ROS middle-layer integration to Unity3D as an interface option for propulsion drive simulations of autonomous vehicles.

Vladimir Kuts1, Anton Rassõlkin2, Andriy Partyshev1, Sergei Jegorov1, Viktor Rjabtšikov2

1Tallinn University of Technology, School of Engineering, Department of Mechanical and Industrial Engineering, Estonia, Harju, Tallinn, Ehitajate str. 5, 19086
2Tallinn University of Technology, School of Engineering, Department of Electrical Power Engineering and Mechatronics, Estonia, Harju, Tallinn, Ehitajate str. 5, 19086

*vladimir.kuts@taltech.ee

Abstract. As autonomous vehicle development continues at growing speeds, so does the need for optimization, diagnosis, and testing of various autonomous systems elements, under different conditions. However, since such processes should be carried out in parallel, it may result in bottlenecks in development and increased complexity. The trend for Digital Twins brings a promising option for the diagnosis and testing to be carried out separately from the physical devices, incl. Autonomous vehicles, in the virtual world. The idea of intercommunication between virtual and physical twins provides possibilities to estimate risks, drawbacks, physical damages to the vehicle's drive systems, and the physical one's critical conditions. Although the problem of providing communications between these systems arises, at the speed that will be adequate to represent the physical vehicle in the virtual world correctly, it is still a trending topic. The paper aims to demonstrate a way to solve this problem - by using ROS as a middleware interface between two twining systems on the autonomous vehicle propulsion drive example. Data gathered from the physical and virtual world can be exchanged in the middle to allow continuous training and optimization of the propulsion drive model, leading to more efficient path planning and energy-efficient drive of the autonomous vehicle itself.

1. Introduction
Simulation is an approximate or 1-to-1 imitation of a real process, often taking part in the virtual environment, troubleshooting, researching, testing, training, monitoring, controlling, or educating. In the past decade, simulations have been vital in production and development as they are capable of preventing many problems related to planning and reducing bottlenecks at early stages, also during the real-time maintenance of the process [1]–[5] and, especially with increasing technology complexity and rise in using fully autonomous systems, enforcing and changing work-safety features. One side of the simulation aspect - the concept of Digital Twin (DT) [6], [7] is being exploited in the related research to develop a precise dual-way synchronized simulation interface for the propulsion drives [8], [9] to be ready to be integrated into the electrical vehicles [10].

Physics simulations are very common and critical nowadays. They are used enormously in such applications as MATLAB Simulink, Simscape, CAD design, SolidWorks, etc., and gaming physical
processes simulations. They should be considered in the mechatronic systems' planning stage [11]. Of course, they all have approximation and simplifications, while not all possible physical laws can be yet simulated simultaneously; however, such simulations provide considerable benefit in research and testing.

In a previous study done by authors of the related paper, which was on the on electrical motors simulation under development of DT for propulsion drive of an autonomous electric vehicle [12], Unity3D was used for simulations of DT that was exchanging messages with Robot Operation System (ROS) node through a ROS bridge [13]. However, ROS is not being used only for robots but also for various drones, self-driving vehicles, and autonomous systems. ROS enables inter-process communication; it is believed to be a quality method of interconnecting a digital twin propulsion drive system with its real counterpart. ROS was used for performance calculation using an empirical performance model for induction motor (IM). As a visualization tool in the related research is being used Unity3D which is connected with ROS directly [14]. Even though Unity3D simulated most of the motor's physical behavior (torque and rotation), the response and received numerical values, unfortunately, do not suit the DT development in the long run. The reason for this is the complexity of the overall system of physics of IM. Moreover, to make the system transferable and usable with other models (not the ones present in Unity3D but also in Gazebo or elsewhere) the physics handling has to be close to standalone.

The research's main aim is to develop a framework and a toolkit, including a middle-layer ROS interface connected with the physical propulsion drive workbench and its DT, which can be visualized in various simulation engines. The related paper aims to develop a methodology to connect the interface with Unity3D for the visualization, considering data exchange and feedback.

2. **Methodology**

2.1. *Working principle of a test bench on a digital twin*

For the current case study, the DT operates on the simulated data generated based on real data measured and gathered from the 7.5 kW IM (ABB 3GAA132214-ADE). The data was gathered using the data acquisition system (DAS) Dewetron Dewe 2 and saved into files with a different extension (*.mat, *.xlsx, *.csv, *.txt). The measured data can be anything regarding the motor's operation, namely input currents and voltages, consumed and shaft power, torque and angular velocity on data acquisition, and other side data calculated from them. According to DAS tuning (16Hz - 100kHz), the parameters can be measured with different frequencies, and received data is relative to time. This feature enables to recreate the motor's behavior precisely as it happened in the real case scenario with the help of ROS Server. An example of such can be seen in Figure 1, where the input current from frequency converter to IM was recorded and now can be simulated in ROS (graph from ROS package rqt plot we were not included to the related paper because it could not handle plotting messages at such high frequency).

In the proposed DT system, ROS Server acts as a data server and physics simulator. The idea behind it is the following: the server is a standalone subsystem of a TB DT that is responsible for processing real, measured data of the motor, calculating other motor parameters based on the processed data, and streaming to the ROS topics available for models.

Figure 2 features the architecture of the DT setup for TB. The real data is fetched to appropriate ROS Nodes (components of ROS server that are performing calculations, real data processing, and streaming of data) present in the server, processed and translated into ROS messages, and finally, sent to the DT model over ROS Bridge. The real data can be based on the empirical model/map of the motor (or its part) or the actual raw data.

Upon receiving ROS messages, the model can perform the necessary actions to simulate the mechanical/electrical/thermal behavior. Models can be present in any simulation environment. They are subscribed to ROS Server's topics over API or ROS Bridge and configured to perform the necessary operations based on the subscribed ROS topic (for example, rotation based on received angular speed). Furthermore, the module can feature simulated 'measurement' devices/sensors that can send back the
data over the ROS bridge. In this case, the ROS Nodes can process and calculate other required values, as it would happen in the real TB.

![Figure 1](image1.png)

**Figure 1.** Input current measurements sampled at 5kHz frequency

The current DT of TB consists of the Unity 3D model and ROS Server. ROS Server streams simulated values regarding input power (3-phase current and voltages), efficiency calculated based on measured torque and angular velocity. The torque is calculated by the physics engine of the Unity3D, whereas other values are based on the real ones. This creates a problem of incorrect data calculation because Unity does not focus on calculating correct values on physics laws, as it is more for games, allowing developers to adjust the physics laws to the game setup. This is why the shift from the physics engine of the model environment to ROS was introduced. ROS server would serve physical parameters based on the real TB data and independent of the modeling environment. Additionally, ROS can record rosbags – files with recorded values from topics/servers that can be played back to repeat the behavior. Such a feature would allow us additional analytical features from the DT side.

![Figure 2](image2.png)

**Figure 2.** The generic architecture of TB DT
2.1.1. **ROS Interfacing**

To allow easy interfacing of ROS with other systems, a ROS Bridge node has to be used. It converts ROS communications into a JSON file format and sends them outside of the ROS ecosystem. JSON is used because of its universal format with existing libraries that support its serialization and deserialization in almost every modern programming language. Taking it one step further, ROS Bridge can be used to port specific ROS topics to and out of Message Queuing Telemetry Transport (MQTT) protocol to upscale the system and allow it to run on multiple machines around the world. This so-called MQTT Bridge sends data to the remote server by taking the serialized message on a specified ROS topic and publishes it into a specified MQTT topic. MQTT Bridge is also capable of the inverse - it receives a JSON-serialized message and attempts to deserialize it into a specified ROS topic in a specific message type. Together these systems make interfacing of ROS with any visualization solution much simpler to develop. To further simplify the deserialization process, classes that match ROS message types were created in C# for Unity3D implementation of the ROS interface. This approach can be considered the most efficient because, in this case, a ROS message delivered in the serialized form via MQTT can be directly deserialized into an object of a matching type. This approach can be implemented in similar ways on the majority of existing programming languages, making it the most straightforward and most versatile option.

Visualization is being done in Unity3D (See Fig. 3) engine connected to the physics simulator via ROS Interface, where it is a 1 to 1 scale propulsion drive model with the transmission, wheel parts, and non-visible gears. Model is being assembled as the physical one, and each part is being controlled by a related script, where data is being fed from the middle layer.

![Figure 3. Visualization of propulsion drive test bench done in Unity3D](image)

3. **Discussion**

The primary outcome of the related part of the more extensive research in developing the fully synchronized DT of the propulsion drive is that the ROS interface was developed. It is possible to feed it with gathered from the physical data and give to the visual simulated, which in related use-case is being Unity3D. The given data simulation runs and gives logged feedback about physical interactions back to the ROS middle layer, where the model is being improved and sent back to the visual side, improving it after each data movement loop. However, some limitations were met during the development of methodology, and more developments go to reach the final aim of the stated research aim (See Table 1).
Table 1. Limitations and further steps

| Limitations                                                                 | Future steps                                                                 |
|----------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| The model was tested with only one type of visual simulation tool. Possible additional integrations should be done in the middle layer to be suitable for additional software tool packages. | To establish correct torque calculations based on the real values collected from the physical TB. |
|                                                                            | To implement a two-way connection between physical TB and its DT.              |
| If DT and TB work simultaneously over the internet, the frequency of data acquisition may be too high to send on time, the possibility of lags | The injection process flow of new components of TB into the DT.                 |
|                                                                            | To create unpredicted behaviors in the system, trigger points, and try to make the system respond to the unpredicted change making it more adaptive to changes |

4. Conclusions

The ROS interface connected with the Digital Twin of the propulsion drive workbench visualized in Unity3D was introduced during the related work. Raw and simulated data and empirical models can be post-processed and fed to the visual simulation, where additional data is being logged and given as feedback to the middleware to improve the model and physical simulation itself. The next crucial step is to feed physical simulation directly with data from the physical drive, enabling synchronization between the real and virtual worlds through the developed interface.

5. Acknowledgments

The research has been supported by the Estonian Research Council under grant PSG453 "Digital Twin for Propulsion Drive of Autonomous Electric Vehicle."

References

[1] R. AhmadiAhangar, A. Rosin, A. N. Niaki, I. Palu, and T. Korôtko, "A review on real-time simulation and analysis methods of microgrids," International Transactions on Electrical Energy Systems. 2019, doi: 10.1002/2050-7038.12106.
[2] S. Venkatesan, K. Manickavasagam, N. Tengenkai, and N. Vijayalakshmi, "Health monitoring and prognosis of electric vehicle motor using intelligent-digital twin," IET Electr. Power Appl., 2019, doi: 10.1049/iet-epa.2018.5732.
[3] G. Turner, "Soaring through virtual aviation: The role of VR in aerospace manufacturing," Manufacturing Global, 2020.
[4] L. Gevorkov, A. Rassolkin, A. Kallaste, and T. Vaimann, "Simulink based model of electric drive for throttle valve in pumping application," in 2018 19th International Scientific Conference on Electric Power Engineering, EPE 2018 - Proceedings, Jun. 2018, pp. 1–4, doi: 10.1109/EPE.2018.8395996.
[5] I. Rasheed et al., "Fast Numerical Techniques Based Analysis of Electromagnetic Problems Using MATLAB," in Proceedings - 12th International Conference on Frontiers of Information Technology, FIT 2014, Jun. 2015, pp. 115–120, doi: 10.1109/FIT.2014.30.
[6] N. Kousi, C. Gkournelos, S. Aivaliotis, C. Giannoulis, G. Michalos, and S. Makris, "Digital twin for adaptation of robots' behavior in flexible robotic assembly lines," in *Procedia Manufacturing*, 2019, doi: 10.1016/j.promfg.2018.12.020.

[7] V. Kuts, M. Sarkans, T. Otto, T. Tähemaa, and Y. Bondarenko, "Digital Twin: Concept of hybrid programming for industrial robots – Use case," in *ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE)*, 2019, vol. 2B-2019, doi: 10.1115/IMECE2019-10583.

[8] V. Kuts et al., "Synchronizing physical factory and its digital twin through an IIoT middleware: a case study," *Proc. Est. Acad. Sci.*, vol. 68, no. 4, pp. 364–370, 2019, doi: 10.3176/proc.2019.4.03.

[9] A. Rassõlkin, T. Vaimann, A. Kallaste, and V. Kuts, "Digital twin for propulsion drive of autonomous electric vehicle," in *60th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, 2019.

[10] N. Khaled, B. Pattel, and A. Siddiqui, "Digital Twin development and cloud deployment for a Hybrid Electric Vehicle," in *Digital Twin Development and Deployment on the Cloud*, 2020.

[11] R. Sell, E. Coatanéa, and F. Christophe, "Important aspects of early design in mechatronic," in *Proceedings of the International Conference of DAAAM Baltic*, 2008.

[12] A. Rassõlkin et al., "Digital Twin for Propulsion Drive of Autonomous Electric Vehicle," in *60th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, Oct. 2019, pp. 1–4, doi: 10.1109/RTUCON48111.2019.8982326.

[13] A. Rassõlkin, V. Rjabšikov, T. Vaimann, A. Kallaste, V. Kuts, and A. Partyshev, "Digital Twin of an Electrical Motor Based on Empirical Performance Model," in *Proceeding of 2020 XI International Conference on Electrical Power Drive Systems (ICEPDS)*, 2020.

[14] E. Sita, C. M. Horváth, T. Thomessen, P. Korondi, and A. G. Pipe, "ROS-Unity3D based system for monitoring of an industrial robotic process," in *SII 2017 - 2017 IEEE/SICE International Symposium on System Integration*, 2018, doi: 10.1109/SII.2017.8279361.