the joint angles are published, which is why they are controlled using the ArticulationBody package. The package lets one set the angles of rotatory joints and distances for linear joints. The movers' data is their distance from their home position along the rail. Their position in the world coordinate system is calculated using a bézier curve along the path. The Delta robot's data is the position of its end effector in the robot's coordinate system. The position is converted into world coordinates using build in Unity functions. The Delta robot's end effector and the movers are GameObjects and can be moved to the correct position using the goal position in the world coordinate system.

The DT can be set to automatically connect to a set ROS topic on start or not. If the ROS connection is not automatically established, it can be started using the UI. The UI also has options to move the robots manually, display the robots' status and change the camera angle.

4. Experiments

Two experiments were conducted, one to find the latency of the ROS connection and one to find the accuracy and repeatability of the robots. The way the two experiments were designed is presented in the following chapters.

4.1. Ros Connection

To measure the latency the published data was recorded in a rosbag file. The file contained 157509 messages and lasted 1224 seconds. The file was played back and the times, at which the messages were received in Unity, were recorded. The recorded data was then compared to the time at which the data was published. This test was conducted for six scenarios:

- Regular operation
- Without simulating the robots' movements
- With the CPU power limited to 80%
- With the GPU power limited to 20%
- Adding an extra load of 30% on the CPU
- Adding an extra load of 60% on the CPU

The first two scenarios are designed to give a benchmark and to test the effect of computational load in Unity on the connection. The other two sets are designed to gauge the influence of the GPU and CPU respectively. Each scenario was recorded two times with no other programmes running on the PC. The scenarios were designed to test whether running the simulation in Unity or the hardware has an effect on the latency.

4.2. Accuracy and Repeatability

The second experiment was testing the positioning and orientational accuracy and repeatability of the robots and movers. The standard ISO 9283:1998 was used as the base to design the methods of measuring and calculating the robots accuracy and repeatability [8]. The methods of the standard could not be followed completely as they are designed for physical systems. The standard defines five measurement points using an ISO Cube. The points are located on the diagonals of the measurement plane. For 6-axis robots the plane lies diagonally in the cube connecting one lower side with the opposite upper side. The cubes are placed in front of the robots within their workspace, see Figure 4.

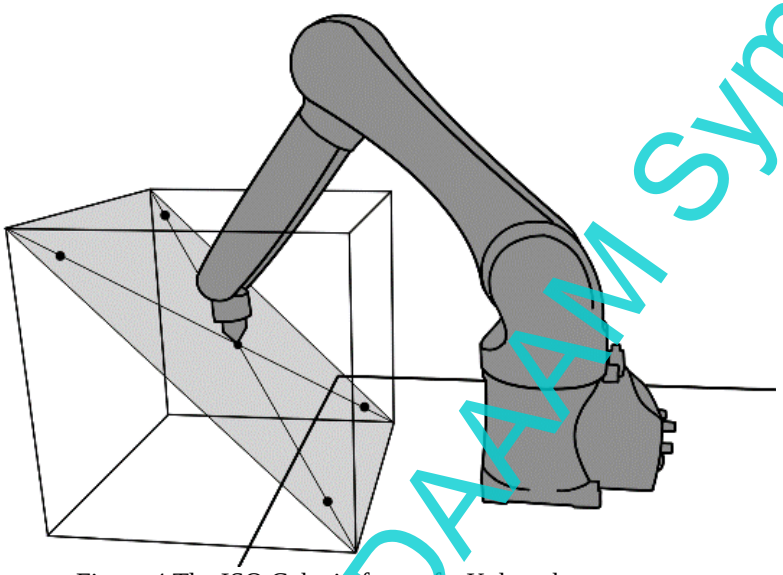

Figure 4 The ISO Cube in front of a Kuka robot

In one measurement cycle the robot moves to each point and stops while the measurement of the end effectors position and orientation is recorded. The BioIK package was used to move the robots by end effector position rather than joint angles. The resulting joint angles for each point were saved and used to move the robot during the measurement cycles. As mandated by the standard 30 measurement cycles as well as a 0 cycle, which is not recorded, were performed. The positional and orientational accuracy and repeatability for each point were calculated as defined in the standard.

5. Results

For the ROS connection the first tests, running with and without the robots, was designed to evaluate if higher workload on the Unity application has a significant impact on the connection speed. The results are presented below in Table 1.

	With robots	With robots	Without robots	Without robots
Mean Error [ms]	16.07	16.65	27.82	19.94
Max. Error [ms]	89.98	55.36	50.49	54.05
Standard Deviation [ms]	11.08	9.49	10.44	10.02

Table 1 The mean and maximum error and the Standard Deviation in milliseconds for the testcases with no hardware limitations.

The average latency remained below 30ms for the tests without hardware constraints, for both scenarios: with and without robots. Although the maximum error was slightly higher in tests involving robots, the standard deviation remained below 11ms in all cases. Considering the application's frame rate is approximately 60 frames per second, equivalent to 16.67ms per frame, the observed 3-7ms difference was considered insignificant. Furthermore, since the tests with robots showed slightly lower errors, it can be concluded that the simulation does not adversely affect latency.

These two test sets were established as benchmarks for the scenarios involving hardware limitations. Subsequent tests were performed with the robot simulation. Specifically, the GPU was restricted to 80% and 60% power consumption, as indicated in Table 2.

	80% GPU	80% GPU	60% GPU	60% GPU
Mean Error [ms]	16.89	30.19	9.53	33.73
Max. Error [ms]	53.87	53.99	81.41	57.24
Standard Deviation [ms]	9.53	10.58	8.19	10.07

Table 2 The mean and maximum error and the Standard Deviation in milliseconds for the testcases with GPU limitations.

The outcomes from both sets of tests involving the GPU remained relatively consistent with those conducted without any constraints. The maximum error and standard deviation in both sets were comparable to the benchmark tests. Given that the increase in the mean was only around 5ms compared to the tests without constraints, and the maximum error was either lower or similar to the benchmark, it can be inferred that GPU performance has very little impact on latency.

Subsequently, the CPU was subjected to testing under additional loads of 30% and 60%, and the recorded results are presented in Table 3.

	30% CPU	**30% CPU**	**60% CPU**	**60% CPU**
Mean Error [ms]	27.19	12.51	22245.49	20526.7
Max. Error [ms]	53.30	186.38	39624.55	36095.82
Standard Deviation [ms]	10.49	10.40	11054.03	10453.98

Table 3 The mean and maximum error and the Standard Deviation in milliseconds for the testcases CPU limitations.

In the first set of tests with additional 30% CPU load, the results remained consistent with previous tests for the mean and maximum error. Although plotting the difference for each messages shows a shift towards higher differences. However, when the CPU load was increased to 60%, a significant difference was observed. Latency reached a maximum of 396.24 seconds, and the standard deviation for both runs exceeded 10,000ms, in contrast to the consistent 10ms observed in all other runs.

Figure **5** illustrates how the difference increased with each message, with errors accumulating over the test duration. This test clearly demonstrates the substantial impact of CPU load on latency. The slight difference observed in the plotted error with smaller additional loads implies that even minor increases in CPU load or limited CPU power can have a noticeable effect.

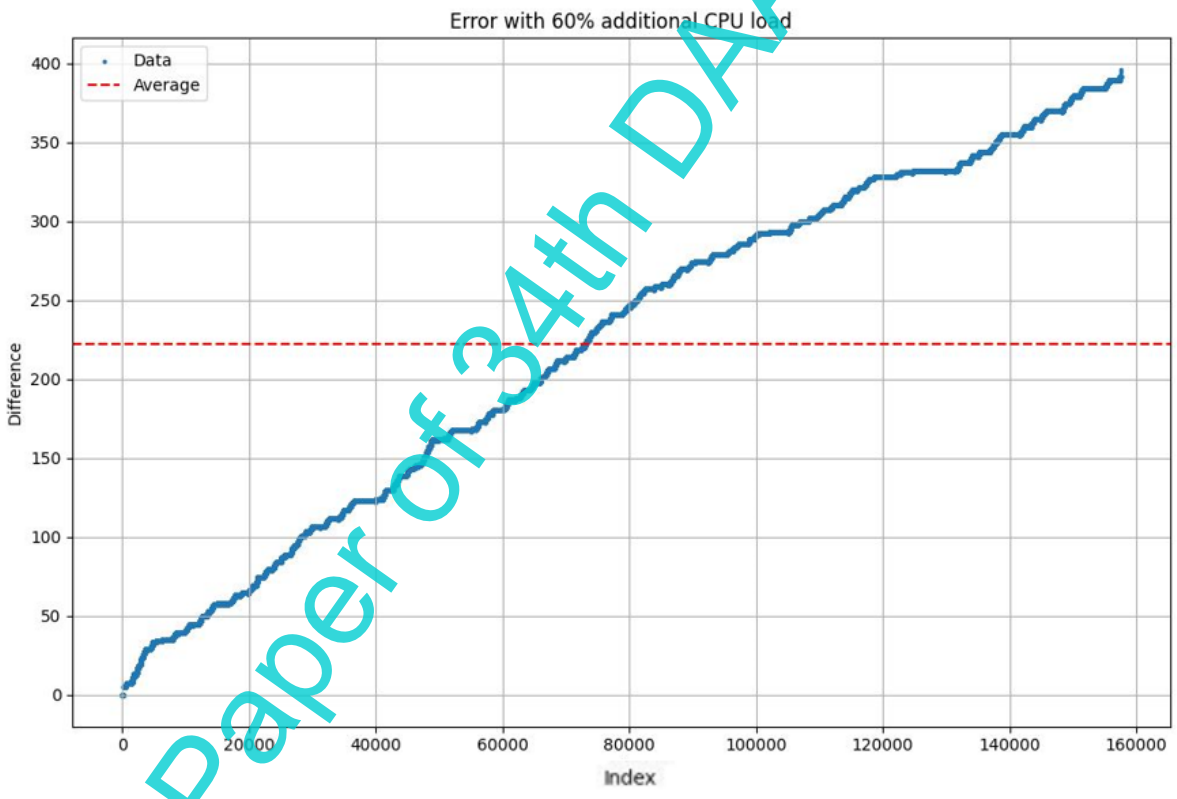

Figure 5 Test case with an additional load of 60% on the CPU. The difference, in seconds, is plotted over the duration of the recording, measured in number of messages. The average is shown by a red dotted line.

The results of the accuracy and repeatability measurements shows that the Kuka and UR5 robots performed worse than their physical counterparts, in all categories except for orientational repeatability. The average positional accuracy (APp) of the Kuka robots lied between 14.4mm and 114.2mm. The average orientational accuracy (APa, APb and APc) were -0.28328°, -0.28885° and -0.01599° respectively. In comparison to the pose accuracy of between 0.5mm and 0.7mm, which was measured on a physical KUKA model AGILUS KR 6 R900 sixx without prior calibration [9]. The positional repeatability (RP_l) of the robots ranged from 0.5mm to 5.6mm, which is up to 3 orders of magnitude greater than ± 0.03 mm, which is the pose repeatability given by the manufacturer [10]. The findings suggest that the accuracy is influenced by the end effector. This is evident from the minor 0.4mm difference in accuracy between the two robots with identical end effectors, with greater differences between robots with different end effectors. This indicates that the accuracy is not solely determined by the number of links or types of joints, but also by the geometry of the links. The

UR5 robots had a similar result with an average APp across all points for both robots of 82.5mm. The repeatability was however a lot better than of the Kuka robots with an average of 0.021mm. The manufacturer of the UR5 robots specifies a repeatability of ± 0.1 mm [11]. However, the orientational repeatability for all robots was high, the results are in the range of 3.8e-14° to 1.9e-12°.

The high APp of the robots can be explained by the method of evaluation. The commanded TCP position was defined by the ISO Cube within the world coordinate system. The robot's movement given by the TCP position was executed using the BioIK package to record the joint positions. During this process, when converting angles from the BioIK script to ArticulationBodies, rounding errors might impact the resulting TCP position. These errors only affect the measurement process but do not impact the DT during regular operation. Because during regular operation the angles are provided by the physical twin's data. However, such errors cannot account for the low positional repeatability observed in the Kuka robots. The ArticulationBody script moves the robots by applying forces to the joints, which should let the joint reach the set position. Various factors, such as collisions, inertia, and controller settings, can influence the accuracy of these movements. To achieve the most precise results, values need to be adjusted to each situation, as they cannot all be set to manufacturer specifications due to the absence of a standardized unit.

The Movers and the Delta robot are not controlled using this method. The Delta robot's end effector is positioned, using the Transform script, according to the data provided by ROS, while the robot's links move using an inverse kinematic script. Since the end effector is accurately set to its desired location, there is no disparity between the commanded and executed poses. The measurement accurately reflects this precision.
APp and RPl for these robots are 0.0mm. Because the end effector doesn't rotate, there is no orientational accuracy or repeatability to consider. This is in contrast to the physical robot, which boasts an accuracy of 0.5mm and a repeatability of ± 0.1mm [12].

6. Conclusion

The connection via ROS with the mentioned packages provides a fast connection for small message sizes. The Connection is reliable as all messages were received by the Unity subscriber. The computational load of the Unity application and the GPU have no significant effect on the connection speed. The latency is however highly affected by the CPUs performance. Even with smaller additional loads of 30% plotting the error for each point showed a shift towards higher errors. At 60% additional load the error becomes greater for each message.

The positional accuracy and repeatability for the robots controlled using the ArticulationBody script is lower than, what the manufacturer of the robots claim, their physical counterparts can achieve, by orders of magnitude.

Although there are many settings and variables which can be changed in the ArticulationBody script. A more in-depth study of the effects of these variables on the accuracy is necessary.

7. References

[1] N. Farkas-Mills, *The Fourth Industrial Revolution: At the intersection of readiness and responsibility.* [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/uk/Documents/energy-resources/deloitte-uk-fourth-industrial-revolution.pdf (accessed: Jul. 27 2023).

[2] H. Kagermann, W. Wahlster, and J. Helbig, *Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final report of the Industrie 4.0 Working Group*: Forschungsunion, acatech, 2013. Accessed: Jul. 27 2023.

[3] S. Haag and R. Anderl, "Digital twin – Proof of concept," *Manufacturing Letters*, vol. 15, pp. 64–66, 2018, doi: 10.1016/j.mfglet.2018.02.006.

[4] Unity Technologies, *Unity-Robotics-Hub.* [Online]. Available: https://github.com/Unity-Technologies/Unity-Robotics-Hub (accessed: Jul. 10 2023).

[5] F. Tao, B. Xiao, Q. Qi, J. Cheng, and P. Ji, "Digital twin modeling," *Journal of Manufacturing Systems*, vol. 64, pp. 372–389, 2022, doi: 10.1016/j.jmsy.2022.06.015.

[6] S. Xu, S. Moore, and A. Cosgun, "Shared-Control Robotic Manipulation in Virtual Reality," May. 2022. [Online]. Available: http://arxiv.org/pdf/2205.10564v1

[7] G. Garg, V. Kuts, and G. Anbarjafari, "Digital Twin for FANUC Robots: Industrial Robot Programming and Simulation Using Virtual Reality," *Sustainability*, vol. 13, no. 18, p. 10336, 2021, doi: 10.3390/su131810336.

[8] *Manipulating industrial robots - Manipulating industrial robots — Performance criteria and related test methods*, 9283:1998. International Organization for Standardization, 1998.

[9] D. Wilde and S. Guillaume, "Calibration of a Six-Axis Robot for GNSS Antenna Phase Center Estimation," *J. Surv. Eng.*, vol. 145, no. 4, 2019, doi: 10.1061/(ASCE)SU.1943-5428.0000291.

[10] KUKA Deutschland GmbH, *KR 6 R900 EX.* [Online]. Available: https://www.kuka.com/-/media/kuka-downloads/imported/8350ff3ca11642998dbdc81dcc2ed44c/0000293617_en.pdf?rev=aaaf08e92a6f44dbb1ad5c48467d59ae&hash=57ED900A4C91C2BE6ADD02875DDBC9DF (accessed: Aug. 1 2023).

[11] Universal Robots A/S, *UR5 Technical specifications: 6-axis robot arm with a working radius of 850 mm / 33.5 in.* [Online]. Available: https://www.universal-robots.com/media/50588/ur5_en.pdf (accessed: Aug. 1 2023).

[12] Festo Corporation, *Electrical Tripod EXPT.* [Online]. Available: https://www.festo.com/net/SupportPortal/Files/13547/EXPT-PSI-US.pdf (accessed: Aug. 1 2023).