MBSE INCORPORATING TIME-DEPENDENT BEHAVIOR FOR THE DESIGN OF ROBOT-LIKE SYSTEMS

Ziegler, Klara;
Volpert, Marcus;
Amm, Maximilian;
Vogel-Heuser, Birgit;
Stahl, Karsten;
Zimmermann, Markus

Technical University of Munich

ABSTRACT

Complex systems typically consist of many components and are subject to many requirements. Approaches like the V-Model support complex systems design by providing guidelines on how to break down large systems into smaller pieces. Models built with SysML provide documentation on an abstract level. However, neither incoporates detailed information on components that may be relevant for design decisions. In robot-like systems, e.g., the choice of transmissions will depend on the system dynamics of the robot. This is modeled in the time domain, where detailed time-dependent component interaction is considered. The design perspective, however, is best represented in the property domain. Here, requirements on static component properties are formulated.

This paper presents a generic approach that connects the property and time domains to enable early-stage design decisions. The approach is applied to a 1-link robot system with a simple demonstrator transmission model, including properties that are typically not considered in the early design phase, like a nonlinear stiffness characteristic with backlash.

Keywords: Systems Engineering (SE), Product modelling / models, Simulation, Mechatronics, multi-disciplinary design

Contact:
Ziegler, Klara
Technical University of Munich, TUM School of Engineering and Design
Germany
klara.ziegler@tum.de

Cite this article: Ziegler, K., Volpert, M., Amm, M., Vogel-Heuser, B., Stahl, K., Zimmermann, M. (2023) 'MBSE Incorporating Time-Dependent Behavior for the Design of Robot-Like Systems', in *Proceedings of the International Conference on Engineering Design (ICED23),* Bordeaux, France, 24-28 July 2023. DOI:10.1017/pds.2023.259

1 INTRODUCTION

Complex systems such as robot-like systems must fulfill many requirements and are made of multiple components from different domains, e.g., transmissions from the mechanical domain, motors from the mechatronic domain, and controllers from the control domain. Components must be designed and selected to fulfill all requirements on the system level. One important requirement for robot-like systems is that the position accuracy does not exceed a threshold value at the end-effector position. It is affected by the performance measures of the components, e.g., transmission properties like backlash.

The V-model from VDI 2206 (VDI, 2021) is used to systematically develop complex systems. (VDI, 2021) It breaks down requirements from the system level to each component and can deal with different disciplines. Through verification of the system performance with the requirements, iterations are part of the process. Zimmermann and Hoessle (2013) concretized the V-model and avoid iterations during the design process. According to the system requirements, intervals for the design variables, so-called solution spaces, can be found. Rötzer et al. (2020) present a method to build a system model out of modular mathematical models. The modularized models include the properties of the system but not the time-dependent behavior.

Model-based systems engineering (MBSE) supports the design process by using models instead of documents (International Council on Systems Engineering, 2007). The semi-formal language SysML supports the interdisciplinary design of complex systems with a particular focus on cooperation in the design process.

The method presented in this paper shows how to structure component behavior models from different domains so they can be easily integrated into a system model for quantitative simulation in the time domain. At the same time, it enables model-based systems engineering on a more abstract level in the property domain, with a focus on the relationship between component and system properties. The system performance can be quantified, and components can be selected or designed according to requirements derived from the system level. The approach provides the possibility to include detailed descriptions of components and multidimensional characteristics. The approach is implemented for a robot-like system with a particular focus on the selected detail properties of a transmission.

2 RELATED WORK

2.1 MBSE

Model-based systems engineering (MBSE) is used as a formalized application of models to support the design phase of a system and throughout the development process. Engineers can apply MBSE through the graphical modeling language SysML introduced by the Object Management Group (OMG) in 2007. SysML is used to model the behavior, structure, and, signal or energy flow of complex systems. SysML is defined by a metamodel and it is a semi-formal language with a partially defined syntax (The Object Management Group).

Zerwas et al. (2022) propose a unified structure for system elements and a model signature to integrate existing domain models into system models automatically. They claim that following the structure, they can be connected when, e.g., the dimensions, data types, and units match. However, it has not yet been demonstrated how this translates into a simulation process in the time domain reproducing physical behavior or a structured design process that leads from top-level requirements to a solution.

Kernschmidt et al. (2018) propose SysML4Mechatronics for interdisciplinary design. It is restricted, however, to structural aspects. This SysML extension includes software, electrical, and mechanical views of the different interfaces and allows applying inconsistency checks with attributes for domain-specific ports of components.

Wolny et al. (2020) identified ten papers that extract SysML models for Simulink simulation; two papers should be highlighted. First, Rösch et al. (2012) propose a software-based support for engineers transforming SysML models to Simulink code. This specific approach, however, simplifies the development of closed-loop control software for programmable logic controllers and cannot be regarded as an approach for simulating the time-dependent behavior of a system in general. Second, Barbieri et al. (2014) proposed using SysML as a joined and more generalized model and integration platform that is used to couple to the discipline-specific simulation models and tools.

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

Simulink is a numerical simulation tool for hardware-in-the-loop testing, physical models, and rapid prototyping (The MathWorks, Inc.). It can be used to simulate the specific time-related behavior of a system. Mathematical models for the individual components and their interaction must be available. Simulink has a graphical representation of models and a defined syntax and can be used to investigate the time behavior via a system simulation. However, Simulink does not offer a more abstract description in the property domain with an appropriate model that can be used for design decisions in the early stage.
The different modeling and simulation approaches are not sufficient to describe the time behavior of systems holistically and on different abstraction levels to support design decisions. Models based on SysML allow a high abstraction level and basic time-related modeling methods, which can for specific components like controllers be transferred to Simulink to simulate the time behavior of systems. However, this is so far only possible for programmable logic controllers belonging to the automation domain and not yet applicable in other domains and is therefore not generalizable. On the other hand, Simulink does offer the possibility to simulate in the time domain but does not offer the possibility to systematically design systems. The resulting research gap consists of a method that can model and compute the time behavior of systems at different levels of abstraction and support design decisions.

2.2 Transmissions in robot-like systems

Transmissions are used to convert torque and rotational speed. By using transmissions in robot-like systems, the size of the motor and power train can be reduced. Transmissions in robot-like systems are characterized by a high transmission ratio, a compact size, and a coaxial input and output shaft. Typical types used are strain wave gears, cycloidal gears, and planetary gears. So far, there are no adequate calculation models for these transmissions that would be needed to compare the different types and select a transmission systematically. (Vogel-Heuser et al., 2020)
Rosenbauer (1995) provides some characteristics that are measured experimentally. He points out seven relevant assessment criteria that influence the accuracy of the robot. The most important criterion is the static load deformation. It depends on the backlash, torsional stiffness, and the hysteresis effect. Rosenbauer also measures the static load deformation after some load cycles. The results show a change in the static load-deformation curve, especially a higher compliance after some load cycles.

2.3 Solution space engineering

Solution space engineering supports the design of complex systems by carefully bookkeeping of system and component properties on various levels of abstraction to enable design. The abstraction levels depend on the relevant scope. Important abstraction levels can be: system, subsystem, component, which can be extended to several levels. Three steps are followed, which are shown in Figure 1. In the first step, *framing*, the qualitative dependency between quantities of interest (QOI), which are related to the customer requirements, and the design variables (DV), which can be influenced by an engineer, are determined. In the second step, denoted as *evaluation*, quantitative models to map DVs onto QoIs on multiple hierarchical levels are created. These quantitative models take time-dependent behavior into account and can be, e.g., provided by Simulink simulation. It is important, however, that an interface in the property domain is available for the following step. So far, there is no structured approach to take time-dependent behavior systematically into account when using solution space engineering or to connect several simulation models in the time domain and enable them to be designed with solution space engineering. In the third step, *design*, requirements on the system level are broken down into requirements on properties of components or subsystems, expressed as permissible intervals for their design variables. This enables design work or supports component selection on various abstraction levels. (Zimmermann et al., 2017)

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

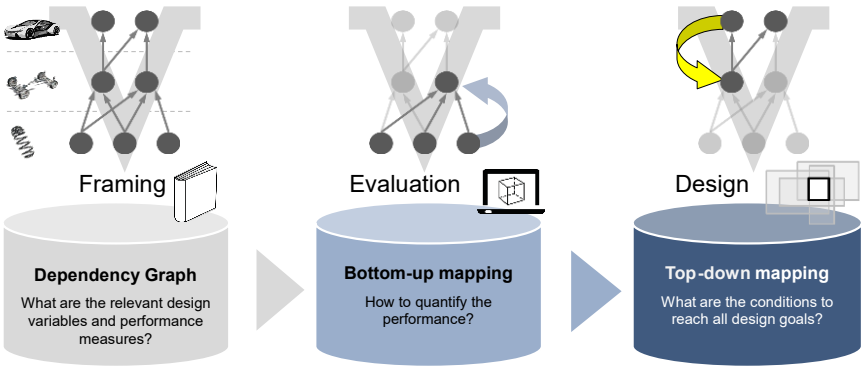

Figure 1. The three steps of solution space engineering, adopted from (Zimmermann et al., 2017)

3 METHOD

The approach follows the first two steps of solution space engineering. In the first step, the framing (1) of the system is done by determining the relevant quantities of interest, which are determined by selecting relevant customer requirements. The dependencies between the quantities of interest (QOI), which are related to the system requirements, and the design variables (DV) that can be adjusted by an engineer are visualized in a so-called attribute dependency graph (ADG), seen in Figure 2 (a); see also Rötzer et al. (2022). The ADG includes only the *properties* of the system and no time-dependent behavior. The ADG does not permit feedback loops to avoid iterations in the product development process. A model in the time domain is needed to compute detailed system behavior. In modeling step (2), the system model is implemented. In this example, Simulink is used because of its good modularity and the ability to simulate in the time domain. The components and the level of detail have to be defined to set up a system model. The interfaces between the components have to be defined. The system model, represented by a block diagram, is able to simulate time-dependent problems but does not visualize the properties of the single component. The connection of the time and property domains is an enabler in determining component properties to fulfill requirements on the system level in systems with time-dependent behavior. A component behavior model is needed to connect the property domain and the time domain. It connects time-dependent input with time-dependent output. In contrast, a component property model maps properties onto higher level properties, e.g., design variables onto quantities of interest. Every component is modeled independently. That means that the definition of a single component does not affect the definition of other components. The last step is the design and so-called top-down mapping (3). It is not included in this paper and will be presented in future work.

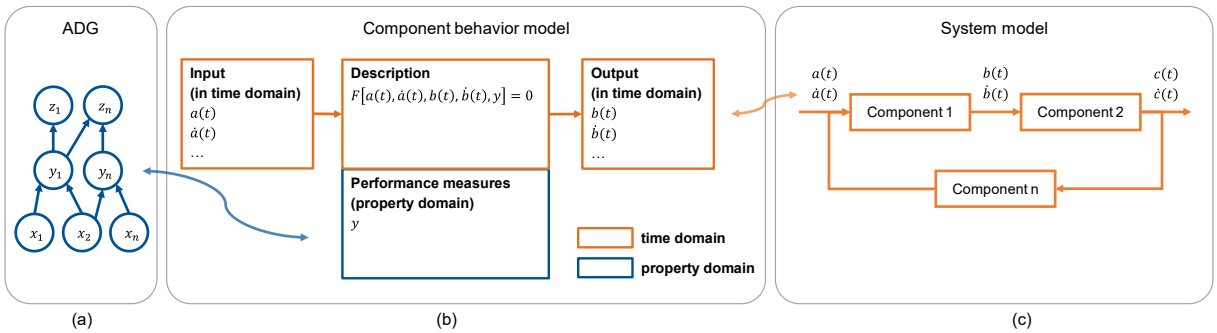

Figure 2. Elements to connect the time and property domains (1) ADG (property domain), (2) Component behavior model (connecting time and property domains), (3) System model (time domain)

3.1.1 Structure of the system model

A system model, shown in Figure 2 (c) as a block diagram, consists of one model, visualized by a box, for each component. The arrows show the flow of information between the components. The signals

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

passed from one component to the other define their interfaces. The properties can be from the mechanical, electrical, or software domain. The system model is used in a time-dependent simulation.

3.1.2 Structure of the component behavior models

Each component behavior model has the same structure as in Figure 2 (b). A component behavior model has a defined interface (input and output) to be integrable into the system model. It is characterized by relevant performance measures, which are the properties of the component. They determine the behavior of the component by mapping from input onto output, e.g., by a formula, a differential equation, or an entire black-box simulation. This part is introduced as description. The orange part of the model represents its action in the time domain. It is part of the system model. The component performance measures serve as design variables (DVs) in systems design. They are shown in the blue box. The component behavior models can be used for different instantiations of a component of one type. For this, the values of the performance measures need to be adjusted.

4 IMPLEMENTATION EXAMPLE

To implement the approach, the software MATLAB/Simulink was used. The approach is tested in the field of robot-like systems. To study the influence of a component behavior model on the system performance and to design the component, two simplifications are made: (1) a simple model of a 1-link robot is chosen, and (2) every component is modeled as ideal except the transmission model.
The 1-link robot in Figure 3 consists of a rotational joint that moves in the horizontal direction, like in a SCARA robot. The power train consists of the transmission, motor, sensor, and controller.

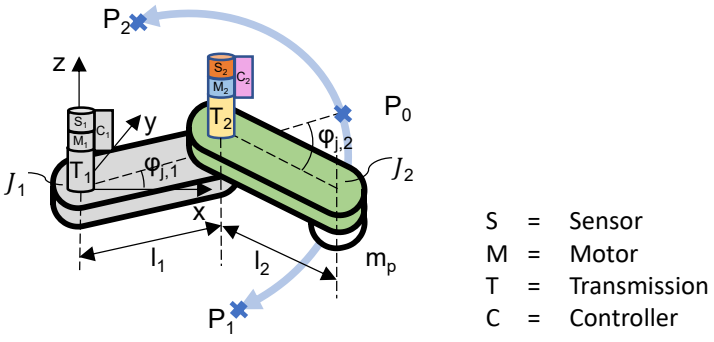

Figure 3. Elements, performance measures, and trajectory of a 1-link robot

The three steps introduced in the previous chapter are applied to design a 1-link system that fulfills the requirements.

4.1 Framing

Typically, SCARA robots have to fulfill many requirements (ABB, 2021). In this paper, the focus is on three requirements. The aim of the paper is not completeness but to demonstrate the concept for a selected example of requirements shown in Table 1.

Table 1. Requirement list for the example problem

			Value			
No.	**Description**	**Variable**	**Min.**	**Exact**	**Max.**	**Unit**
1	Cycle time	t_c			2	s
2	Position accuracy	AP_P	-0.1		0.1	mm
3	Coordinates start and end position			(0/-0.5/0) (0/0.5/0)		m

From the requirements, the quantities of interest on the top level of the ADG in Figure 4 are derived. They are influenced by the performance measures on the component level, like transmission ratio i, backlash $\Delta\varphi_B$, efficiency $\eta(\mathrm{T_M}, \dot{\varphi}_M)$ and load-dependent angle $\varphi_L(T_T)$. At the bottom level of the

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

ADG, design variables for detailed design can be added, like the number of teeth z_1 and z_2. The quantities of interest are also influenced by the properties of the other components besides the transmission.

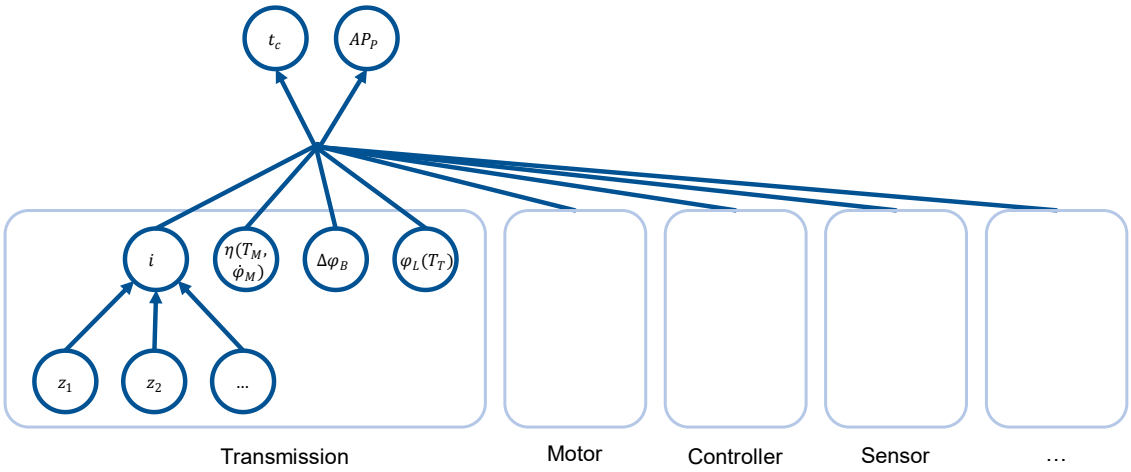

Figure 4. ADG of a 1-link robot

The trajectory is defined according to Figure 3. The link length l_2 is 0.5 m. The robot starts from the homing position (P0), moves three times between the end positions P1 and P2, and stops in the homing position. As the actual behavior of the robot depends on the previous state, multiple cycles are observed. The cycle time t_c is defined as the time for a given trajectory that the robot has to follow multiple times. In this example, a cycle is defined as a movement from P1 to P2 to P1. At each point, a waiting time is implemented, in which the robot usually fulfills its task.

4.2 Modeling

Once the framing is completed, the modeling is applied to the 1-link system. First, the system model is set up. Then the components are modeled with reference to the ADG and integrated into the system model.

4.2.1 System model

The system model of the 1-link system is shown in Figure 5. It includes the path planning that calculates the desired position $q_{ref}(t)$ depending on the time. The controller has as input $q_{ref}(t)$ and $q_s(t)$, which is the position measured by the sensor. It outputs the demanded torque $u(t)$ to the motor. The motor outputs the realized torque $T_M(t)$ and the current angle $\varphi_M(t)$. These quantities are the input for the transmission. The output of the transmission are the torque $T_T(t)$ and the current position $\varphi_T(t)$. The dynamic block calculates the acceleration of the joint, which after integration, gives the current joint position $q_j(t)$. The direction of information flow indicates the source of the produced values. E.g. the sensor produces a signal as output based on an observed state as input.

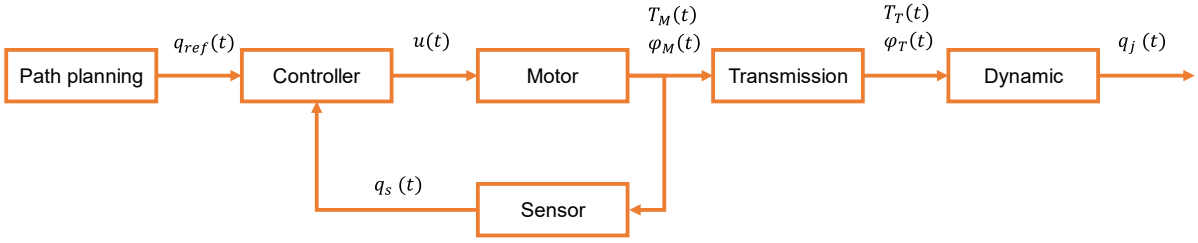

Figure 5. System model of a 1-link robot

As all the component behavior models are shown as black boxes in the system model, the setup of a component behavior model is presented in the next section, with the example of the transmission model.

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

4.2.2 Transmission model

The transmission model presented in Figure 6 is not ideal. The properties of the model are specified as performance measures in the blue box. In the model definition, they are defined as parameters. A data model is needed, and specific values are inserted to use the model for computation. Most important for the general behavior of the transmission is the transmission ratio i. Also included is the efficiency $\eta(\mathrm{T_M}, \dot{\varphi}_\mathrm{M})$. The performance measures that affect the accuracy the most, like backlash $\Delta\varphi_B$ and load-dependent angle $\varphi_\mathrm{L}(\mathrm{T_T})$, are included in the property domain of the model. Effects that lead to hysteresis are also relevant but not generally describable for all possible cases.

As the main functionalities of transmissions are to convert speed and torque, the input of the model is the torque of the motor $T_M(t)$ and the angle after the motor $\varphi_M(t)$. The output of the model is the transmission torque $T_T(t)$ and the angle after the transmission $\varphi_T(t)$. The nominal value for the output $T_T(t)$ is calculated with the transmission ratio i by

$$T_T(t) = T_M(t) * i * \eta(T_M(t), \dot{\varphi}_M(t)). \tag{1}$$

As the efficiency affects the output torque, it is included in the model as a constant value of $\eta(\mathrm{T_M}, \dot{\varphi}_\mathrm{M}) = 0.8$ derived from Pham and Ahn (2018).

The output angle of the model is affected by the backlash $\Delta\varphi_\mathrm{B}$ and the load-dependent angle $\varphi_\mathrm{L}(\mathrm{T_T})$. In the component behavior model, half of the backlash is added for a positive torque $T_T(t)$ and subtracted for negative torque. The output angle is calculated by subtracting the backlash and load-dependent angle from the ideal angle.

$$\varphi_T(t) = \frac{\varphi_M(t)}{i} - \begin{cases} \varphi_L(T_T(t)) + \Delta\varphi_B, T_T(t) \geq 0 \\ \varphi_L(T_T(t)) - \Delta\varphi_B, T_T(t) < 0 \end{cases} \tag{2}$$

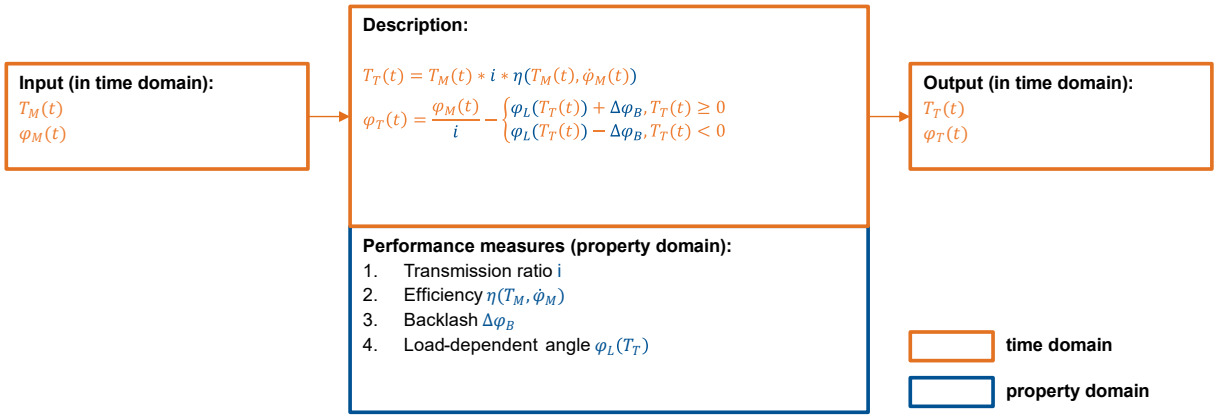

Figure 6. Transmission model with backlash and stiffness

The behavior of the model can be seen in Figure 7.

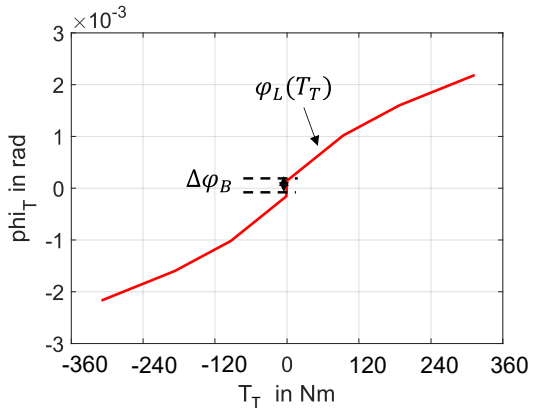

Figure 7. Stiffness of transmission model

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

4.2.3 Other component behavior models

As the approach is explained with a focus on the detailed model of a transmission, the other component behavior models are kept as simple as possible. The motor and the sensor are defined as ideal, which leads to the following relations:

$$T_M(t) = u(t) \tag{3}$$

$$q_s(t) = \varphi_M(t) \tag{4}$$

The path planning calculates the trajectory with the given points. A time waiting time of 0.5 s is defined between each point. This ensures a cycle time of 2 s to fulfill the requirements. The velocity is defined as a trapezoidal velocity profile.

The controller is set up as PD Controller, as proposed in Mareczek (2020). In order to adjust the controller, an ideal transmission is chosen, and the difference between $q_{ref}(t)$ and $q_s(t)$ minimized in the waiting position.

The dynamic block is already defined in Simulink and computes the joint acceleration according to the equation of motion

$$M(q_j)\ddot{q}_j = -C(q_j, \dot{q}_j) - G(q_j) - J(q_j)^T F_{Ext} + \tau \tag{5}$$

with the mass matrix $M(q_j)$, the Coriolis term $C(q_j, \dot{q}_j)$, the gravity torques and forces $G(q_j)$, the geometric Jacobian matrix $J(q_j)$, the external forces F_{Ext}, and the joint torques and forces τ (The MathWorks, Inc.). After setting up the parametrized component behavior model and including it in the system model, either the performance for one set of DVs can be evaluated, or solution spaces can be calculated.

4.3 Experimental setup

The setup in Table 2 is chosen for simulation to evaluate the influence of the transmission performance measures on the QOIs. The setup for h^0 represents an ideal transmission, and for t^1 and t^2, two transmissions with different values for backlash and load-dependent deformation. $\varphi_L(T_T)$ in the transmission model is approximated with three support points.

Table 2. Simulation setup

Simulation label	h^0	t^1	t^2
$\Delta\varphi_B$	0	$1\ arcmin$	$0.5\ arcmin$
$\varphi_L(T_T)$	0	$P_1(75\ Nm, 3\ arcmin)$ $P_2(150\ Nm, 5\ arcmin)$ $P_N(250\ Nm, 7\ arcmin)$	$P_1(75\ Nm, 1\ arcmin)$ $P_2(150\ Nm, 4\ arcmin)$ $P_N(250\ Nm, 6\ arcmin)$

According to DIN EN ISO 9283, the position accuracy is defined as the difference between the demanded and the actual position:

$$AP_P = \sqrt{(\bar{x} - x_c)^2 + (\bar{y} - y_c)^2 + (\bar{z} - z_c)^2} \tag{6}$$

When the robot reaches the end position, the maximum error is measured to determine the position accuracy.

5 RESULTS

The demanded position and the resulting positions are presented in Figure 8. The position of the ideal transmission follows the demanded position the best with a maximum accuracy of $3.41 * 10^{-5}\ m$. When the SCARA reaches the end position, the torque starts to oscillate to control the position. With every change from the sign of the torque, the position with backlash oscillates around the demanded position.

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

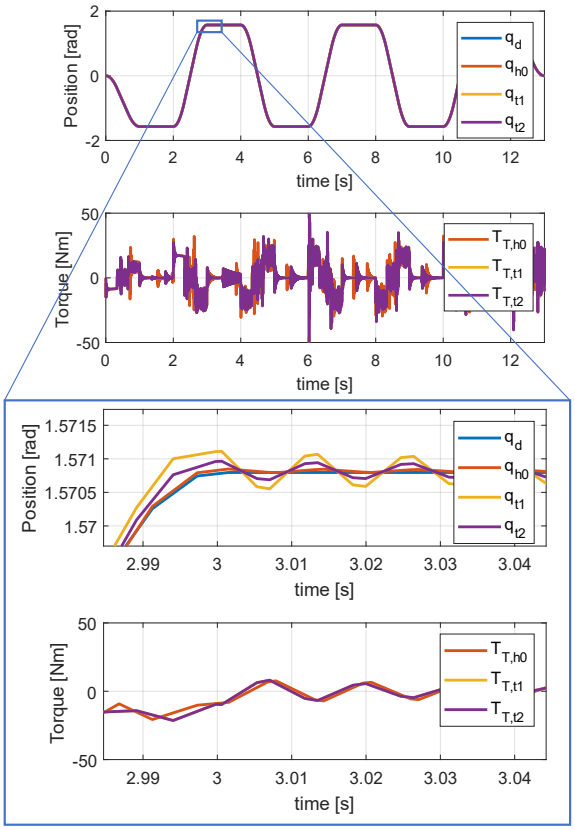

Figure 8. Demanded position (q_d), position with ideal transmission (q_{h0}), position with backlash and load-dependent effect for two different transmissions (q_{t1}, q_{t2})

Table 3. Position accuracy at the end position

Simulation label	h^0	t^1	t^2
$AP_{P,max}$	$3.41 * 10^{-5}\ m$	$1.03 * 10^{-4}\ m$	$8.39 * 10^{-5}\ m$

In order to design a 1-link system, the requirements must be fulfilled. As the cycle time and the end positions are predetermined, it must be ensured that the requirement on the accuracy is fulfilled. In this case, the requirement on position accuracy is not fulfilled for transmission t^1. In order to meet the requirement with the transmission model, the values of the performance measures, like stiffness and backlash, have to be adjusted, or a different transmission has to be chosen. Transmission t^2 fulfills the requirement on the posistion accuracy due to its higher stiffness and lower backlash. The system design is sufficient when an appropriate component like transmission t^2 is chosen.

6 DISCUSSION AND CONCLUSION

In this paper, an approach is presented that connects the time domain and property domain using a standardized component behavior model. The component behavior model includes properties that are documented in a so-called attribute dependency graph (ADG) to enable systematic requirement development from the system to the component level. These properties are also included in a component behavior model and determine how input is mapped onto output in the time domain, thus enabling the simulation of system behavior. This way, it is possible to adopt a more abstract view, e.g., to design product architectures in the early stage of the product development process, as well as to evaluate and design detailed components. A component behavior model of a transmission, including the properties backlash and load-dependent deformation, was presented and integrated into the system model of a 1-link robot. The position accuracy was assessed. It was shown how the component properties influence the position accuracy and that different components can lead to an inadequate or adequate system design. This simple, standardized component behavior model can be easily extended to other components and be used to integrate different disciplines.

https://doi.org/10.1017/pds.2023.259 Published online by Cambridge University Press

In further work, the approach will be used to design systems with many components from different disciplines and conflicting requirements by computing solution spaces.

ACKNOWLEDGMENTS

The authors thank the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) for funding this work under project number 461993234.

REFERENCES

ABB (2021), IRB 910SC.

Barbieri, G., Kernschmidt, K., Fantuzzi, C. and Vogel-Heuser, B. (2014), "A SysML based design pattern for the high-level development of mechatronic systems to enhance re-usability," *IFAC Proceedings Volumes*, Vol. 47 No. 3, pp. 3431–3437. http://doi.org/10.3182/20140824-6-ZA-1003.00615

International Council on Systems Engineering (INCOSE) (2007), "SYSTEMS ENGINEERING VISION 2020". Technical paper no. INCOSETP-2004-004-02, 2007.

Kernschmidt, K., Feldmann, S. and Vogel-Heuser, B. (2018), "A model-based framework for increasing the interdisciplinary design of mechatronic production systems", *Journal of Engineering Design*, Vol. 29 No. 11, pp. 617–643. http://doi.org/10.1080/09544828.2018.1520205

Mareczek, J. (2020), *Grundlagen der Roboter-Manipulatoren – Band 1*, Springer Berlin Heidelberg, Berlin, Heidelberg.

Pham, A.-D. and Ahn, H.-J. (2018), "High Precision Reducers for Industrial Robots Driving 4th Industrial Revolution: State of Arts, Analysis, Design, Performance Evaluation and Perspective", *International Journal of Precision Engineering and Manufacturing-Green Technology*, Vol. 5 No. 4, pp. 519–533. http://doi.org/10.1007/s40684-018-0058-x

Rösch, S., Schutz, D., Bayrak, G. and Vogel-Heuser, B. (2012), "Supporting integrated development of closed-loop PLC control software for production systems", *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, 25.10.2012 - 28.10.2012, Montreal, QC, Canada, IEEE, pp. 6185–6190. http://doi.org/10.1109/IECON.2012.6389069

Rosenbauer, T. (1995), *Getriebe für Industrieroboter: Beurteilungskriterien, Kenndaten, Einsatzhinweise*, Zugl.: Aachen, Univ., Diss., 1994, Berichte aus der Produktionstechnik, Vol. 94,32, Als Ms. gedr, Shaker, Aachen.

Rötzer, S., Rostan, N., Steger, H.C., Vogel-Heuser, B. and Zimmermann, M. (2020), "Sequencing of Information in Modular Model-based Systems Design", *in DS 103: Proceedings of the 22nd International DSM Conference (DSM 2020)*, MIT, Cambridge, Massachusetts, October 13th - 15th, 2020, The Design Society, p. 10. http://doi.org/10.35199/dsm2020.7

Rötzer, S., Schweigert-Recksiek, S., Thoma, D. and Zimmermann, M. (2022), "Attribute dependency graphs: modelling cause and effect in systems design", *Design Science*, Vol. 8. http://doi.org/10.1017/dsj.2022.20

The MathWorks, Inc., "Robot Dynamics", available at: https://de.mathworks.com/help/robotics/ug/robot-dynamics.html (accessed 29 November 2022).

The MathWorks, Inc., "Simulink. gemacht für Model-Based Design", available at: https://de.mathworks.com/products/simulink.html (accessed 29 November 2022).

The Object Management Group, "WHAT IS SYSML?", available at: http://www.omgsysml.org/what-is-sysml.htm (accessed 29 November 2022).

VDI (2021), *Development of mechatronic and cyber-physical systems* No. VDI/VDE 2206.

Vogel-Heuser, B., Zimmermann, M., Stahl, K., Land, K., Ocker, F., Rötzer, S., Landler, S. and Otto, M. (2020), "Current Challenges in the Design of Drives for Robot-Like Systems", in 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 11.10.2020 - 14.10.2020, Toronto, ON, Canada, IEEE, pp. 1923–1928.

Wolny, S., Mazak, A., Carpella, C., Geist, V. and Wimmer, M. (2020), "Thirteen years of SysML: a systematic mapping study", *Software and Systems Modeling*, Vol. 19 No. 1, pp. 111–169. http://doi.org/10.1007/s10270-019-00735-y

Zerwas, T., Jacobs, G., Kowalski, J., Husung, S., Gerhard, D., Rumpe, B., Zeman, K., Vafaei, S., König, F. and Höpfner, G. (2022), "Model Signatures for the Integration of Simulation Models into System Models", *Systems*, Vol. 10 No. 6, p. 199. http://doi.org/10.3390/systems10060199

Zimmermann, M. and Hoessle, J.E. von (2013), "Computing solution spaces for robust design", *International Journal for Numerical Methods in Engineering*, Vol. 94 No. 3, pp. 290–307. http://doi.org/10.1002/nme.4450

Zimmermann, M., Königs, S., Niemeyer, C., Fender, J., Zeherbauer, C., Vitale, R. and Wahle, M. (2017), "On the design of large systems subject to uncertainty", *Journal of Engineering Design*, Vol. 28 No. 4, pp. 233–254. http://doi.org/10.1080/09544828.2017.1303664