This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Peake, Ian; La Delfa, Joseph; Bejarano, Ronal; Blech, Jan Olaf
Simulation Components in Gazebo

Published in:
Proceedings of 22nd IEEE International Conference on Industrial Technology, ICIT 2021

DOI:
[10.1109/ICIT46573.2021.9453594](https://doi.org/10.1109/ICIT46573.2021.9453594)

Published: 18/06/2021

Document Version
Peer reviewed version

Please cite the original version:
Peake, I., La Delfa, J., Bejarano, R., & Blech, J. O. (2021). Simulation Components in Gazebo. In *Proceedings of 22nd IEEE International Conference on Industrial Technology, ICIT 2021* (pp. 1169-1175). [9453594] IEEE.
<https://doi.org/10.1109/ICIT46573.2021.9453594>

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

© 2021 IEEE. This is the author's version of an article that has been published by IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Simulation Components in Gazebo

Ian Peake*, Joseph La Delfa^{§¶}, Ronal Bejarano**, Jan Olaf Blech**

*College of Science, Engineering and Health, RMIT University, Melbourne, Australia

[§]School of Design, RMIT University, Melbourne, Australia

[¶]Exertion Games Lab, Monash University, Melbourne, Australia

**Department of Electrical Engineering and Automation, Aalto University, Espoo, Finland

Email: ian.peake(at)rmit.edu.au, cafeciaojoe(at)gmail.com, {ronal.bejarano,jan.blech}(at)aalto.fi

Abstract—We present work on structuring robotics simulation scenarios into components. Components can comprise simulation entities with a physical counterpart such as automated guided vehicles, drones, robots and machines. Components are typically further structured into sub-components, e.g., a robot arm or a gripper, and can be used to build digital twins. Different aspects for classifying components can be distinguished; here we concentrate on composability, performance and fidelity.

In this paper, we are mainly concerned with components in the robotics simulation tool Gazebo used with Robot Operating System. Several structured scenario case studies are described. We investigate how these scenarios behave in Gazebo with respect to performance and fidelity for selected cases.

I. INTRODUCTION

Simulation of industrial equipment plays an important role in predicting the behavior of new devices in existing environments before they are commissioned. Furthermore, simulation allows engineers to get a feeling for new usage scenarios and to react to future events even before these events occur. For example, critical situations can be simulated to train staff and support them to find solutions. Like in the real world, distinct components can be identified in a simulated scenario. Typically these components can be further decomposed into sub-components.

A *digital twin* (see e.g., [1], [2], [3]) is typically defined as a digital representation of a (usually) physical counterpart in the real world, whereby the digital representation is linked to the real world counterpart, so that updates on the state of the counterpart are reflected in the digital twin and vice versa. Simulated components can be connected to the real devices, thereby serving as digital twins. Digital twins may also serve as blueprints for the assembly of the corresponding physical counterparts. [4].

We focus on decomposition of robot simulations intended for use in digital twins. We use the term *simulation components* to refer to in-principle separable elements that correspond to distinct assets in digital twins, such as different types of robots. It follows that a simulation component is also an encapsulated collection of reusable artifacts for simulation, control and visualization of a cyber-physical entity, including representations of the physical entity to be simulated, optionally packaged with configuration or code that defines its behaviour. Robot software is assumed to be based on the Robot Operating System (ROS) [5], a framework for assembling robot software from existing components. Robot simulations

are assumed to be defined in Gazebo, a robotics simulation environment typically used with ROS. *Fidelity* refers to a twin's capacity to predict the behaviour of the real system, i.e. how close the simulation is to the actual physical system under simulation. *Performance* refers to user-observable properties such as prediction latency and throughput as well as resource requirements. Latency is related to fidelity, since timely predictions are more useful. Latency may be improved at the expense of cost.

In this paper, we present a classification of simulation components with respect to composability, fidelity and performance. The presentation features components with ROS interfaces and realized in Gazebo that can be used to construct digital twins with visual aspects, including potentially Mixed Reality aspects. We present examples and an analysis of fidelity and performance of the components from the examples.

The notion of simulation components is closely related to Mixed Reality (MR), which is concerned with high fidelity co-existence and interaction of virtual objects with physical objects in real-time [6], typically presented visually to a human user. MR arises naturally from the concept of digital twins, since developers must analyse combinations of physical and virtual entities and their correspondences. MR aspects will require additional development effort and become an attractive target for reuse as components. MR has been used in industrial settings (see, e.g., [7]. In addition, A MR study on human behavior based on a Gazebo simulation is presented in [8]. The MirrorLabs project features the development of MR applications primarily for educational purposes [9]. Our work continues ideas sketched in [10] where we evaluated latency issues in distributed Gazebo usage scenarios. Related to our Gazebo work, production line planning using VR has been studied in [11] and some work on Functional Mockup Interface-based simulation is presented in [12]. Early Gazebo simulation work of unmanned aerial vehicles (UAV) is presented in [13]. Other Gazebo/ROS simulation work includes [14] and [15].

The paper is structured as follows: Sect. II summarizes the general context, depicting the main components and research spaces used. Sect. III describes the simulated components concept mapped to ROS/Gazebo. Sect. IV presents several new laboratory implementations based on ROS and Gazebo, evaluated on composition, fidelity and performance. Sect. V illustrates results obtained. Finally, Sect. VI presents conclusions and future work.

II. BACKGROUND

The notion of software components and composition, and the used software stacks ROS and Gazebo as well as the two labs are described in this section.

A. Software components and composition

There is yet no universally agreed definition of a software component, although Szypersky's[16] definition is famously cited, namely: "A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties." The notion of software components has been associated with the goal of reducing life cycle costs and improving the quality of complex systems through several inter-related principles: *Reusability* and *portability*: code packaged as a component can be used in different deployment contexts with minimal configuration. *Encapsulation*: related data and functions are grouped into components. *Contractual interfaces*: components interact with each other only via well-defined, abstractly-defined protocols, and in all other respects should not interfere with each other's operation. *Independent deployment*: Components may be deployed independently of each other, including to separate computational resources such as threads, cores and hosts, to improve performance. Independent deployment typically relies on contractual interfaces. *Substitutability*: a component may in principle replace another, provided that the replacement meets all contractual requirements imposed on the original. *Composability*: *composite* components may be assembled together in standard ways to achieve new features and properties. *Independent deployment*: individual components may be added or removed from an assembly without interfering with the running of other components. *Compositionality*[17]: properties of a composite component are predictable from individual component abstractions of properties and the nature of the type of assembly performed.

B. ROS

ROS is a framework for defining control systems for robotics [5]. Among several important features, ROS provides means for programming computational elements called *nodes* to communicate abstractly via virtual message queue abstractions called *topics*, using just the topic's name. Similarly nodes may also access *services* from each other and *parameters* from a designated *master* node. ROS also provides the so-called Transformation subsystem ("TF") which enables nodes to send messages locating physical objects via their relative offsets and orientations via a special topic named `/tf`. In this paper we focus solely on ROS version 1, referring to it as "ROS". A successor version of ROS, ROS 2, not in the scope of this paper, and not fully compatible with ROS version 1, is in an advanced stage of development, capable of running demonstrators and incorporating ROS version 1 components.

ROS supports aspects of software composition as follows:

- *nodes* are run time components which are independently deployable, including on separate hosts, and communicate via topics, parameters and services, thereby supporting composition and encapsulation. It is common for ROS nodes to be substitutable for one another; common functions such as navigation and their subsystems have multiple alternative implementations.
- *packages* are reusable and independent deployable collections of sources or binaries from which ROS nodes may be run. ROS provides standardization for building binaries from source, including specification of dependencies, although this can at times be a manual process. ROS inherits external packaging standards such as from the Debian and Ubuntu projects.

C. Gazebo

Gazebo¹ is a robot simulation tool box. Gazebo is commonly used to develop tests for software developed in ROS. Gazebo provides a framework, extending ROS, which unifies aspects such as physics-based simulation and visualization using an XML-based notation to define *objects*, and a plugin architecture. Gazebo's main process *gzserver* is primarily a physics simulation engine. Gazebo also provides a web-based visualization front-end (Gzweb) and a more traditional visualization client *gzclient* based on X Windows,

D. Virtual Experiences Laboratory at RMIT

The Virtual Experiences Laboratory (VXLab) at RMIT University in Melbourne, Australia, is a multi-disciplinary laboratory supporting combinations of physical automation, virtualization and visualization [18]. VXLab includes a significant training component through capstone work-integrated learning projects for bachelor and coursework masters students near completion of their degree program [19]. VXLab hosts robots and drones as part of a broader set of facilities including a Xeon data centre cluster for development/simulation, 7m tiled display wall and virtual reality work spaces. In particular VXLab includes a mobile Cobot "Rosie" which is a Rethink Robotics Baxter two-armed Cobot, mounted on a Dataspeed Inc. mobile battery powered mecanum wheel base a Mobile Industrial Robotics MIR 100 and a Cyber-physical Simulation platform (CSRack).²

For capstone projects, VXLab provides technical support and onboarding including handover of code from prior projects. Curation is therefore also important, with deliverables typically as open source, with an Apache 2.0 style license, plus a demonstration video.

E. Aalto Factory of the Future

The Aalto Factory of the Future³ (AFoF) comprises a space shared by humans, robots and production stations. The facility

¹<http://gazebo-sim.org/>

²For background see (overview) <https://youtu.be/2Y5EyVRSVU0>, (drones) <https://youtu.be/7dW0RnZafRw>, (picking/delivery) <https://youtu.be/2XB5GXtKr6w>, and (navigation/simulation) https://youtu.be/Jtl_j8n0Mf8

³<https://www.aalto.fi/en/futurefactory>

serves as a platform for projects in the area of software, AR and VR applied to future production systems. We are particularly looking into flexible manufacturing and dynamic reconfiguration aspects.

III. APPROACH

This paper investigates the relationship between digital twin components, their simulation fidelity, and performance.

Simulation components and composition: Simulation components in our work are simulation objects that may feature a counterpart in the physical world such as a single robot, or automated guided vehicles. This is similar to the definition of a digital twin. Simulation components may be expected to reflect aspects of software components.

Gazebo supports aspects of composition as follows:

- *Models* correspond roughly to the notion of simulation component, and may be active, such as robots, vehicles or humans, or passive, such as furniture. Models consist primarily of a 3D rigid body model with physical attributes in a format such as “xacro” definition. Moreover models are typically defined by reference to Gazebo’s plugin API, in particular for sensors and actuator controllers.
- *Actors* also correspond to our notion of simulation component but focus on a more limited, scripted notion of behaviour.
- *Worlds* are reusable entities defining a single simulation and its objects and associated artefacts.
- *Launch files* are reusable entities defining a configuration context for loading an entity such as a world, actor or model.

Models may be deployed separately in the sense that it is feasible to define models and include them in a simulation independently, without modification and reference to each other. However model definitions are declarative, and are simulated in a common simulation process in the same limited number of threads. Consequently model deployment constrained to a single host. Moreover models are typically defined by reference to plugins defining certain core behaviour such as for sensors or actuators, which must be available on the simulation process host to be loaded by Gazebo.

There are also additional limitations of the combined Gazebo/ROS ecosystem in relation to TF which make it difficult to include multiple models without TF frame name collisions. Although model definition supports a notion of *namespaces*, Gazebo’s conventions for translating model joint names into TF frame names may result in TF frames from separate objects interfering with each other. The preferred approach to dealing with this is to ensure a separate TF subsystem for each model by use a distinct master node system for each distinct model.

The notions of model composability and encapsulation may be regarded as complex, since objects may interact according to multiple physical aspects. The dominant complexity arises in relation to collision, with accompanying notions of friction, etc. Experience suggests that debugging and optimization of

simulations is significantly concerned with the unexpected consequences of such interactions.

Fidelity: Fidelity refers to the correspondence of the simulation to the physical system or device being simulated. Fidelity can be further distinguished for individual aspects such as physical correspondence or correspondence of a software user interface. The notion of fidelity in simulation can be traced back at least as far as the design of mechanical simulators [22].

Performance: For software components, performance refers to a collection of attributes which are either user-visible properties, such as response time or throughput, or computational resource requirements, such as system load or execution time[20].

IV. CASE STUDIES

This section presents our case studies, implemented in ROS and Gazebo, both on the construction of simulation artifacts and the measuring of latency.

A. FollowBot (VXLab)

FollowBot is specified as a MIR100 robot in the VXLab using computer vision to follow a designated human user wearing a coloured vest via a colour blob recognition algorithm, with the intention that the MIR100 will act as a mobile tool box carrier for the user. Due to delayed delivery of the MIR100 to VXLab, in the delivered implementation (see Fig. 1)⁴, FollowBot uses VXLab’s Rosie as a stand-in for the MIR100, combined with a digital twin of VXLab and a MIR100. The real time command stream of *Twist* messages generated by the vision and decision systems is sent both to Rosie and the digital twin.

B. DeliveryTeam (VXlab)

DeliveryTeam is specified as a pair of mobile robots collaborating on acceptance of a delivery to the VXLab. Due to COVID-19 restrictions, the prototype is built on a digital twin of VXLab and the robots. In the delivered implementation, VXLab’s robot Rosie navigates to the lab entrance and interacting with a proxy for the door control⁵. The digital twin of VXLab includes pre-integrated components for robots, as well as navigation and vision-guided pickup.

C. Corona Clean Team (VXLab)

Corona CleanTeam is specified as a pair of mobile robots collaborating on COVID-19 measures; a mix of direct measures such as cleaning/temperature monitoring and automation to remove human-human contact. Due to COVID-19 restrictions, robots located in VXLab are simulated, using an updated version of the digital VXLab twin for DeliveryTeam above. In the delivered implementation (see Fig. 2), a MIR100 robot navigates the lab, searching for people, and stopping to offer hand sanitizer to each person. The system incorporates additional configuration for e.g. navigation, off the shelf actors

⁴See also FollowBot video at <https://youtu.be/50XRS81Iv0>

⁵See also DeliveryTeam video at <https://youtu.be/aYiqchH3MK4>

Fig. 1. Baxter following operator in orange vest. Inset: Simulated MIR100 with clone of command stream from physical Baxter.

for humans, a custom hand sanitizer bottle model, and an off the shelf time of flight sensor-based vision system for the MIR100.

D. RoboCoStack (VXLab)

RoboCoStack is specified as enabling two distant human users to interact via a common virtual stack of toy blocks, realised at each location by local physical blocks and a robot avatar. The virtual stack could be used as an intuitive co-design tool; as users move blocks, both stacks stay in conformance as much as possible with the common virtual stack determined by the overall system which (somehow) resolves potential conflicts such as incompatible block moves. Due to COVID-19 restrictions, the system, consisting of two physical Baxter robots in separate locations, namely VXLab and RMIT University South Saigon Campus, are simulated. Development is an agile process progressively removing simplifications. Most importantly, in the delivered implementation (see Fig. 3): (i) a small number of blocks are used, (ii) blocks are marked with visual codes, or (iii) only one user at a time may rearrange blocks. The system is based on the digital VXLab twin for DeliveryTeam above, incorporating pre-existing components⁶.

E. DroneFlote (VXLab)

DroneFlote is specified as a drone controlled by a camera-based human motion tracking system; the primary goal is to implement and experiment with simple metric-based approaches to evaluating human posture, to direct different ways drones respond to the body. Due to COVID-19 restrictions, physical drones are simulated. In the delivered implementation (see Fig. 4), the user is tracked using Posenet, deployed in a web browser, with a simple metric based on the angle of lean of the tracked user to determine whether the drone stays stationary, rises or falls.

⁶See vision guided block grasping video at <https://youtu.be/Yw1vRZyYNg>

F. AGV Control in the Factory of the Future

Here, we built a Gazebo-based simulation environment of the Aalto Factory of the Future with an AGV (a MIR 100) operating inside. The project is primarily concentrating on the investigation of flexible manufacturing scenarios that require dynamic usage of the factory equipment for small lot sizes and the manufacturing of different products in the same facility at the same time. The AGV operates in the physical world while a simulation is happening in a virtual machine in parallel. One objective of this project is to measure the latency between simulation and physical world.

V. EVALUATION

Table I summarises our case studies with respect to composability, fidelity and performance, with detail as follows.

A. Composability

In Corona CleanTeam, two different robots are simulated. As discussed, this requires use of two ROS masters to ensure independent deployment and encapsulation of their respective navigation subsystems and associated robot state infrastructure.

An interesting exception to typical constraints arises in the case of telepresence prototypes as illustrated in RoboCoStack: there two robots interact *purely virtually*, thus a system simulation may be realized without compromising fidelity by creating two separate Gazebo simulations, with separate ROS masters, one for each robot. The two simulations are loosely coupled. This is in emulation of an expected real world implementation where the two robots are remote from each other and coupled via the ROS remote topic relay mechanism. Therefore typical limitations on composition and scalability, and therefore performance, need not apply: the two robots models need not be run in a single simulation, nor must the two simulations be run on a common host.

Fig. 2. Corona Clean Team in operation: (i) Full lab with MIR100 attending to human at door; (ii) Close-up view of MIR100 carrying sanitizer.

Fig. 3. RoboCoStack in operation: (a) scanning a stack with in-arm camera; (b) replicating stack in second location

B. Fidelity

Low fidelity might be expected to mask bugs in the real system: For example, in DeliveryTeam, due to development time limitations, several approximations are made, notably the door pushbutton is approximated by an off the shelf Gazebo model. In DroneFlote, simulation is based on emergent behaviour of the four drone rotor blades.

There was one instance where poor fidelity compromised the simulation but would not be expected directly to compromise the real-world case. In RoboCoStack, marked blocks drift at the default time step, affecting the integrity of the constructed block stacks and limiting the effective simulation time to a few minutes. However this would not be expected

to lead to issues in real world behaviour of the system.

Moreover, in developing the simulation platform used for Corona CleanTeam it emerged that the MIR100 is too short to be detected by Rosie's ring LIDAR. This had not been identified previously in risk assessments. Finally, use of combined physical and digital twins in FollowBot potentially enables comparison of physical and digital behaviours, including automatically, potentially reducing the weaknesses of purely physical or digital twins.

Simulation Latency: Simulation latency is an important aspect of fidelity especially when the simulation is incorporated in real time into control of the real system. Fig. 5 shows the experimental setup for measuring the delay between a

Fig. 4. DroneFlote in operation: (i) Human operator and PoseNet configuration; (ii) Simulated CrazyFly drone in Gazebo

Case study (Site)	Composition <i>How to optimally compose?</i> <i>- typically single process, single host</i>	Fidelity <i>Trade vs performance</i> <i>and cost</i>	Performance <i>Trade vs fidelity</i> <i>and cost</i>
FollowBot (VXLab)	Physical robot Rosie follows coloured vest. Twist message topic copied to MIR100 twin	Potential divergence due to different robots and initial states	
DeliveryTeam (VXLab)	Simulated robot Rosie navigates to door, manipulates “button”	Simulated lab; door “button” is approximate	
Corona CleanTeam (VXLab)	Two simulated robots: Rosie + MIR100. Separate ROS masters	Rosie cannot detect MIR100 (too short)	gzserver CPU: 531% gzclient CPU: 141%
DroneFlote (VXLab)	Single simulated CrazyFly drone controlled by human gestures	Simulation based on emergent behaviour of quadrotor blades	gzserver CPU: 137% gzclient CPU: 135%
RoboCoStack (VXLab)	Two simulated Baxters (2x Gazebo), “Slave” Baxter manipulates blocks with visual codes	Block drift—requires 10x finer timestep (slower) Marker recognition can be inaccurate.	gzserver CPU: 370 + 359% gzclient CPU: 154 + 152%
AGV in the AFoF (AFoF)	One simulated AGV	Comparison between real and simulated AGV	

TABLE I
CASE STUDY OVERVIEW

Gazebo based simulation and the real device (for details, see [21]).

The idea is to have a simulation environment in the cloud and a MIR 100 AGV in the local environment, both running in parallel and capable of seamlessly exchanging data. For testing purposes, the results of the simulation are displayed in a local machine next to the AGV. A script running on the same host as the web client (black block in Fig. 5), monitors the localization information published from the AGV. The data is obtained by subscribing to the corresponding ROS topic. Simultaneously, the same script captures the changes displayed on the 3D visualization, rendered by the Gzweb client through.

The fidelity test consists of executing a controlled iterative mission programmed in the AGV. The time stamps between the sources are compared to obtain the time difference (delay) between the visualization of the simulation and the local robot behavior. We measured an average delay of 387.53 ms with a

local setup in Finland and a cloud provider in the Netherlands.

C. Performance

Table I shows CPU consumed by gzserver and gzclient for selected projects, each running on their own host with identical CPU and memory. CPU consumed is shown as a percentage of one single CPU core, as observed by the Linux `top` utility in non-threaded mode. For RoboCoStack two separate gzserver/gzclient instances run, one per robot, with figures given for both. For all projects, percentages are greater than 100, signifying that gzserver is multithreaded and is in fact capable of utilizing more than a single core. Thus, for RoboCoStack, the sum of utilization is higher overall, since there is greater scope for parallelism as well as duplication of computation by the core simulation engine.

Fig. 5. Fidelity evaluation setup for cloud visualization

VI. CONCLUSION

We presented work on structuring automation simulation scenarios using components in ROS and Gazebo. A notion of simulation components was introduced. Different examples based on development case studies were evaluated with respect to component composition, fidelity and performance.

Future work comprises looking at more complex and distributed scenarios. In addition, we are working on an estimator to predict latency delays. These can be used to increase the fidelity of simulated components in real-time or near real-time simulations.

Another line of exploration would exploit simulation (model) component *validity*, which in contrast to fidelity, emphasises fitness for purpose, and thus relates to notions of parameterized contracts and their effective representation [24].

Acknowledgements: The authors thank Roope Pääkkönen, and Nicholas Huppert, Joseph Williams, Bryce George, Susannah Huang and Thomas Pfundt for their contribution of screen shots to this paper.

REFERENCES

- [1] Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. "Digital twin-driven product design, manufacturing and service with big data." *The International Journal of Advanced Manufacturing Technology*, 94(9-12), 3563-3576. Springer, 2018.
- [2] Boschert, Stefan; Rosen, Roland. "Digital twin—the simulation aspect." In: *Mechatronic futures* (pp. 59-74). Springer, Cham, 2016.
- [3] Kritzing, W., Karner, M., Traar, G., Henjes, J., & Sih, W. "Digital Twin in manufacturing: A categorical literature review and classification." *IFAC-PapersOnLine*, 51(11), 1016-1022. IFAC 2018.
- [4] Kuliaev, V., Atmojo, U. D., Sierla, S., Blech, J. O., & Vyatkin, V. "Towards Product Centric Manufacturing: From Digital Twins to Product Assembly." In *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)* (Vol. 1, pp. 164-171). IEEE, July 2019.

- [5] Quigley, Morgan, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. "ROS: an open-source Robot Operating System." In *ICRA workshop on open source software*, vol. 3, no. 3.2, p. 5. 2009.
- [6] Milgram, Paul, and Kishino, Fumio. "A taxonomy of mixed reality visual displays." *IEICE TRANSACTIONS on Information and Systems* 77.12 (1994): 1321-1329.
- [7] Peake, Ian D., Blech, Jan Olaf & Schembri, Matthew "A software framework for augmented reality-based support of industrial operations." *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2016.
- [8] Valner, R., Dydynski, J. M., Cho, S. & Kruusamäe, K. (2020). "Communication of Hazards in Mixed-Reality Telerobotic Systems: The Usage of Naturalistic Avoidance Cues in Driving Tasks." *Human Factors*, <https://doi.org/10.1177/0018720820902293>.
- [9] D. Aschenbrenner et al., "Mirrorlabs - creating accessible Digital Twins of robotic production environment with Mixed Reality," *2020 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, Utrecht, Netherlands, pp. 43-48, 2020.
- [10] Bejarano, Ronal, Pääkkönen, Roope, Blech, Jan Olaf, Peake, Ian, Herrmann, Peter, Vyatkin, Valeriy. "Assessing Long Distance Communication Alternatives for the Remote Control of AGVs." *2020 IEEE 25th International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2020.
- [11] Aschenbrenner, D., Li, M., Dukalski, R., Verlinden, J., & Lukosch, S. "Collaborative production line planning with augmented fabrication." In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (pp. 509-510). IEEE, March 2018.
- [12] Neema, Himanshu, et al. "Model-based integration platform for FMI co-simulation and heterogeneous simulations of cyber-physical systems." *Proceedings of the 10th International Modelica Conference*; March 10-12; 2014; Lund; Sweden. No. 096. Linköping University Electronic Press, 2014.
- [13] Meyer, J., Sendobry, A., Kohlbrecher, S., Klingauf, U., & Von Stryk, O. "Comprehensive simulation of quadrotor UAVs using ROS and Gazebo." In *International conference on simulation, modeling, and programming for autonomous robots* (pp. 400-411). Springer, Berlin, Heidelberg, November 2012.
- [14] Qian, W., Xia, Z., Xiong, J., Gan, Y., Guo, Y., Weng, S., ... & Zhang, J. "Manipulation task simulation using ROS and Gazebo." In *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)* (pp. 2594-2598). IEEE, December 2014.
- [15] Sokolov, M., Lavrenov, R., Gabdullin, A., Afanasyev, I., & Magid, E. "3D modelling and simulation of a crawler robot in ROS/Gazebo." In *Proceedings of the 4th International Conference on Control, Mechatronics and Automation* (pp. 61-65), December 2016.
- [16] Szyperski, C., "Component Software," Addison-Wesley Professional, 2002.
- [17] De Roeper, Willem-Paul, Hans Langmaack, and Amir Pnueli, eds. "Compositionality: The Significant Difference: International Symposium," COMPOS'97 Bad Malente, Germany, September 8–12, 1997 Revised Lectures. Springer, 2003.
- [18] Peake, I., Blech, J. O., Fernando, L., Schmidt, H., Sreenivasamurthy, R. and Sudarsan, S. D. "Visualization facilities for distributed and remote industrial automation: VxLab." In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)* (pp. 1-4). IEEE, September 2015.
- [19] Simic, M, Spichkova, M, Schmidt, H & Peake, I 2016, "Enhancing learning experience by collaborative industrial projects," in Ataur Rahman, Vojislav Ilic (ed.) *Proceedings of the International Conference on Engineering Education and Research (ICEER 2016)*, Sydney, Australia, 21-24 November 2016.
- [20] Koziolek, H. "Performance evaluation of component-based software systems: A survey," *Performance evaluation* 67(8), 634-658, Elsevier 2010.
- [21] Pääkkönen, Roope. "Cloud-based Remote Control of Autonomous Mobile Robots in Industrial Environments." MS-Thesis, Aalto University, 2020.
- [22] Cox, John A., et al. "Functional and Appearance Fidelity of Training Devices for Fixed-Procedures Tasks." HUMRRO-TR-65-4. George Washington University Alexandria VA human resources research office, 1965.
- [23] Schlesinger, Stewart. "Terminology for model credibility." *Simulation* 32.3 (1979): 103-104.
- [24] Ian D. Peake, Heinz W. Schmidt: "Systematic simplicity-accuracy trade-offs in parameterised contract models." *QoSA/ISARCS 2011*: 95-104