Towards Spatial Conceptual Modeling for Robotic Digital Twins Based on URDF

Daniel Borcard[1,*,†], Hans-Georg Fill[1,†]

[1]University of Fribourg, Boulevard de Pérolles 90, 1700 Fribourg, Switzerland

Abstract

In this paper we report on the design and implementation of a metamodel for the Unified Robot Description Format (URDF) on the MM-AR metamodeling platform. The goal of this approach is to provide the foundation for the integration of digital twins of robots as specified in URDF with conceptual enterprise models. In particular, the MM-AR platform and its underlying meta^2 model permit the inherent integration of principles from spatial computing for enabling the representation and manipulation of conceptual models in spatial environments. Ultimately, this can be used to link behavioral specifications of robots on various abstraction levels with the structure and physical properties of robots as given by URDF.

Keywords

Robotics, Spatial Computing, Conceptual Modeling, Metamodeling

1. Introduction

Robotics is a complex technical domain combining multiple disciplines, such as mechanical engineering, computer science and specific application domains. On a general level, we can distinguish in robotics between the *structure* of robots and the *behavior* that they exhibit. The structure of robots can today be well described thanks to the Unified Robot Description Format (URDF), which is supported by various design tools such as SolidWorks [1], MATLAB[1] or Fusion 360 [2]. URDF is an XML-based format that specifies how a robot is designed in terms of physical geometry and movements. It is mainly used as a basis for simulation purposes [3, 4].

As explored in [5], the behavior of robots has been described through models using languages such as BPMN [6, 7], Petri nets [8] or behavior trees [9]. Because robots operate in spatial environments, representing and editing their behavior poses some challenges. In particular, modeling languages that have been previously used for this purpose lack inherent concepts for three-dimensional space. With the proposal of *spatial conceptual modeling*, it has been demonstrated how principles from spatial computing can be integrated on the meta level of modeling languages [10]. This makes it possible to consider spatial concepts inherently in any modeling language, including the corresponding tool support.

Therefore, we propose a combination of URDF with spatial conceptual modeling. The goal is to enable easier model-based design of robotic behaviors, while at the same time allowing users to view or edit the robot state using URDF concepts. To achieve this, we first need to represent URDF using spatial conceptual modeling concepts. For this purpose we rely on a meta^2 model for spatial conceptual modeling and a corresponding software implementation in the form of the MM-AR metamodeling platform. This allows us to provide a first demonstration of the interactions between behavioral models and 3D robotic architectures in a pick-and-place scenario.

With this article, we investigate the following research questions (RQs):

PoEM2025: Companion Proceedings of the 18th IFIP Working Conference on the Practice of Enterprise Modeling: PoEM Forum, Doctoral Consortium, Business Case and Tool Forum, Workshops, December 3-5, 2025, Geneva, Switzerland

*Corresponding author.

†These authors contributed equally.

daniel.borcard@unifr.ch (D. Borcard); hans-georg.fill@unifr.ch (H. Fill)

https://www.unifr.ch/inf/digits/en/group/team/daniel-borcard.html (D. Borcard);

https://www.unifr.ch/inf/digits/en/group/team/fill.html (H. Fill)

0000-0001-7211-4793 (D. Borcard); 0000-0001-5076-5341 (H. Fill)

published 2026-02-20

- *RQ1*: What added value does a URDF metamodel provide over using URDF files directly?
- *RQ2*: Which URDF concepts and relationships are necessary and sufficient for the URDF metamodel to faithfully represent fixed industrial robots and support digital twins?
- *RQ3*: How can spatial conceptual modeling be embedded in the URDF metamodel to align robot structure with process models?

RQ1 evaluates the benefits and trade-offs of a metamodel compared to a static, file-based approach. RQ2 delineates the core elements and associations the metamodel must capture. RQ3 focuses on integrating spatial conceptual modeling so that structural views are correctly represented in the 3D space.

The remainder of the paper is structured as follows: Section 2 introduces foundations required for describing the approach and related work. Section 3 presents how the URDF metamodel was derived and how it has been implemented in the MM-AR web tool. Section 4 then explores how the approach was applied to the pick-and-place scenario. Finally, in Section 5, we discuss the approach and how it relates to the field of Digital Twins. In Section 6 we conclude the findings and propose further research directions.

2. Foundations & Related Work

In this section we will present the necessary foundations and related work in robotics and URDF, knowledge-based digital twins, as well as spatial conceptual modeling.

2.1. Robotics & URDF

Robots come in various shapes and sizes. From small mobile units to robot vacuum cleaners [11] to large 6-Degree-of-Freedom (6-DoF) robotic arms in automotive construction [12]. Stationary robots, i.e., as opposed to mobile robots, are autonomous systems that are fixed to a specific point in space and cannot move by themselves [13]. This includes for example industrial 6-DoF arms (illustrated in Figure 5), delta robots, or SCARA robots. A static robot is composed of a set of interconnected *links* and *joints* that support the movement of an *end effector* through space [13]. A link is a rigid body connected to one or more other links by joints. A joint is a mechanical part that connects two rigid bodies [13]. Joints can be of different types depending on the desired motion and constraints, such as prismatic, revolute, or planar [14]. The end effector is a tool that allows the robot to perform its tasks. End effectors can be, for example, a gripper (e.g., pneumatic gripper), a processing tool (e.g., cutting tool), or a sensor (e.g., proximity sensor). By combining links, joints and end effectors, one can create various robot types with different capabilities and reach. This variation in shape, size and capabilities can lead to difficulties when it comes to the digital representation and manipulation of the robot.

For this purpose, the Unified Robot Description Format (URDF) was developed to standardize the representation of robots and their properties [3]. URDF is an XML-based format used in various robotic technologies such as the Robot Operating System (ROS) or robotic 3D modeling tools (e.g., Fusion 360). URDF allows the modeler to represent a robot composed of rigid links and joints. URDF links have attributes such as inertia, visual and collision that can be used for simulation purposes. Similarly, URDF joints are defined by their axis (i.e., rotation, translation or planar depending on the desired type of joint), limit and dynamics. Combining these aspects, URDF can be used for kinematics calculation, simulation and prediction in specific tools such as Gazebo [15] or RViz [16]. Gazebo is a popular open source robotic simulation tool developed by Nathan Koenig and Andrew Howard, and now maintained by Open Robotics and the broader community. Similarly, RViz is a 3D visualization tool for ROS that allows the visualization of sensor data, robot models, and coordinate frames in real time. It was originally developed at the Willow Garage robotic lab and is now maintained by the ROS and Open Robotics community. URDF, however, lacks semantic information about the robot. The Semantic Robot Description Format (SRDF) aims to address this by adding new concepts to URDF such as groups of

links and joints and end effectors. Xacro is an XML macro language used to construct and generate large URDF files by using for example macros, properties, math expressions and conditional blocks [3].

URDF itself also covers mobile platforms (e.g., bases with planar/floating joints and wheel assemblies). A complete mobile-robot digital twin additionally needs environment/localization frames (map/odom) and motion constraints, which are, for now, outside of the scope of this work.

URDF further allows representation of the physical aspects of a robot, but not its behavior. There is a gap between low-level definitions of robots and high level tasks. Robot programming itself is a complex undertaking that requires deep knowledge of the field. The use of conceptual modeling in this field aims to abstract from the technical aspects of a specific domain to make it more accessible for domain specialists to define robotic behavior.

2.2. Knowledge Based Digital Twin in Robotics

According to Singh et al. [17] a digital twin is a dynamic and self-evolving digital model that aims to represent a physical system. These digital models may pertain to four categories [18]:

1. **Geometric models** capture shape, size, internal structure, spatial frames, and poses. In robotics, this includes kinematic chains, link/joint topology, and collision/visual meshes.
2. **Physical models** represent material and dynamic properties (e.g., mass and inertia, stiffness/-compliance, thermal or electrical parameters). These enable virtual commissioning and quality control by allowing the twin to predict working envelopes, tolerance propagation, and energy consumption.
3. **Behavioral models** encode operational logic (i.e., sequential, concurrent, periodic, or event-driven) including control policies, task plans, and exception handling. They support monitoring, anomaly detection, and online parameter adaptation so that the twin's control/motion traces match observations from the shop floor.
4. **Rule models** formalize domain knowledge and life-cycle constraints (e.g., safety zones, capability and payload limits, maintenance policies). By codifying expert experience, they expose evolutionary patterns and enable intelligent decisions such as reconfiguration, scheduling, and compliance checks.

The digital twin concept is widely used in robotics [19]. For the scope of this paper, we introduce the notion of *knowledge-based digital twins*, in which the models of a digital twin are grounded in a common meta^2 model and can therefore be integrated with other conceptual and enterprise models. This integration enables the re-use of knowledge from other model types in digital-twin scenarios. For example, a user may define a pick-and-place task in some enterprise modeling scenarios for a 6-DoF robotic arm. Then, the knowledge-based digital twin can exploit properties of the geometric model to verify that the specified poses are physically reachable; the physical model can assess whether joint efforts remain within predefined constraints; the behavior model can analyze the path required to achieve the goal; and, finally, the rule model can check that the pick-and-place action does not violate system rules, such as entering a safety area. We therefore leverage the capabilities of URDF to analyze and simulate robotic tasks. By representing URDF models using spatial conceptual modeling, we can reference and re-use their concepts and visualizations in other conceptual models such as business process models or other types of enterprise models.

2.3. Spatial Conceptual Modeling

Spatial conceptual modeling integrates principles from spatial computing in conceptual modeling [10]. The goal is to anchor knowledge of conceptual models in the physical world. This may then be used for augmented reality (AR) or virtual reality (VR) applications, as well as applications that contain references to spatial entities in general, e.g., in architecture, cultural heritage, or digital twins. Such an integration enhances the interaction with models that refer to the physical space by inherently providing concepts for three-dimensional representations, such as location and transform properties. In addition,

spatial conceptual modeling can be used to represent further data requirements in spatial computing such as the representation of fields, objects, networks, and events. A first implementation for proving the feasibility of spatial conceptual modeling is provided in the form of the MM-AR metamodeling platform[1]. It has been used for implementing and evaluating a tool for the Augmented Reality Workflow Modeling Language (ARWFML), which makes it possible to create AR applications in a no-code fashion and integrate them with other types of modeling languages [20, 21, 22].

2.4. Related Work

Generating URDF files has been extensively studied, with most contributions focusing on the transformation pipeline between modeling tools and URDF artefacts [23, 24]. Schneider et al. present a DSL-based development process for specifying grasping problems, reusing established standards, such as URDF, to define robotic structure and dynamics [25]. While this strategy facilitates the integration of emerging standards, it lacks a structured modeling approach to systematically exploit URDF concepts.

Ringe et al. propose a metamodeling approach to classify robot morphologies by introducing a dedicated metamodeling language [26]. Thereby, they can represent a broad spectrum of robot morphologies, i.e., the physical structure, shape, and configuration of robots. However, despite its breadth, this approach does not benefit from the extensive ecosystem of tools that support URDF. MetaMorph therefore requires an explicit mapping back to the URDF format and does not solve the challenge of URDF generation.

With the approach we propose in the following we therefore aim to combine the strengths of the URDF ecosystem with the flexibility of conceptual enterprise modeling. Thus, we want to bridge the gap between robotic morphology definition and conceptual modeling, e.g., for defining robotic behavior.

3. Metamodel for URDF

For integrating URDF with the approach of spatial conceptual modeling we use the meta^2 model of the MM-AR approach and the corresponding metamodeling platform. In this way, we will show how URDF can be realized as a meta model using these foundations. We will first briefly present the MM-AR meta^2 model and then advance to the derivation of the URDF meta model including the expected user interface.

3.1. MM-AR Meta^2 Model and Platform

The MM-AR meta^2 model and the according implementation allow a flexible realization of modeling languages based on the approach of spatial conceptual modeling. The main concepts of the meta^2 model of MM-AR that will be required for describing the URDF meta model are as follows:

- *SceneType* denotes a template for a container of Classes with some common purpose, similar to diagram types in 2D modeling environments.
- *Class* denotes a template for objects that have a common structure in the form of Attributes.
- *Attribute* denotes a template for a property, which can be attached to SceneTypes or Classes and whose values are constrained through an AttributeTable or an AttributeType.
- *AttributeTable* denotes a template for a tabular structure of properties in the form of two or more Attributes.
- *AttributeType* denotes a template for constraining the value of an Attribute, e.g., a number, a mass, or a reference to another object. For values other than references to objects, regular expressions are used to constrain the range of values.

For integrating fundamental properties of spatial conceptual modeling, all Classes have pre-defined attributes for locations in different coordinate systems, transform attributes, as well as an attribute for

[1] https://github.com/MM-AR/mmar

the three-dimensional graphical representation. As a consequence, every object that is instantiated based on these concepts can be inherently represented in a 3D modeling environment. However, as will be shown in the next section for robotics, further attributes for spatial information may be needed depending on the application domain.

3.2. URDF Metamodel using MM-AR

URDF describes a robot as a set of *links* connected by *joints*. These links and joints have associated properties. In our URDF metamodel, links and joints are modeled as *Classes*. Both have attributes that capture the corresponding URDF properties.

Every concept in URDF has its counterpart in the URDF metamodel, which builds on the MM-AR meta^2 model. For example, *Origin* is an AttributeTable with the Attributes $x, y, z, roll, pitch, yaw$. Using a table serves a dual purpose: it aggregates recurrent concepts and enables the inclusion of a temporal perspective in the model with rows representing different states over time.

Figure 1 depicts the URDF metamodel of the *link* concept. The *Link* Class has three optional AttributeTables: *Inertial*, *Collision*, and *Visual*. The URDF `name` property is mapped to the default `Name` Attribute available for every object in the MM-AR meta^2 model; the same mapping applies to joint names.

The *Inertial* AttributeTable comprises two further AttributeTables: *Origin* and *Inertia* and one scalar attribute, *Mass*. *Inertia* represents the link's inertia tensor, *Origin* specifies the pose of the inertia frame, and *Mass* records the link mass.

The *Visual* Attribute table comprises three attribute tables: *Origin*, *Geometry*, and *Material*. *Origin* is as defined above. *Geometry* defines the shape of the visual object and can be a sphere, cylinder, or box, or a reference to an external 3D mesh (e.g., STL or glTF). *Material* may be a color defined by an RGBA value or a reference to an external specification (e.g., a `.material` file). Some 3D mesh files include embedded materials and textures; in such cases, a separate material attribute is unnecessary. The *Collision* AttributeTable contains the *Geometry* and *Origin* tables.

Figure 2 represents the URDF metamodel of the *joint* concept in URDF. This is a Class meaning that it is a visual instance. It has seven AttributeTables: *Origin*, *Axis*, *Calibration*, *Dynamics*, *Limit*, *Mimic* and *Safety Controller*. Parent and Child are references to links. The joint type attribute defines the type of joint that links two parts. This type can be one of the following: *revolute*, *continuous*, *prismatic*, *fixed*, *floating*, *planar*. *Joint* units are derived from the type of selected joint outside of the metamodel. For example, revolute joints uses radians whereas prismatic joint have units in meters. The axis attribute table reuses the *x,y,z* concept of the origin attribute table. The axis defines the joint axis in the joint frame. The x,y,z components represent a vector and must be normalized. The Dynamics AttributeTable defines the *Damping* and *Friction* attributes. They define the physical properties of the joint, that can be leveraged for simulation. The units are derived from *joint* types. For example, the damping in newton-seconds per meter for prismatic joints and in newton-meter-seconds per radian for revolute joints and the friction in newtons for prismatic joints, in newton-meters for revolute joints. The *Limit* attribute table defines the *Lower*, *Upper*, *Effort* and *Velocity* Attribute of a joint. This AttributeTable defines the limits of the joint. The *Calibration* attribute type defines the *Rising* and *Falling* attributes. They specify the joint position at which a calibration sensor or encoder index transitions high ("rising") or low ("falling") during homing. The *Safety* controller defines *soft lower limit*, *soft upper limit*, *k position* and *k velocity*. The soft limits are not physical limits but limits that can be defined by the user. K position scales the maximum permitted speed based on how close the joint is to the soft limits, while K velocity sets an overall speed ceiling, ensuring smooth deceleration near limits and preventing excessive velocities throughout the motion range. The *Mimic* attribute table contains the reference to a joint to mimic, the multiplier and the offset.

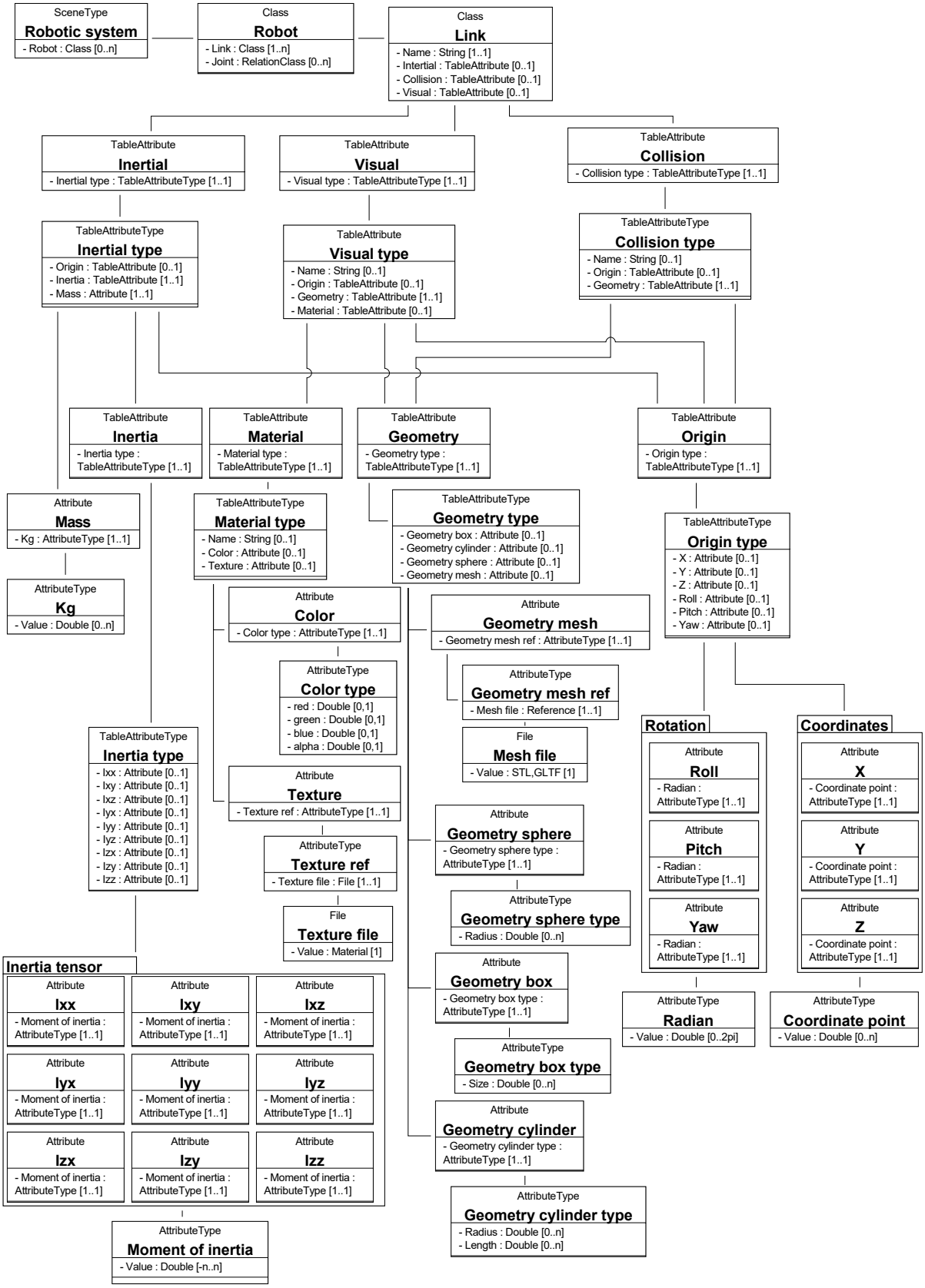


Figure 1: The URDF metamodel of the URDF link. (navigability suppressed for clarity)

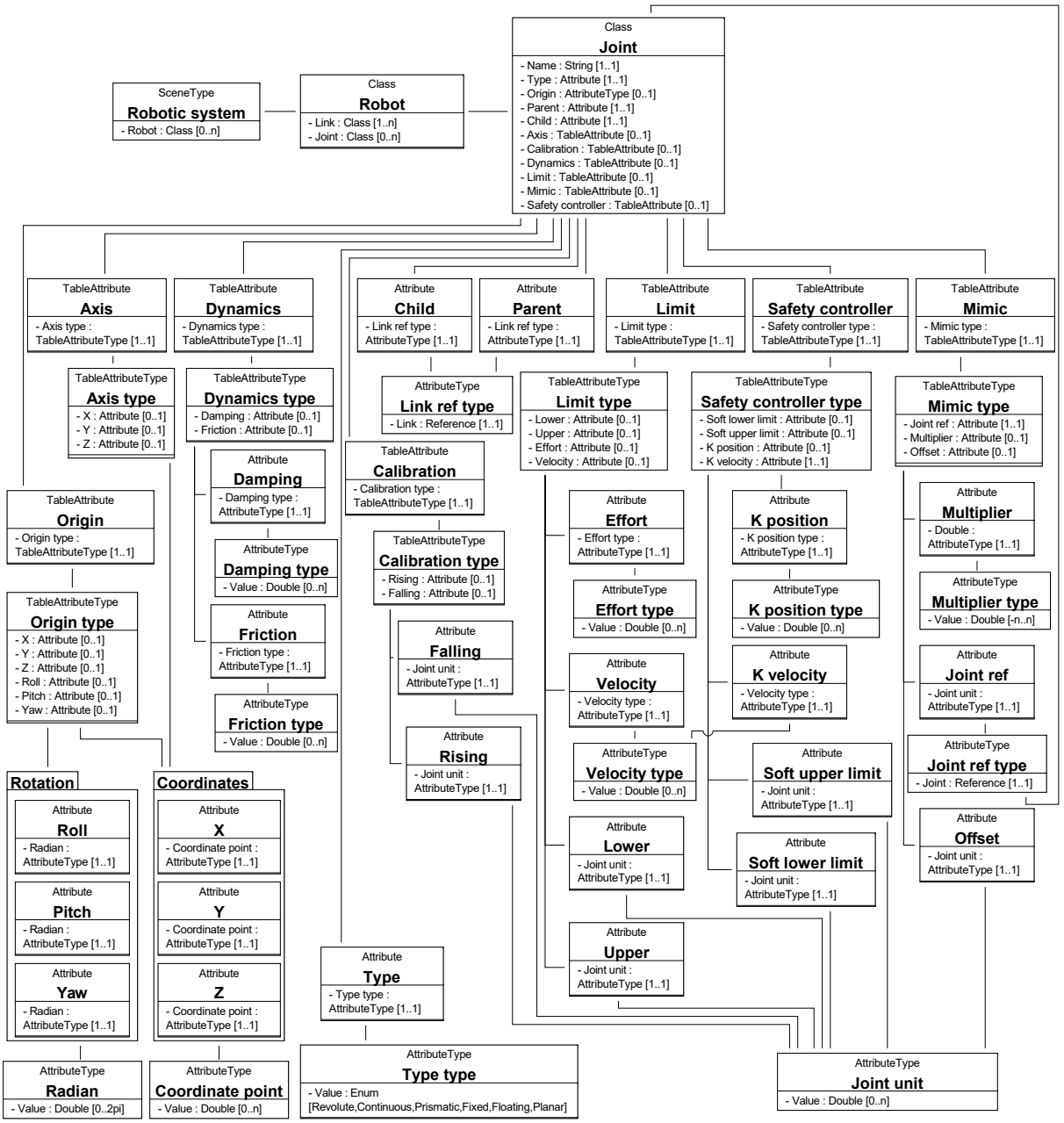


Figure 2: The URDF metamodel of the URDF joint. (navigability suppressed for clarity)

3.3. Implementation of the URDF Metamodel on the MM-AR Platform

The URDF metamodel has been implemented on the MM-AR platform. Although all data structures including the access via a REST-API could be realized, the front-end of the platform still misses some UI components required for editing instances of the URDF metamodel. Figure 3 therefore shows a mock-up of the *Visual* attribute of a link as the currently used user interface does not yet support the concept of a table of a table as detailed in Section 3.2. The *Origin* can have multiple rows representing multiple states (i.e., positions) of the visual model.

When loading URDF files into the MM-AR platform, the process as depicted in Figure 4 is initiated. The URDF file is first imported into the MM-AR web client tool and converted to an instance of a SceneType with the URDF metamodel. In this way, the concepts of the URDF file can be directly accessed from other scene instances on the platform. The 3D view can be an AR scene or a VR environment. Users then interact with the controller models that allow or restrict the interaction with the 3D model.

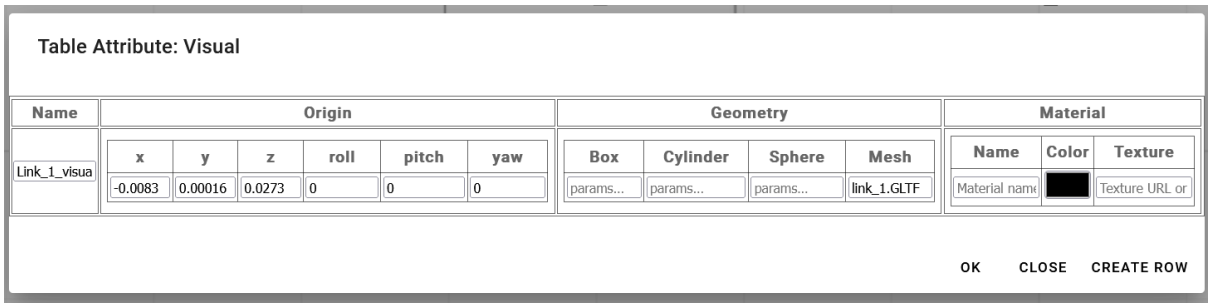


Figure 3: Mock-Up of the link's visual attribute in MM-AR

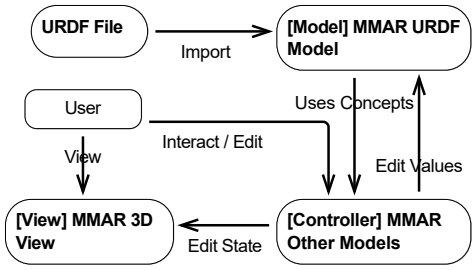


Figure 4: The URDF metamodel process overview

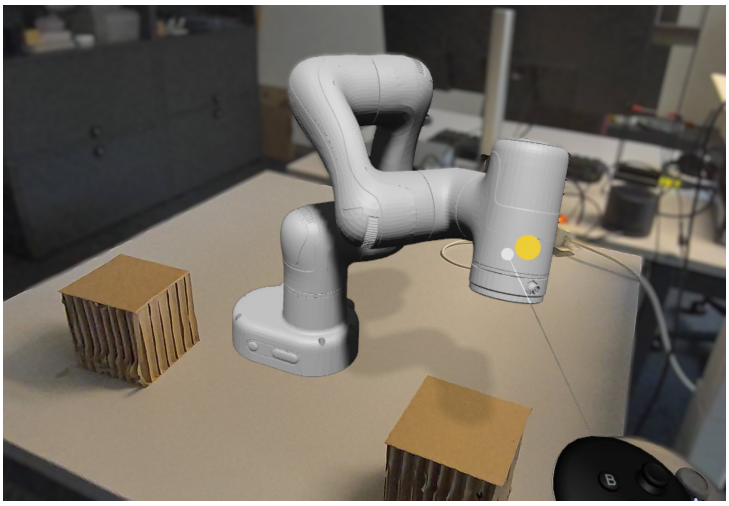


Figure 5: Mock-Up 3D view of the URDF model representing the 6-DoF robotic arm

4. Example: Fixed-Robot Pick-and-Place

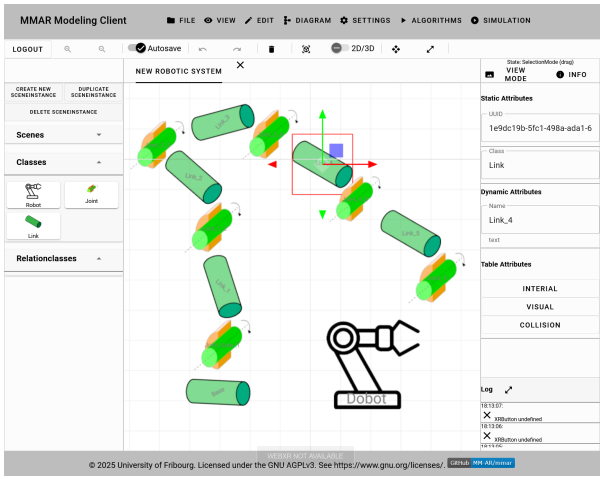
In this section, we illustrate the concepts introduced above through a pick-and-place scenario. Pick-and-place is among the most common tasks in robotics [27] and can be generalized across a variety of contexts. In our example, a box is picked from an initial location and placed, after a prescribed rotation, at a target location.

Figure 6 shows the URDF model obtained by converting the original URDF file so that it matches the URDF metamodel defined in the previous sections. The model represents the 6-DoF robotic arm Dobot Magician E6[2]. It consists of a base and six links connected by six joints. The BPMN model in Figure 7 captures the pick-and-place process and is linked to the URDF model in Figure 6. The selected activity can be visualized in augmented reality (AR), as illustrated in Figure 5. The 3D view can be aligned spatially to the robot's physical position, yielding a ghost-like digital overlay of the real robot.

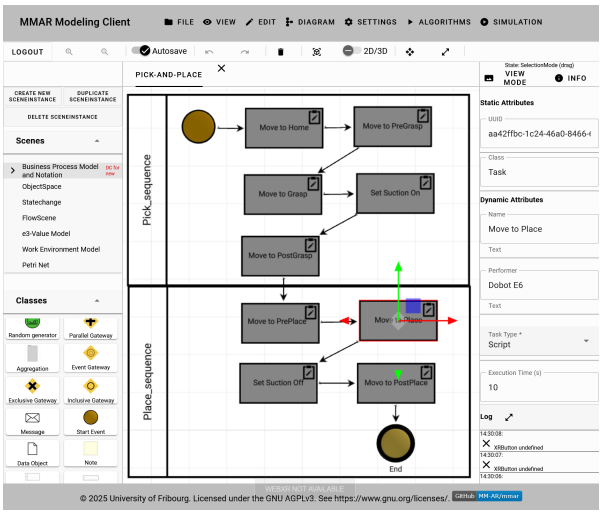
5. Discussion

The URDF schema could be successfully implemented on the MM-AR metamodeling platform. Although the current user interface implementation of the MM-AR web client does not yet fully support all necessary UI components - e.g., the above-mentioned tables-in-tables representations, we can assess that the MM-AR meta[2] model provides all concepts required for the implementation of URDF. Further, although MM-AR natively supports the three-dimensional representation of objects - currently in GLTF format only - and URDF is format-agnostic with regard to 3D formats, the sample robot files used so far

[2]https://www.dobot-robots.com/products/education/magician-e6.html



Current visualization of the URDF model
Figure 6: of the 6-DoF arm Dobot Magician E6 in the MM-AR platform



A BPMN implementation of the pick-and-
Figure 7: place using the URDF model in the MM-AR platform

were only available in STL format. Therefore, the next step will be to either conduct conversions into GLTF format if possible, or to extend the implementation of MM-AR to also support the STL format.

Robotic tasks inherently unfold in 3D environments, which are often difficult to capture faithfully with traditional conceptual modeling notations.

In our approach, the 3D environment serves as a first-class representation of the digital twin's current state. Conventional conceptual modeling approaches typically constrain digital twin representations to two-dimensional diagrams or isolated 3D visualizations with limited semantics. By integrating a URDF model, we enhance the ability to represent digital twins in 3D while enriching them with explicit conceptual semantics, e.g. for specifying behavior in relation to the URDF components.

URDF benefits from a mature ecosystem of algorithms and simulation tools. Our metamodel implementation enables these capabilities to be integrated into the MM-AR platform, thereby supporting simulation and analysis scenarios. For example, inverse kinematics can be applied to assess reachability between two poses, and collision checking can verify that no interferences occur for a given task model. These extensions benefit both communities: robotics, which contributes established analyses and simulations, and conceptual enterprise modeling, which provides rigorous domain-specific languages and tooling. In doing so, the approach helps bridge the gap between conceptual modeling and knowledge-based digital twins.

The current metamodel does not yet include proposed URDF extensions such as sensors or end effectors. Nevertheless, its structure permits the addition of non-robotic artifacts and attributes without compromising interoperability with other models or the integration mechanisms.

6. Conclusion & Outlook

In this paper, we proposed a URDF metamodel that integrates with other modeling languages based on the MM-AR meta^2 model. The metamodel is directly mapped from the URDF XML specification, enabling a faithful representation of robotic architectures composed of links and joints. Each element can be associated with properties such as visual and geometry descriptions, mass and inertia, and joint limits. Mapping URDF concepts to a metamodel allows the instantiation of models representing diverse robotic systems; these instances can then be referenced by other (meta)models to support broader modeling and analysis scenarios. By adopting the URDF metamodel, we reduce the effort required to represent and edit 3D robotic tasks and leverage the existing ecosystem of URDF-based algorithms and analyses. Designing and analyzing robotic tasks digitally could not only accelerate prototyping and

development but may also reduce costs compared to testing on physical machinery. We can answer the research questions as follows. (RQ1) The metamodel provides first-class integration with enterprise and process models via typed cross-model references. The inherent spatial anchoring (locations, transforms, and 3D representations) can be consumed in AR/VR. The validation and analysis hooks (e.g., unit constraints, joint-limit consistency, pose reachability) are defined at the modeling level, as well as the queryability and versioning of structure and state beyond file-level XML. And crucially, it maintains interoperability with the URDF ecosystem, while enabling conceptual links to behavior models that raw URDF files do not support. (RQ2) For fixed industrial robots and digital-twin support, the minimal set realized in our metamodel is:

- *Link* with *Inertial* (Origin, Inertia, Mass), *Visual* (Origin, Geometry, Material), and *Collision* (Origin, Geometry).
- *Joint* (a relation between Links) with *type* (revolute, continuous, prismatic, fixed, floating, planar), *Origin*, *Axis*, *Limit*, *Dynamics*, *Calibration*, *Safety Controller*, and *Mimic*, plus a scene-level root link and coordinate frames.

These cover kinematic topology, core dynamics, geometry/meshes, and operational bounds, which we found sufficient to import an industrial 6-DoF arm, render it in 3D, attach time-varying poses, and run typical analyses (e.g., reachability/collision) used in digital twins. This implementation aligns with the elements defined by the URDF XML specification. Elements like SRDF groupings or sensors are useful extensions but not required for faithful fixed-robot representation.

(RQ3) We embed spatial conceptual modeling by leveraging MM-AR's native spatial attributes so that every URDF element has a pose and 3D representation. We store pose rows in AttributeTables (e.g., Origins) so the same robot structure can express different runtime states over time along a process. We created typed relations between process activities (e.g., BPMN tasks) and URDF elements (links, joints, target poses). This allows process models to constrain or derive spatial states (paths, keep-out zones, approach poses) and to drive AR overlays, thereby aligning behavioral steps with the robot's structural and physical semantics.

As next steps, we will implement a working prototype that combines the URDF (meta)model with a process model to specify and execute robotic tasks in a 3D environment. We plan to include concepts from SRDF and Xacro to have an even more complete robotic modeling metamodel.

Acknowledgments

Financial support is gratefully acknowledged by the SmartLiving Lab (https://www.smartlivinglab.ch/en/) funded by the University of Fribourg, EPFL, and HEIA-FR.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] M. Gouasmi, M. Ouali, B. Fernini, M. Meghatria, Kinematic Modelling and Simulation of a 2-R Robot Using SolidWorks and Verification by MATLAB/Simulink, International Journal of Advanced Robotic Systems 9 (2012) 245. URL: https://doi.org/10.5772/50203. doi:10.5772/50203, publisher: SAGE Publications.

[2] K. K. Pandey, T. Shah, S. Yadrave, S. Shinde, V. Pagare, Design and Development of an Autonomous Robot Assistant, in: B. B. V. L. Deepak, M. V. A. R. Bahubalendruni, D. R. K. Parhi, B. B. Biswal (Eds.), Intelligent Manufacturing Systems in Industry 4.0, Springer Nature, Singapore, 2023, pp. 381–389. doi:10.1007/978-981-99-1665-8_34.

[3] D. Tola, P. Corke, Understanding URDF: A Dataset and Analysis, IEEE Robotics and Automation Letters 9 (2024) 4479–4486. URL: https://ieeexplore.ieee.org/abstract/document/10478618. doi:10.1109/LRA.2024.3381482.

[4] D. Tola, P. Corke, Understanding URDF: A Survey Based on User Experience, in: 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE), 2023, pp. 1–7. URL: https://ieeexplore.ieee.org/abstract/document/10260660. doi:10.1109/CASE56687.2023.10260660, iSSN: 2161-8089.

[5] D. Borcard, H.-G. Fill, State of the Art and Research Directions for Visual Conceptual Modeling in Robotics, in: R. Guizzardi, L. Pufahl, A. Sturm, H. van der Aa (Eds.), Enterprise, Business-Process and Information Systems Modeling, Springer Nature Switzerland, Cham, 2025, pp. 415–430. doi:10.1007/978-3-031-95397-2_26.

[6] K. Bourr, F. Corradini, S. Pettinari, B. Re, L. Rossi, F. Tiezzi, Disciplined use of BPMN for mission modeling of Multi-Robot Systems, in: Proceedings, volume 1613, 2021, p. 0073. URL: https://ceur-ws.org/Vol-3045/paper01.pdf.

[7] F. Corradini, S. Pettinari, B. Re, L. Rossi, F. Tiezzi, A BPMN-driven framework for Multi-Robot System development, Robotics and Autonomous Systems 160 (2023) 104322. URL: https://www.sciencedirect.com/science/article/pii/S0921889022002111. doi:10.1016/j.robot.2022.104322.

[8] C. Azevedo, A. Matos, P. U. Lima, J. Avendaño, Petri Net Toolbox for Multi-Robot Planning under Uncertainty, Applied Sciences 11 (2021) 12087. URL: https://www.mdpi.com/2076-3417/11/24/12087. doi:10.3390/app112412087, publisher: Multidisciplinary Digital Publishing Institute.

[9] M. Shin, S. Jung, A Survey of Behavior Tree-Based Task Planning Algorithms for Autonomous Robotic Systems, in: 2024 15th International Conference on Information and Communication Technology Convergence (ICTC), 2024, pp. 2039–2041. URL: https://ieeexplore.ieee.org/abstract/document/10827191. doi:10.1109/ICTC62082.2024.10827191, iSSN: 2162-1241.

[10] H.-G. Fill, Spatial Conceptual Modeling: Anchoring Knowledge in the Real World, in: H.-G. Fill, H. Kühn (Eds.), Metamodeling: Applications and Trajectories to the Future: Essays in Honor of Dimitris Karagiannis, Springer Nature Switzerland, Cham, 2024, pp. 35–50. URL: https://doi.org/10.1007/978-3-031-56862-6_3. doi:10.1007/978-3-031-56862-6_3.

[11] T. B. Asafa, T. M. Afonja, E. A. Olaniyan, H. O. Alade, Development of a vacuum cleaner robot, Alexandria Engineering Journal 57 (2018) 2911–2920. URL: https://www.sciencedirect.com/science/article/pii/S1110016818300899. doi:10.1016/j.aej.2018.07.005.

[12] M. Bartoš, V. Bulej, M. Bohušík, J. Stanček, V. Ivanov, P. Macek, An overview of robot applications in automotive industry, Transportation Research Procedia 55 (2021) 837–844. URL: https://www.sciencedirect.com/science/article/pii/S2352146521004543. doi:10.1016/j.trpro.2021.07.052.

[13] International Organization for Standardization (ISO), ISO 8373:2021 - Robotics — Vocabulary, 2021. URL: https://www.iso.org/standard/75539.html.

[14] K. J. Waldron, J. Schmiedeler, Kinematics, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer International Publishing, Cham, 2016, pp. 11–36. URL: https://doi.org/10.1007/978-3-319-32552-1_2. doi:10.1007/978-3-319-32552-1_2.

[15] N. Koenig, A. Howard, Design and use paradigms for Gazebo, an open-source multi-robot simulator, in: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), volume 3, 2004, pp. 2149–2154 vol.3. URL: https://ieeexplore.ieee.org/abstract/document/1389727. doi:10.1109/IROS.2004.1389727.

[16] H. R. Kam, S.-H. Lee, T. Park, C.-H. Kim, RViz: a toolkit for real domain data visualization, Telecommunication Systems 60 (2015) 337–345. URL: https://doi.org/10.1007/s11235-015-0034-5. doi:10.1007/s11235-015-0034-5.

[17] M. Singh, E. Fuenmayor, E. P. Hinchy, Y. Qiao, N. Murray, D. Devine, Digital Twin: Origin to Future, Applied System Innovation 4 (2021) 36. URL: https://www.mdpi.com/2571-5577/4/2/36. doi:10.3390/asi4020036, publisher: Multidisciplinary Digital Publishing Institute.

[18] F. Tao, B. Xiao, Q. Qi, J. Cheng, P. Ji, Digital twin modeling, Journal of Manufacturing Systems

64 (2022) 372–389. URL: https://www.sciencedirect.com/science/article/pii/S0278612522001108. doi:10.1016/j.jmsy.2022.06.015.

[19] A. Mazumder, M. F. Sahed, Z. Tasneem, P. Das, F. R. Badal, M. F. Ali, M. H. Ahamed, S. H. Abhi, S. K. Sarker, S. K. Das, M. M. Hasan, M. M. Islam, M. R. Islam, Towards next generation digital twin in robotics: Trends, scopes, challenges, and future, Heliyon 9 (2023). URL: https://www.cell.com/heliyon/abstract/S2405-8440(23)00566-2. doi:10.1016/j.heliyon.2023.e13359, publisher: Elsevier.

[20] F. Muff, H.-G. Fill, A Domain-Specific Visual Modeling Language for Augmented Reality Applications Using WebXR, in: J. P. A. Almeida, J. Borbinha, G. Guizzardi, S. Link, J. Zdravkovic (Eds.), Conceptual Modeling, volume 14320, Springer Nature Switzerland, Cham, 2023, pp. 334–353. URL: https://link.springer.com/10.1007/978-3-031-47262-6_18. doi:10.1007/978-3-031-47262-6_18, series Title: Lecture Notes in Computer Science.

[21] F. Muff, H.-G. Fill, M2AR: A Web-based Modeling Environment for the Augmented Reality Workflow Modeling Language, in: Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, MODELS Companion '24, Association for Computing Machinery, New York, NY, USA, 2024, pp. 1–5. URL: https://dl.acm.org/doi/10.1145/3652620.3687779. doi:10.1145/3652620.3687779.

[22] F. Muff, H.-G. Fill, Multi-faceted Evaluation of Modeling Languages for Augmented Reality Applications The Case of ARWFML, in: W. Maass, H. Han, H. Yasar, N. Multari (Eds.), Conceptual Modeling, Springer Nature Switzerland, Cham, 2025, pp. 75–93. doi:10.1007/978-3-031-75872-0_5.

[23] Y. Kang, D. Kim, K. Kim, URDF Generator for Manipulator Robot, in: 2019 Third IEEE International Conference on Robotic Computing (IRC), 2019, pp. 483–487. URL: https://ieeexplore.ieee.org/abstract/document/8675569. doi:10.1109/IRC.2019.00101.

[24] M. Feder, A. Giusti, R. Vidoni, An approach for automatic generation of the URDF file of modular robots from modules designed using SolidWorks, Procedia Computer Science 200 (2022) 858–864. URL: https://www.sciencedirect.com/science/article/pii/S1877050922002927. doi:10.1016/j.procs.2022.01.283.

[25] S. Schneider, N. Hochgeschwender, G. K. Kraetzschmar, Structured Design and Development of Domain-Specific Languages in Robotics, in: D. Brugali, J. F. Broenink, T. Kroeger, B. A. MacDonald (Eds.), Simulation, Modeling, and Programming for Autonomous Robots, Springer International Publishing, Cham, 2014, pp. 231–242. doi:10.1007/978-3-319-11900-7_20.

[26] R. Ringe, R. Nolte, N. Zargham, R. Porzel, R. Malaka, METAMORPH - A Metamodeling Approach for Robot Morphology, in: 2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2025, pp. 627–636. URL: https://ieeexplore.ieee.org/abstract/document/10973806. doi:10.1109/HRI61500.2025.10973806.

[27] J. A. Haustein, K. Hang, J. Stork, D. Kragic, Object Placement Planning and optimization for Robot Manipulators, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Macau, China, 2019, pp. 7417–7424. URL: https://ieeexplore.ieee.org/document/8967732/. doi:10.1109/IROS40897.2019.8967732.