Describing Robots from Design to Learning: Towards an Interactive Lifecycle Representation of Robots

Nuofan Qiu[1], Fang Wan[2,*], and Chaoyang Song[3,*]

***Abstract*— The robot development process is divided into several stages, which create barriers to the exchange of information between these different stages. We advocate for an interactive lifecycle representation, extending from robot morphology design to learning, and introduce the role of robot description formats in facilitating information transfer throughout this pipeline. We analyzed the relationship between design and simulation, enabling us to employ robot process automation methods for transferring information from the design phase to the learning phase in simulation. As part of this effort, we have developed an open-source plugin called ACDC4Robot for Fusion 360, which automates this process and transforms CAD software into a user-friendly graphical interface for creating and editing robot description formats to solve the Design2Sim problem. Additionally, we offer an out-of-the-box robot model library to streamline and reduce repetitive tasks. All codes are hosted open-source. (`https://github.com/bionicdl-sustech/ACDC4Robot`)**

I. Introduction

Robot development, as a systematical process, encompasses several distinct phases, commencing with the design of the robot's morphology. The information about robot morphology design plays a pivotal role throughout the development lifecycle, particularly in facilitating learning through simulation. A robot's morphology critically influences its configuration space, which in turn determines the robot's functional capabilities [1]. This morphology is primarily established during the design phase. The advancement in computer-aided design (CAD) technologies has been instrumental in streamlining the design process and enhancing interactivity through modern graphical user interfaces (GUI).

In addition to the consideration of robot morphology, the aspect of learning has emerged as a crucial focus in robotics. This is attributed to its role in enabling robots to perform complex tasks, enhancing their interaction with the environment. However, training robots directly in hardware can result in failures or damage, which can be costly and time-consuming. In contrast, simulation offers a more economical and safer approach to robot development. Moreover, robot simulators incorporate domain randomization techniques that increase the exploration of the state-action space, facilitating the transfer of knowledge learned in simulation to real robots [2]. Consequently, most contemporary learning methodologies incorporate simulation to expedite the learning process.

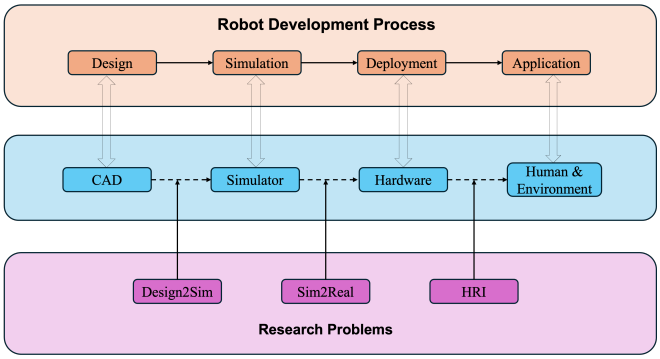

Fig. 1. **Robot Development Process and Research Problems between them.** The development process of robotic systems can be segmented into multiple distinct phases. Discrepancies and transitional challenges between these phases often result in significant issues in the overall development of robots. Consequently, this has led to the emergence of specific research areas, including 'Design-to-Simulation' (Design2Sim), 'Simulation-to-Reality' (Sim2Real), and Human-Robot Interaction (HRI).

For effective simulation of a robot system, comprehensive models of both the robot and its operating environment are indispensable. These models are typically delineated using the Robot Description Format (RDF). RDF represents a category of formats designed to structurally depict robot systems, adhering to a predefined set of rules. It encapsulates critical details of the robot system, encompassing aspects such as kinematics, dynamics, actuators, and sensors. Moreover, it extends to incorporate environmental information, which is vital for the robot's interactive capabilities. Given the comprehensive nature of the data encompassed by RDF, it is a pivotal interface that bridges the gap between robot morphology design and facilitating robot learning within simulations. However, the robot description formats are tedious to generate for simulation [3].

A. File Formats from Design to Learning

In a simulation environment, several file formats are used in robot morphology design and learning. These file formats have specific features tailored to different application scenarios, hindering process interoperability. Various file formats make it challenging to transfer information from the design phase to the learning process in simulation.

This work was partly supported by the Ministry of Education of China-Autodesk Joint Project on Engineering Education, the National Natural Science Foundation of China [62206119, 52335003], and the Science, Technology, and Innovation Commission of Shenzhen Municipality [JCYJ20220818100417038, ZDSYS20220527171403009]. Corresponding Emails: wanf@sustech.edu.cn (F.W.), songcy@ieee.org (C.S.).

[1]Nuofan Qiu is with the School of Design, Southern University of Science and Technology, Shenzhen 518055, China

[2]Fang Wan is with Shenzhen Key Laboratory of Intelligent Robotics and Flexible Manufacturing, Southern University of Science and Technology, Shenzhen, Guangdong 518055, China.

[3]Chaoyang Song is with the Department of Mechanical and Energy Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

In contemporary practice, robot morphology is typically designed using CAD software. File formats in the CAD field can be categorized into neutral and native formats. Neutral file formats adhere to cross-platform compatibility standards, including STEP files (.stp, .step), IGES files (.igs, .ige), COLLADA, and STL. Native file formats are platform-specific and contain precise information optimized for the respective platform, examples of which include SolidWorks (.sldprt, .sldasm), Fusion 360 (.f3d), Blender (.blend), and many others.

Several robot description formats are used in robot simulation. The most common format is the Unified Robotics Description Format (URDF), which is supported by various robot simulators, including PyBullet, Gazebo, and MuJoCo. SDFormat is natively supported by Gazebo and partially supported by PyBullet. MuJoCo natively supports MJCF and is also supported by Isaac Sim and PyBullet. Other robot description formats resemble native formats specific to particular simulators than URDF. For example, the CoppeliaSim file is designed for use with CoppeliaSim, and WBT is used in Webots.

B. A Brief Historical Review of Robot Description Formats

Robot Description Formats provide information for modeling the robot system and are used in robot simulators as input. Currently, research resources on robot description formats are limited, with most of the relevant information available only on their respective websites and forums, making research challenging. The authors in [4] compared existing formats and summarized their main advantages and limitations. Here, we offer a concise historical perspective on robot description formats to enhance understanding.

1) Before Unified Robot Description Format (URDF): Research on robot modeling predates the concept of a robot description format by a considerable margin. Denavit and Hartenberg formulated a convention using four parameters to model robot manipulators in 1955 [5], which is still widely used in robotics. With the advent of computer simulation, robots can be defined using programming languages with variables [6]. While it is theoretically possible to describe a robotic system through a programming language's variables and data structures, relying on programming language features can make exchanging robot system information across different platforms cumbersome for various purposes. Therefore, representing robot system information in a unified, programming language-independent manner will facilitate interchangeability across different platforms and enhance development efficiency. Park et al. [7] discussed XML-based formats, which can describe robots due to XML's convenience in delivering information.

2) URDF, SDFormat, and Others: While developing a personal robotics platform, the idea of creating a "Linux for robotics" came to the minds of Eric Berger and Keenan Wyrobek [8]. With the first distribution of ROS released in 2009, URDF was simultaneously introduced. URDF is an XML-based file format that enhances readability and describes robot links' information, including kinematics, dynamics, geometries, and robot joints' information organized in a tree structure. URDF universally models robots, making them suitable for visualization, simulation, and planning within the ROS framework.

With the growing popularity of ROS, URDF has become a widely used robot description format supported by various simulation platforms, such as PyBullet, MuJoCo, and Isaac Sim, among others. However, an increasing number of roboticists have recognized the limitations and issues of URDF, such as its inability to support closed-loop chains. To address these concerns, the community has endorsed proposals like URDF2[1]. The problems stemming from URDF's design may become increasingly challenging to resolve over time due to the diminishing activity in its development (the repository's[2] update frequency has become very low). Therefore, new formats can draw upon URDF's experience to avoid such issues from the outset and expand their ability to describe a broader range of scenarios.

Rosen Diankov et al. [9] promoted an XML-based open standard called COLLADA, which allows for complex kinematics with closed-loop chains. SDFormat (Simulation Description Format) was initially developed as part of the Gazebo simulator and separated from Gazebo as an independent project to enhance versatility across different platforms. SDFormat is also an XML-based format that shares a similar grammar with URDF but extends its ability to describe the environment with which the robot interacts. Furthermore, SDFormat is actively developing, making it more responsive to future robotics needs. MJCF is another XML-based file format initially used in the MuJoCo simulator. It can describe robot structures, including kinematics, dynamics, and other elements like sensors and motors.

Although these robot description formats enable more comprehensive modeling information for robotic systems and have resolved some of the limitations of URDF, URDF remains the most universally adopted robot description format in academia and industry. Fig. 2 provides a timeline representation of the release times of these robot description formats.

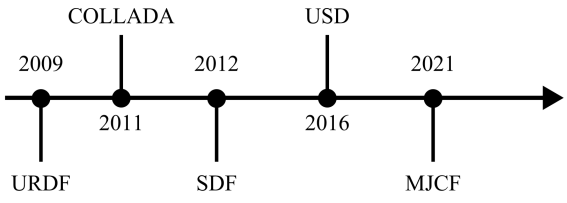

Fig. 2. **Robot Description Format History**: Timeline of the release times for each robot description format.

3) Beyond URDF: Daniella Tola et al. [10], [11] surveyed the user experience of URDF within the robotics community, including academia and industry. Their survey

[1]https://sachinchitta.github.io/urdf2/
[2]https://github.com/ros/urdf

TABLE I
COMPARING CAD TO RDF TOOLS.

Tool	CAD software	Output RDF
sw_urdf_exporter	SolidWorks	URDF
RobotDescriptor	FreeCAD	SDF
OnShape to Robot	OnShape	URDF, SDF
Phobos	Blender	URDF, SDF, SMURF
Robot Editor	Blender	SDF
fusion2urdf	Fusion 360	URDF
ACDC4Robot	Fusion 360	URDF, SDF, MJCF

revealed problems associated with using URDF and inspired the research of robot description formats. Some challenges are specific to URDF, for instance, the lack of support for a closed-chain mechanism. Additionally, some challenges are common to other robot description formats, such as the complex workflow involving multiple tools, including CAD software, text editors, and simulators.

One of the solutions is to create a new robot description format that can adequately describe robot systems and is also easy to use. A new attempt in this regard is the OpenUSD format[3], which combines the strengths of academia and industry to drive progress in this field.

Another solution is to provide more tools to enhance the usability of robot description formats. Some tools, such as `gz-usd`[4] and `sdformat_mjcf`[5], improve the interoperability of different robot description formats. Software tools such as "Nuts-and-Bolts" [3] will make the simulation more useful in robotics. In table I, we list several tools that can be used for exporting robot designs to robot description formats.

In the rest of this paper, Section II introduces methods for structuring the workflow from design to learning and presents an automation tool, ACDC4Robot, designed to address these challenges. Section III demonstrates the usage of ACDC4Robot with examples and offers a robot model library for users that can be readily utilized. We conclude in Section IV and discuss the limitations of our work and the future of the format for robot system development. This article's contributions include promoting a lifecycle representation from robot design to robot learning, offering the ACDC4Robot tool within Fusion 360 to streamline the workflow from robot design to robot learning, and constructing an out-of-the-box robot model library for robot design and learning.

II. METHODOLOGY

We analyze the workflow to describe robots from design to learning, then describe an interactive lifecycle representation. Next, we employ robot process automation to streamline the processes of robot design to robot learning. An automation tool integrated with a CAD platform can achieve this lifecycle representation interactively.

A. An Interactive Lifecycle Representation

Information about robot morphology design is crucial for subsequent phases in the robot development process, particularly for simulation. RDF serves as an effective medium to bridge the gap between CAD systems and simulators. However, manual creation or modification of RDFs is often tedious, time-consuming, and error-prone. Modern CAD software, equipped with Graphical User Interfaces (GUI), alleviates this by rendering the design process more interactive. Leveraging CAD as an editor for RDF allows for a "What You See Is What You Get" (WYSIWYG) approach, facilitating intuitive interactions with the robot description format. Consequently, this integration transforms the robot model into an interactive representation throughout the development lifecycle.

B. Robotic Process Automation from Design to Simulation

CAD software and robot simulators are two systems with distinct functions, each emphasizing different aspects of the robot. However, some features in CAD and robot simulators represent different forms of the same information. The way components are joined to construct a robot assembly in CAD software determines the kinematics of the robot. The physical properties of robot components in CAD software can also pertain to the dynamics in the robot simulator. The geometric shape of components can be utilized for visualization and collision information in the simulator. Fig. 3 shows that a one-to-one relationship between CAD and simulation systems enables the realization of automated conversions between these two processes, which was previously feasible.

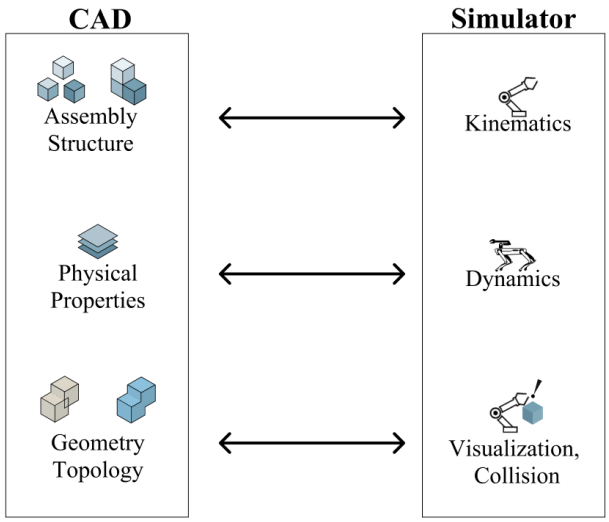

Fig. 3. **Robot Process Automation from Design to Simulation.** A one-to-one relationship exists between CAD software and robot simulators in the following features: (i) assembly structure and kinematics; (ii) physical properties and dynamics; (iii) geometry topology, visualization, and collision.

C. An Open-source Plug-in Using Fusion 360

Software tools can be useful to remove the barriers for using simulation in robotics, thus, we present an open-source plugin using Fusion 360 to achieve the interactive lifecycle

[3]https://aousd.org/
[4]https://github.com/gazebosim/gz-mujoco/tree/main/sdformat_mjcf
[5]https://github.com/gazebosim/gz-usd

process automation from robot design to robot learning. Fusion 360 is a popular CAD software developed by Autodesk within the roboticist community. It provides API access for developers, allowing it to accomplish automation tasks.

Following J. Collins et al.'s work [12], we selected a set of popular simulators used in robotics learning, including RaiSim, Gazebo, Nvidia Isaac, MuJoCo, PyBullet, CARLA, Webots, and CoppeliaSim for comparing the compatibility of robot description formats: URDF, SDFormat, MJCF, and USD. Since we have opted to utilize Gazebo, PyBullet, and MuJoCo as our target simulation platforms, we have decided to use URDF, SDFormat, and MJCF according to Table II as the robot description formats for transforming the design into the learning process.

TABLE II
COMPARING SIMULATOR SUPPORT LEVELS WITH DIFFERENT ROBOT DESCRIPTION FORMATS.

Robot Description Format	Year	Supported Simulators
URDF	2009	Gazebo, Nvidia Isaac, MuJoCo, PyBullet, CoppeliaSim
SDFormat	2012	Gazebo, PyBullet
USD	2016	Nvidia Isaac
MJCF	2021	RaiSim, Nvidia Isaac, MuJoCo, PyBullet

III. RESULTS

In this section, we will introduce the GUI of the Fusion 360 plugin that enables the interactive lifecycle process from robot design to robot simulation, along with a guide on how to use the plugin. And demonstrate the ability of ACDC4Robot by examples. Additionally, we will show a robot library containing various out-of-the-box robot models to help users reduce the time spent on repetitive tasks.

A. The ACDC4Robot Plug-in with Fusion 360

ACDC4Robot is an open-source plugin for Fusion 360 that can automatically convert design information into a simulation data structure (robot description format) for learning. The pipeline for using the ACDC4Robot plugin from design to learning is illustrated in Fig. 4. Users can import an existing robot model into Fusion 360 or design a robot from scratch using Fusion 360.

ACDC4Robot provides a straightforward GUI to simplify the conversion process, as shown in Fig. 5. After completing the robot morphology design, click the start button of the ACDC4Robot plugin, and a settings panel will appear on the right side of Fusion 360. This panel allows the choice between URDF, SDFormat, or MJCF as the format for transferring design information to the simulation and selecting the target simulator for exporting in a format compatible with the chosen simulator. The exported files can then be used in the simulation for robot learning.

Compared to the traditional method of creating and editing robot description formats using text editors, Fusion 360, a graphical interface to modify robot design, where modifications are directly reflected on the robot description file, is a more intuitive approach. Users can click a few buttons to generate robot descriptions for learning in simulation using ACDC4Robot, freeing them from the previous tedious workflow.

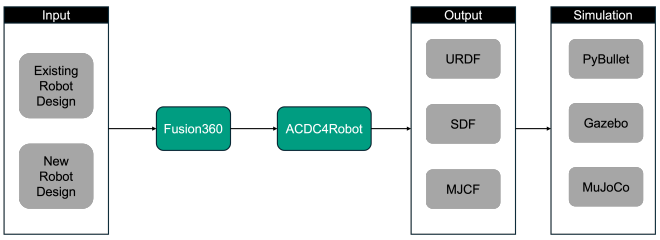

Fig. 4. **Transfer robot model from design to learning simulation using ACDC4Robot.** (i) Step 1: Import an existing robot model or create a new robot model from scratch to Fusion 360; (ii) Step 2: ACDC4Robot plug-in converts robot design to URDF, SDF, or MJCF according to aimed simulators; (iii) Step 3: Import the robot description format into the simulator for learning.

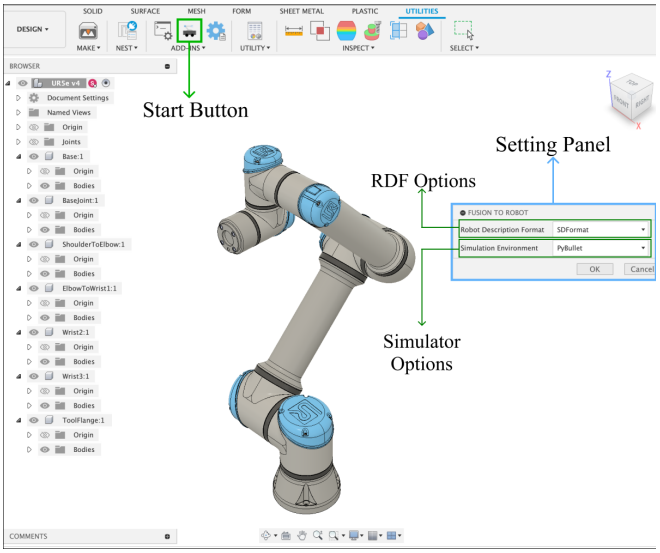

Fig. 5. **ACDC4Robot Plugin GUI.** Clicking the ACDC4Robot start button will open the settings panel, allowing the user to select the target robot description format and the target simulation environment.

B. Demonstrations with Serial Chain and Closed Chain

It is prevalent to use existing robot models for learning purposes. Here, we use the UR5e robot manipulator model downloaded from UR's website to demonstrate the design-to-learning process with a serial chain robot, representing a typical robot type. Fig. 6 shows the process of using ACDC4Robot for design to learning.

Closed-chain mechanisms are also widely used in robots. We demonstrate creating a four-bar linkage from scratch to simulation in Fig. 7.

C. Design to Learning using ACDC4Robot

In our laboratory, we utilize the ACDC4Robot framework to facilitate the transfer of robot models from the design phase to the learning phase in simulations. Fig 8 illustrates

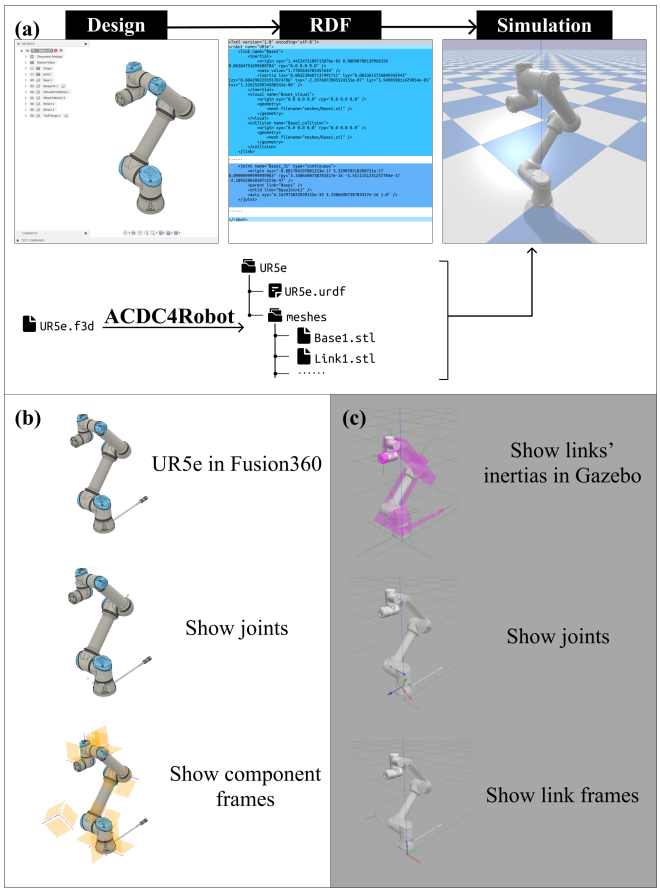

Fig. 6. **Serial chain robot example using ACDC4Robot.** (a) This figure shows the same UR5e robot manipulator model in three different representations: (i) Fusion 360 design model; (ii) XML-based robot description format; (iii) PyBullet simulation environment for learning. The Fusion 360 design file `UR5e.f3d` is converted by ACDC4Robot into a `UR5e` folder that contains `UR5e.urdf` and mesh files, which are then loaded into the PyBullet simulation environment. (b) UR5e manipulator model and its joints and frames in Fusion 360. (c) UR5e manipulator model and its joints, link frames that come from ACDC4Robot conversion in Gazebo

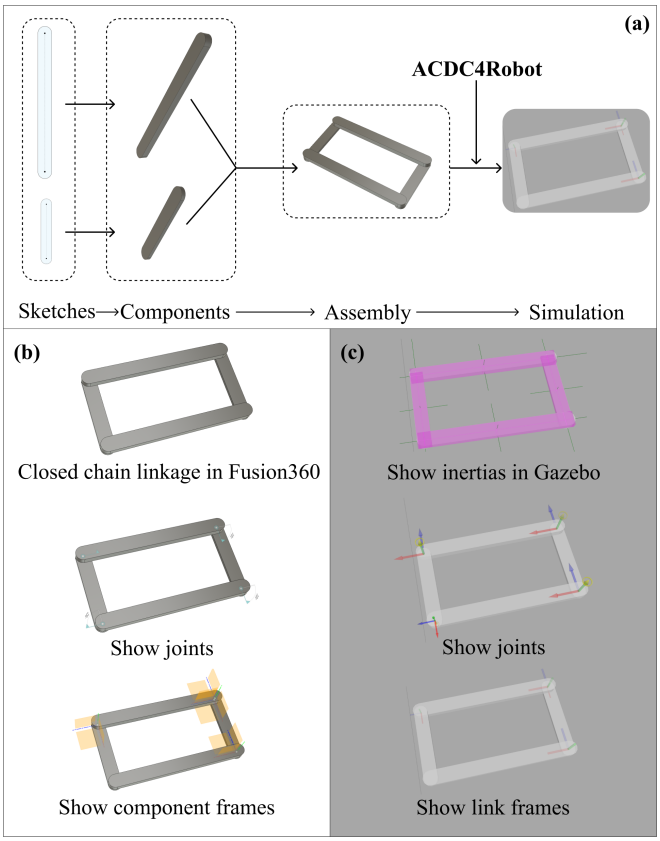

Fig. 7. **Closed loop robot example using ACDC4Robot.** (a). Create a closed-chain four-bar linkage from scratch in Fusion 360 for simulation in Gazebo; (b). Closed-chain four-bar linkage model and its joints component frames in Fusion 360; (c). Closed-chain four-bar linkage model and its joints, component frames from ACDC4Robot conversion in Gazebo.

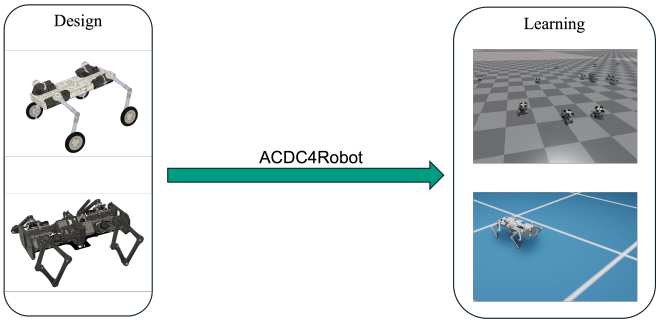

Fig. 8. **Examples of Using ACDC4Robot for Design to Learning.**

the designs of both a wheeled quadruped robot and a Bennett quadruped robot created in Fusion 360. Through the application of ACDC4Robot, researchers can seamlessly integrate these robot design models into the Isaac Sim learning environment. This process streamlines the transition from conceptual design to practical, simulation-based learning applications.

D. Robot Library of the ACDC4Robot Plugin

By enabling the automatic transfer of robot design information to learning in simulation through the ACDC4Robot plugin, the robot design model becomes metadata in the interactive lifecycle representation of a robot. Consequently, a library of robot design models helps construct robot applications using the interactive lifecycle pipeline.

Based on a survey of robot types modeled in URDF within the roboticist community [10], it was found that robotic arms, mobile robots, end effectors, and dual-arm robots are the most frequently used types of robots. Using this knowledge and investigating several robot datasets, including RoboSuite[6], awesome-robot-descriptions[7], Gazebo models[8], and CoppeliaRobotics models[9], we have created a robot model library [10] for the community. This library, listed in Table III, has been tested with the ACDC4Robot plugin and can be used out of the box. The robot library can be downloaded from the ACDC4Robot repository, and we continue to add content for this library.

[6]https://robosuite.ai/docs/modules/robots.html
[7]https://github.com/robot-descriptions/awesome-robot-descriptions
[8]http://models.gazebosim.org/
[9]https://github.com/CoppeliaRobotics/models
[10]https://github.com/bionicdl-sustech/ACDC4Robot/blob/main/RobotLibrary.md

TABLE III
A ROBOT LIBRARY FOR DESIGN TO LEARNING

Robot Name	Robot Type	Structure
Franka Emika Panda	Robotic Arm	Serial Chain
Franka Emika Hand	End Effector	Serial Chain
Kinova Gen3	Robotic Arm	Serial Chain
Rethink Sawyer	Robotic Arm	Serial Chain
Robotiq 2F85 Gripper	End Effector	Closed Chain
UR5e	Robotic Arm	Serial Chain
ABB YuMi	Dual Arm Robot	Serial Chain
KUKA youBot	Mobile Robot	Serial Chain

IV. DISCUSSIONS AND CONCLUSION

A. Towards a Lifecycle Representation

This article introduces an interactive lifecycle representation that spans from robot design to robot learning in simulation. We identify the gap in transferring design information to the robot simulation environment, leading us to advocate for using a robot description format to bridge the robot morphology design and robot learning in simulation. To facilitate smoother information transfer throughout the process, we have developed a robot process automation tool capable of converting design information into simulation data. This automation is made possible by the one-to-one mapping relationship between design and simulation platforms. As a result, we have created an open-source plugin called ACDC4Robot for Fusion 360. This plugin allows users to convert design information into robot description format, turning Fusion 360 into a graphical user interface (GUI) for interactive robot modeling. This interactive lifecycle process, spanning from robot design to learning, simplifies the development of robot applications.

B. Limitations of the ACDC4Robot Plugin

Although the ACDC4Robot plugin can achieve an interactive lifecycle process from robot design to learning in simulation, it still has some limitations.

The ACDC4Robot plugin is developed using the Fusion 360 API, making it dependent on Fusion 360. Developing a platform-independent tool capable of directly converting a design file into a robot description format for the entire lifecycle process from robot design to learning would be more versatile.

Besides, the ACDC4Robot plugin currently only supports URDF and SDFormat. While SDFormat compensates to some extent for the limitations of URDF, such as modeling closed-chain robots, these two robot description formats can meet most of the needs of academia and industry. However, including support for exporting other robot description formats, such as MJCF, enhances the application of this tool.

Furthermore, the number of robot models in the robot library is still relatively low compared to other robot databases. This is partly due to the challenge of obtaining publicly available robot models. Additionally, assembling these acquired robot models in Fusion 360 and testing them with the ACDC4Robot plugin is quite time-consuming. We plan to incrementally expand the robot library during the future development process.

C. Towards a Unified Robot Lifecycle Format

The modularity of the robot development process has led to separate formats for storing information, creating barriers to data exchange between different stages of development. While conversion tools have partially addressed this issue, they must improve efficiency. An ultimate solution would involve adopting a universal format describing all the information across the robot development lifecycle. The adoption of such a universal format has the potential to enhance the efficiency of robot development significantly. The Universal Scene Description (USD) format is moving in this direction.

REFERENCES

[1] Benedikt Feldotto, Fabrice O Morin, and Alois Knoll. The neurorobotics platform robot designer: modeling morphologies for embodied learning experiments. *Frontiers in Neurorobotics*, 16:856727, 2022.

[2] Fabio Muratore, Fabio Ramos, Greg Turk, Wenhao Yu, Michael Gienger, and Jan Peters. Robot learning from randomized simulations: A review. *Frontiers in Robotics and AI*, page 31, 2022.

[3] HeeSun Choi, Cindy Crump, Christian Duriez, Asher Elmquist, Gregory Hager, David Han, Frank Hearl, Jessica Hodgins, Abhinandan Jain, Frederick Leve, et al. On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proceedings of the National Academy of Sciences*, 118(1):e1907856118, 2021.

[4] Mikhail Ivanou, Stanislav Mikhel, and Sergei Savin. Robot description formats and approaches. In *2021 International Conference" Nonlinearity, Information and Robotics"(NIR)*, pages 1–5. IEEE, 2021.

[5] Jacques Denavit and Richard S Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. 1955.

[6] Leon Žlajpah. Simulation in robotics. *Mathematics and Computers in Simulation*, 79(4):879–897, 2008.

[7] Ji hwan Park, Tae Houn Song, Soon Mook Jung, and Jae Wook Jeon. Xml based robot description language. In *2007 International Conference on Control, Automation and Systems*, pages 2477–2482. IEEE, 2007.

[8] Keenan A Wyrobek, Eric H Berger, HF Machiel Van der Loos, and J Kenneth Salisbury. Towards a personal robotics development platform: Rationale and design of an intrinsically safe personal robot. In *2008 IEEE International Conference on Robotics and Automation*, pages 2165–2170. IEEE, 2008.

[9] Rosen Diankov, Ryohei Ueda, Kei Okada, and Hajime Saito. Collada: An open standard for robot file formats. In *Proceedings of the 29th Annual Conference of the Robotics Society of Japan, AC2Q1–5*, 2011.

[10] Daniella Tola and Peter Corke. Understanding urdf: A survey based on user experience. *arXiv preprint arXiv:2302.13442*, 2023.

[11] Daniella Tola and Peter Corke. Understanding urdf: A dataset and analysis. *arXiv preprint arXiv:2308.00514*, 2023.

[12] Jack Collins, Shelvin Chand, Anthony Vanderkop, and David Howard. A review of physics simulators for robotic applications. *IEEE Access*, 9:51416–51431, 2021.