Towards Building AI-CPS with NVIDIA Isaac Sim: An Industrial Benchmark and Case Study for Robotics Manipulation

Zhehua Zhou[1†], Jiayang Song[1†], Xuan Xie[1†], Zhan Shu[1], Lei Ma[2,1*]
Dikai Liu[3], Jianxiong Yin[3], Simon See[3]
[1]University of Alberta, Canada   [2]The University of Tokyo, Japan   [3]NVIDIA AI Tech Centre

ABSTRACT

As a representative cyber-physical system (CPS), robotic manipulators have been widely adopted in various academic research and industrial processes, indicating their potential to act as a universal interface between the cyber and the physical worlds. Recent studies in robotics manipulation have started employing artificial intelligence (AI) approaches as controllers to achieve better adaptability and performance. However, the inherent challenge of explaining AI components introduces uncertainty and unreliability to these AI-enabled robotics systems, necessitating a reliable development platform for system design and performance assessment. As a foundational step towards building reliable AI-enabled robotics systems, in this paper, we propose a public benchmark for robotics manipulation. It leverages NVIDIA Omniverse Isaac Sim as the simulation platform, encompassing eight representative manipulation tasks and multiple AI software controllers. An extensive empirical evaluation is conducted to analyze the performance of AI controllers in solving robotics manipulation tasks, enabling a relatively thorough understanding of their effectiveness. To further demonstrate the applicability of our benchmark, we also developed the first falsification framework that is compatible with Isaac Sim. This framework bridges the gap between traditional falsification methods and modern physics engine-based simulations. The effectiveness of different optimization methods in falsifying AI-enabled robotics manipulation with physical simulators is also examined. Our work not only establishes a foundation for the design and development of AI-enabled robotics systems but also provides practical experience and guidance to practitioners in this field, promoting further research in this critical academic and industrial domain. The benchmarks, source code, and detailed evaluation results are available at https://sites.google.com/view/ai-cps-robotics-manipulation/home.

KEYWORDS

Cyber-physical System, Robotics Manipulation, Benchmarks, Deep Reinforcement Learning, Falsification

---

[†] Contributed equally to this research.

[*] Corresponding author, ma.lei@acm.org.

---

1 INTRODUCTION

Over the past decades, cyber-physical systems (CPSs) have emerged as an important area of research and development across various industrial sectors [13, 32, 54, 63]. As a complex system that integrates physical processes with software elements, modern CPS often comprises a diverse range of applications, e.g., autonomous vehicles [35, 52], smart grids [39, 60], and medical devices [16]. Among these applications, robotic manipulators hold great importance due to their wide-ranging industrial applications, spanning domains such as manufacturing [8], logistics [41], and agriculture [70]. Serving as a potential universal interface bridging the software and the physical worlds, robotic manipulators enable in-depth cyber-physical interactions. Recent studies in robotics manipulation have exhibited an increasing trend to embrace artificial intelligence (AI) techniques, particularly deep reinforcement learning (DRL) methods, as controllers to overcome the challenges and limitations associated with traditional control software [34]. As a notable example of AI-enabled CPS (AI-CPS) [48], these AI-enabled robotics manipulation systems exhibit the potential to address a broad spectrum of complex tasks and achieve higher levels of autonomy and universality.

However, the integration of AI techniques into robotics systems also introduces uncertainty and unreliability due to the challenges in interpreting the behavior of AI software controllers [38], necessitating a powerful development platform for system design and analysis. Considering factors such as safety, design/development cost, and time efficiency, a reliable and accurate simulator is often the central component of such a development platform [29, 30]. In recent robotics research, software simulators based on physics engines (referred to as *physical simulators* in this paper) have gained high preference for simulating complex industrial systems due to their enhanced accuracy, scalability, and flexibility [14, 62]. To address this need, our industrial partner, NVIDIA, has recently introduced NVIDIA Omniverse Isaac Sim [40, 44] (referred to as Isaac Sim in this paper), an advanced physical simulator that offers seamless integration with NVIDIA hardware and software, providing robust support for high-speed, GPU-accelerated simulation and AI training. Moreover, Isaac Sim provides an extensive asset library that simplifies and accelerates the development of diverse robotics
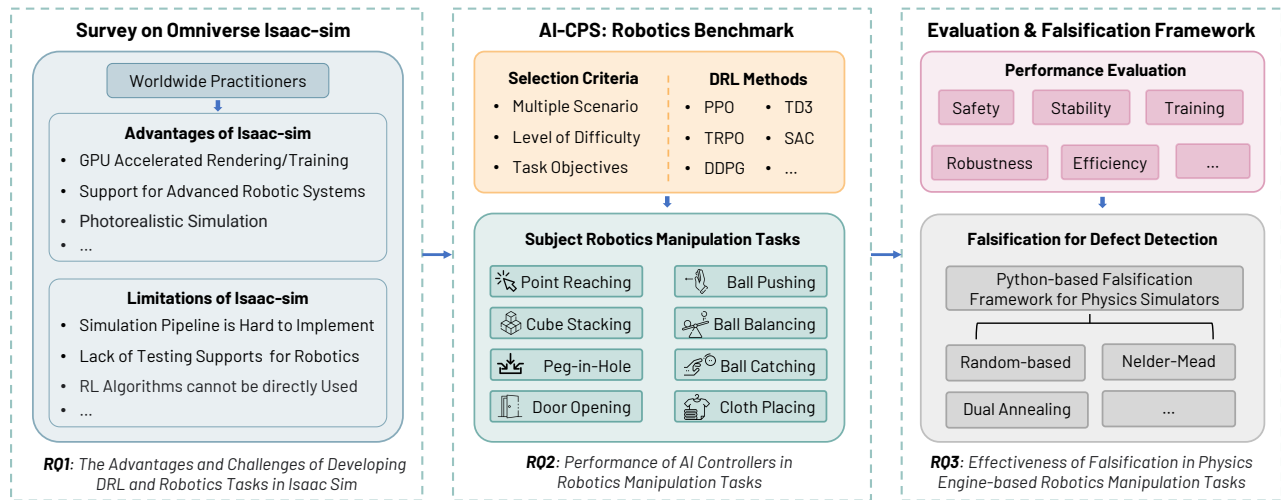
Figure 1: Workflow summary of this paper.

control scenarios. It is capable of providing highly realistic simulations that accurately model the system behaviors in the real world, making it an ideal simulator for the development of AI-enabled robotics applications.

Despite the promising capabilities of Isaac Sim, there remains a need for a deep understanding of its performance in various robotics tasks involving AI components. Furthermore, to the best of our knowledge, there is a lack of research on how to effectively and systematically utilize Isaac Sim as a development platform for AI-enabled robotic manipulators. Therefore, in this paper, we take the first step in investigating the perspectives of practitioners toward establishing the foundations to support research and development in AI-enabled robotics manipulation with Isaac Sim. Fig. 1 shows a high-level workflow overview of this paper, the key stages of which are summarized as follows.

To identify the most critical industrial demands, we first conducted a *survey* involving academic and industrial practitioners from the global robotics and AI communities. Through a series of questions, we gathered valuable insights into the advantages and drawbacks of Isaac Sim from various perspectives. In addition, we tried our best to seek participants' input on their requirements for designing and developing AI-enabled robotics applications. This survey not only reveals under which application scenarios Isaac Sim may outperform other simulators but also uncovers the community demands and future directions for developing a high-performing physical simulator. Based on the survey results, we recognize the urgent demand for a benchmark to unveil the characteristics of Isaac Sim in the context of developing AI-enabled robotics manipulation.

Therefore, to bridge this gap, we initiate an early exploratory study that establishes a public *benchmark* comprising eight representative robotics manipulation tasks, along with multiple software controllers trained with DRL algorithms. This benchmark could potentially serve as a cornerstone for supporting the design and development of AI-enabled robotics manipulation, promoting further research in this critical academic and industrial domain. To further evaluate the performance of AI controllers in manipulation tasks simulated with Isaac Sim, we conduct an extensive and in-depth

evaluation to assess their capabilities across different tasks. Our experimental results show that these controllers deliver satisfactory performance in a wide range of tasks.

In addition, we further notice a lack of dedicated *falsification* techniques specifically designed for physical simulators like Isaac Sim, which could be of great importance for defect detection at an early stage. In general, falsification plays a crucial role in the CPS development lifecycle as it helps identify scenarios or conditions in which the CPS fails or exhibits unsafe behaviors. However, most existing falsification tools are MATLAB-based and cannot interface directly with simulations driven by physics engines. The compressed data within these simulations introduces additional challenges in terms of accessibility and falsification analysis. Moreover, it remains uncertain whether traditional falsification methods are still effective when applied to the scope of physical simulators. To address this gap and demonstrate the practical applicability of our benchmark, we developed the first Python-based falsification framework that is compatible with Isaac Sim. Our findings indicate that the effectiveness of traditional falsification techniques varies across different tasks, calling for novel falsification techniques that can combine the characteristics of the domain-specific task, as well as the information from AI components.

The contributions of this work are summarized as follows.

- We conduct a survey among worldwide industrial and academic practitioners to compare Isaac Sim with other state-of-the-art physical simulators. The results offer valuable insights into the strengths and limitations of Isaac Sim and reveal the community demands and potential future developments.
- We establish a public benchmark of robotics manipulation with AI software controllers. Leveraging the capabilities of Isaac Sim, we integrate diverse manipulation tasks with unique challenges into a unified benchmark. It lays the foundation for the development of reliable and adaptable AI-enabled robotics manipulation systems.
- We perform a large-scale analysis and study to evaluate and explore the performance and characteristics of AI software controllers in robotics manipulation. Such an analysis discloses the
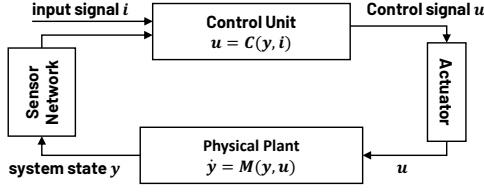
Figure 2: Abstract workflow of CPS.

benefits and challenges of AI controllers and provides insights into developing robust and competent CPSs.

- We developed the first Python-based falsification framework that is compatible with Isaac Sim. It bridges the gap between traditional falsification methods and modern simulation platforms. We additionally investigate the performance and effectiveness of existing optimization methods in falsifying AI-enabled robotics manipulation tasks simulated using physics engines.

As an early step towards developing resilient, secure, and fault-tolerant AI-enabled industrial-level robotics systems, our benchmark could potentially serve as a valuable resource for efficiently utilizing Isaac Sim as the development platform for AI software controllers in robotics manipulations, providing practical experience and guidance for researchers and practitioners in this field. Furthermore, the proposed benchmark is highly extensible and has the potential to be extended to support the entire development lifecycle of AI-CPSs, encompassing planning, design, testing, and deployment stages. The early exploration, benchmark, and study in this emerging direction could potentially contribute to advancements in AI-enabled robotics and other diverse CPSs, paving the way for innovative software solutions in various real-world scenarios.

2 BACKGROUND

In this section, we first provide an overview of AI-CPS and AI-enabled robotics manipulation. Then, a high-level introduction to signal temporal logic (STL) and falsification technique is presented.

2.1 Robotics Manipulation as AI-CPS

As shown in Fig. 2, a CPS typically comprises four major components: a physical plant M, a software control unit, an actuator, and a network of sensors. Currently, AI technologies have been applied across various components in CPSs, such as perception units and decision-making units. In this work, we specifically focus on AI-CPSs that use DRL algorithms as AI software controllers to achieve real-time control of a physical plant M. The control command u is computed by the AI controller C, which makes control decisions based on the current state y and an external input i. After receiving the control command u, the plant evolves to the next state according to the dynamics of the physical system.

As a typical example of CPS, the robotic manipulator integrates physical components, e.g., the robot arm and the sensors, with digital devices, e.g., the software controller, to perform tasks in the real world. The dynamics of the manipulator is often described as

$$M(q)\dot{q} + C(q, \dot{q}) + G(q) = \tau \tag{1}$$

where $q = [q_1, q_2, \ldots, q_n]^T$ and $\dot{q} = [\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n]^T$ are the joint angles and angular velocities, respectively. n indicates the degrees of freedom (DoF) of the robotic manipulator. $M(q)$, $C(q, \dot{q})$ and

$G(q)$ are the mass matrix, the vector of centrifugal and Coriolis terms, and the gravity terms, respectively. τ represents the torques applied on each joint of the manipulator.

By taking the current system states, e.g., q and $\dot{q}$, as well as external sensor information, e.g., the status of the manipulation objects, as the input, the control objective of robotics manipulation is thus to design a controller C that generates a sequence of control commands $u = \tau$ to fulfill the desired task requirements. However, traditional controllers usually rely on accurate modelling of the system behavior, which in complicated tasks, e.g. manipulating deformable objects, is hard to achieve. Moreover, designing a controller based on a specific model limits its capability of performing well in different environments and tasks, which further diminishes its applicability and generalizability.

To address these challenges, recent studies in robotics manipulation leverage AI techniques, e.g., DRL [4], to learn a control policy directly from data, making it a typical and representative example of AI-CPS. Therefore, by focusing on evaluating and falsifying these AI-enabled robotics manipulation tasks, we are able to also obtain a thorough understanding of how modern AI-CPS will perform in actual critical industrial domains.

2.2 STL and Falsification

STL [18] is a specification language designed to describe the expected *temporal*-related behavior of a system, e.g., safety and performance. It is equipped with *quantitative robust semantics*, which quantitatively measures the degree of satisfaction of the specification. Formally, given an STL specification φ and a system output $M(i)$, where i is the input signal, the STL semantics $\text{ROB}(M(i), \varphi)$ maps $M(i)$ and φ to a real number. A positive/negative value of $\text{ROB}(M(i), \varphi)$ represents that the specification is satisfied/violated, and a larger value implies a stronger satisfaction/violation. See [17] for a detailed description of the robust semantics.

As a widely used safety assurance technique, *falsification* [15, 17, 74–76] aims to discover input signals to a system that could violate a desired specification. It transforms the discovery process into an optimization problem, with the goal of minimizing the robust semantics such that $\text{ROB}(M(i), \varphi)$ is less than zero. Hill-climbing based optimization methods [69], e.g., dual annealing [65] and Nelder-Mead [25], are commonly used to solve this problem. Typically, the optimization process involves first providing some initial samples to the system and then proposing new samples based on the feedback received, which is represented by the robust semantics in this case.

3 RESEARCH QUESTIONS

In this paper, We mainly investigate the following three research questions (RQs) to have a better understanding of the challenges and opportunities of building reliable AI-enabled robotics manipulation systems utilizing Isaac Sim.

RQ1: What are the advantages and limitations of Isaac Sim compared to other physical simulators?

Although Isaac Sim is recognized for its seamless integration with NVIDIA hardware, the requirements and challenges in simulating robotics manipulations are diverse and often dependent on the tasks. In the field of robotics, there are numerous physical

simulators, such as Gazebo [36] and PyBullet [20], that have been widely used by researchers and practitioners to meet their specific demands. As a novel simulation platform that is still in its developmental stages, there is a need for an analysis of the strengths and limitations of Isaac Sim compared to other physical simulators, as well as its performance in simulating various robotics tasks.

Therefore, to study RQ1, we conducted a survey with industrial and academic practitioners around the world to gather their comments on the advantages and limitations of Isaac Sim as compared to other physical simulators. The purpose of the survey is to gain insights into the unique features and benefits of Isaac Sim that may make it stand out from other simulators, as well as to identify any potential drawbacks or areas where improvements could be made. Details about the RQ1 are presented in Sec. 4.

RQ2: How do AI controllers perform in various robotics manipulation tasks?

The findings from RQ1 (see Sec. 4.2 for details) highlight the necessity of a benchmark that aids users in initiating and familiarizing themselves with the utilization of Isaac Sim for the development of AI-enabled robotics applications. To address this requirement, we develop a benchmark comprising eight typical robotics manipulation tasks that cover rigid body manipulation, deformable object manipulation, prehensile manipulation, and non-prehensile manipulation, using Isaac Sim.

In the meantime, despite extensive research on DRL controllers in robotics, there is still a lack of systematic and detailed analysis of their performance in diverse robotics manipulation tasks. This knowledge gap emphasizes the need to compare and evaluate different DRL controllers under various task requirements. Therefore, we design RQ2 that aims to compare different DRL controllers across a wide range of manipulation tasks. For this purpose, we propose various metrics to evaluate and examine the performance of different DRL controllers on the tasks presented in our benchmark. The result reveals the strengths and limitations of different AI controllers in robotics manipulation and allows researchers to identify the most effective methods for achieving specific control objectives. Details about this RQ are presented in Sec. 5.

RQ3: How effective are different optimization methods in falsifying physics engine-based robotics manipulation tasks?

The findings of RQ1 also emphasize the need for falsification support in the development of AI-enabled robotics applications with Isaac Sim. Building upon this insight, we develop the first Python-based falsification framework that can be directly used with Isaac Sim. However, although falsification has proved to be effective on traditional CPSs, its efficacy in robotics tasks simulated using modern physical simulators remains unclear. To address this gap, we introduce RQ3 to analyze the performance of different optimization methods, i.e., random, Nelder-Mead, and dual annealing, in falsifying robotics tasks with AI software controllers. By using our benchmark, we conduct a falsification experiment that reveals the robustness and reliability of AI controllers and helps identify potential failures and vulnerabilities in the system. Additionally, this experiment also demonstrates the extensibility and applicability of our benchmark. Details about RQ3 are given in Sec. 6.

Hardware & Software Dependencies. The simulation, DRL training and falsification were conducted using SKRL [53], RTAMT [43] and Scipy [61] libraries. For the hardware platform, we use Lambda Tensorbook equipped with Intel® Core™ i7-10870H CPU and NVIDIA RTX 3080 Max-Q GPU.

4 ADVANTAGES AND LIMITATIONS OF ISAAC SIM

As a first step towards understanding the demands of industrial and academic practitioners, we conducted a survey to get opinions about the advantages and limitations of Isaac Sim, as an answer to **RQ1: What are the advantages and limitations of Isaac Sim compared to other physical simulators?** In this section, we describe the survey setting and present the results.

4.1 Design and Conduct of Survey

Survey Design. The survey consists of three question types: short-answer, multiple-choice, and Likert scale questions. The Likert scale questions ask respondents to indicate their level of agreement with statements such as Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree, and I don't know. The survey questions are designed to understand robotics development with Issac Sim, which consist of five parts:

- *Demographics:* We aim to gather information about the practitioners' demographics and experience in robotics development.
- *General robotics design using Isaac Sim:* We target to investigate the pros and cons of Isaac Sim in terms of general robotics simulation and development pipeline.
- *Development challenges of DRL robotics tasks in Isaac Sim:* In this part, we ask about the practitioners' experience in developing DRL robotics tasks in Isaac Sim.
- *Comparison with other robotics simulators:* This part asks the practitioners to compare Isaac Sim with other popular robotics simulators, such as Gazebo and PyBullet.
- *Testing of robotics tasks in Isaac Sim:* This part studies the practitioners' habits in performing testing in robotics tasks and the status of testing support in Isaac Sim.

We also provide an open-ended question at the end of the survey, inviting practitioners to share their opinions and comments about Isaac Sim in the context of robotics and DRL development.

Participant Recruitment. We recruited practitioners through two methods: email contact and forum recruitment. To obtain email contacts, we searched for research papers published in top conferences such as ICSE, ESEC/FSE, ASE, ICRA, IROS, AAAI, IJCAI, and NeurIPS from 2019 to 2023 that involve developing robotics tasks in Isaac Sim. We collected the email addresses of the authors of these papers, resulting in a list of 75 researchers whom we contacted individually via email. For forum recruitment, we posted topics about the survey in various online communities, including the Isaac Sim forum [45], the NVIDIA official Discord community, and the GitHub discussion page of relevant libraries, e.g., SKRL [53] and Isaac Gym [40]. In total, we received 17 responses from nine different countries. It should be noted that the participant recruitment is reasonably biased towards robotics experts, as the aim of the study is to understand their opinions. Due to the page limit, details about the information of the respondents can be found on our website.

(a) Isaac Sim for developing general robotics simulation

(b) Isaac Sim for developing DRL modules

(c) Comparison to other platforms

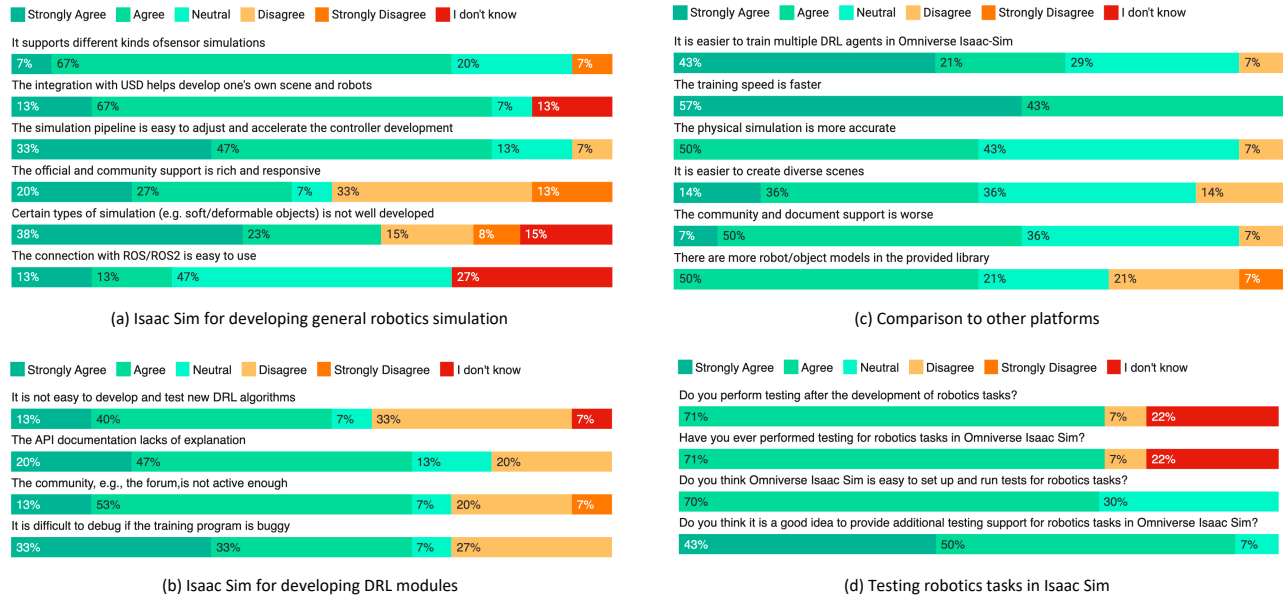(d) Testing robotics tasks in Isaac Sim

Figure 3: RQ1 - Partial questions and results from survey

4.2 Result Analysis and Findings

Fig. 3 presents the survey results from four important perspectives (the complete version can be found on our website). As depicted in Fig. 3(a), the majority of practitioners agree that Isaac Sim supports diverse types of sensors (74%), and the USD (Universal Scene Design) file format [57] makes it easier to develop custom scenes and robots (80%). However, some respondents point out that certain simulation features, such as simulating soft or deformable objects, are not well developed.

Regarding challenges in developing DRL tasks in Isaac Sim, Fig. 3(b) shows that most practitioners think that the API documentation lacks proper explanation (67%) and the community support is not active enough (66%), which highlights the need for improved documentation and official support for users of Isaac Sim. Moreover, more than half (53%) of the practitioners consider that it is not easy to develop and test new DRL algorithms.

In terms of comparison with other platforms (see Fig. 3(c)), all practitioners believe that the training speed in Isaac Sim is faster than other simulation platforms due to its good compatibility with NVIDIA GPUs. Additionally, most of them find it easier to train multiple DRL agents (63%) and create simulation scenes (50%) in Isaac Sim due to its rich asset library. However, more than half of the practitioners (57%) think that the community and documentation support is not as good as in other platforms. Moreover, as the simulation prototype is ultimately expected to be deployed in the real world, we are also interested in the accuracy of Isaac Sim's physical simulation. While half of the practitioners think that the simulation is more accurate, the other half are neutral or disagree with this point. In addition, we ask practitioners about their opinions on testing robotics tasks in Isaac Sim (see Fig. 3(d)). Most of them (93%) believe that it is a good idea to provide additional testing support, with one practitioner stating, "*An advanced plug-and-play*

testing module will be an asset.". This encourages our development of a falsification framework for Isaac Sim.

In conclusion, numerous participants have pointed out the lack of a clear pipeline for DRL and robotics development in the current version of Isaac Sim. Notably, many responses indicate that users face challenges in initiating their development process due to insufficient instructions and documentation. Moreover, the absence of a baseline for comparison and calibration poses difficulties for developers who lack guidance and standards to assess their progress accurately. According to this, we propose an AI-enabled robotics manipulation benchmark that is able to greatly benefit practitioners in the following aspects: 1) an easy-to-use development playground and pipeline; 2) a standard baseline for observation and comparison; and 3) an extensible and applicable environment for quick demonstration and prototyping.

Answer to RQ1: Isaac Sim has gained recognition for its fast training speed and extensive asset library. However, practitioners have expressed concerns regarding the lack of a clear pipeline for DRL training and robotics development. A benchmark of AI-enabled robotics manipulation is desired. Moreover, there is a demand for testing support within Issac Sim.

---

5 BENCHMARK OF ROBOTICS MANIPULATION

As revealed by Sec. 4.2, practitioners are seeking a versatile and user-friendly environment that facilitates quick prototyping and demonstration of AI-enabled robotics systems. Consequently, we develop a public benchmark for AI-enabled robotics manipulation with high extensibility and applicability as a timely solution to this demand, as well as to **RQ2: How do AI controllers perform in various robotics manipulation tasks?** In this section, we first describe how the benchmark is construed based on Isaac Sim. Then,
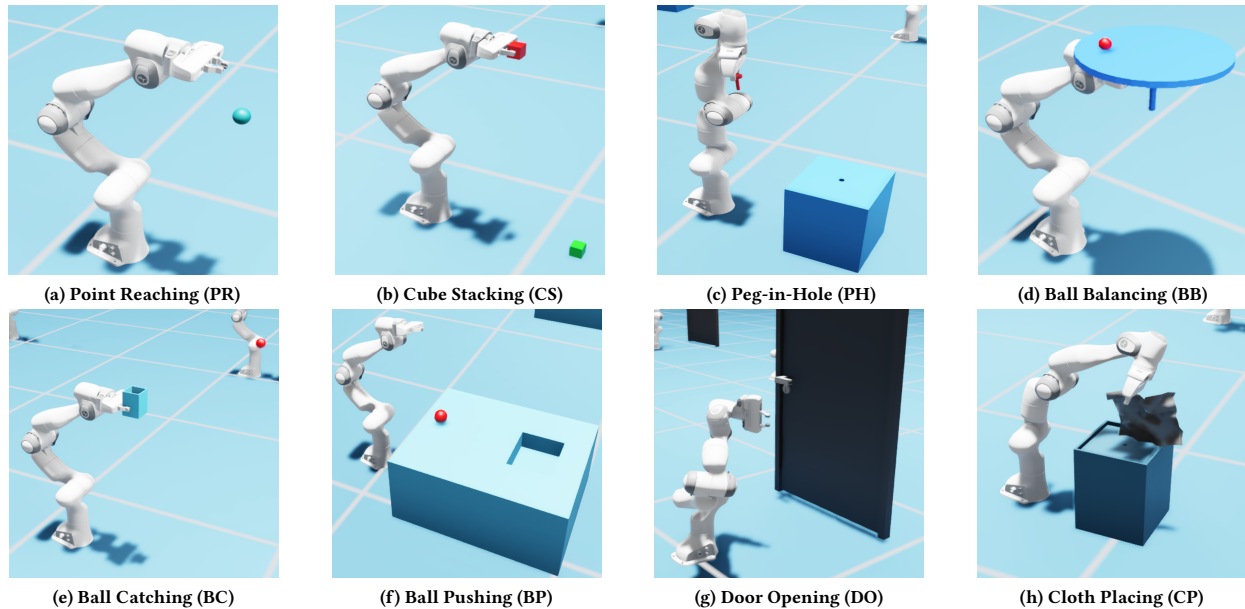
Figure 4: Robotics manipulation tasks included in the proposed benchmark based on Isaac Sim.

by using the developed benchmark, we systematically evaluate the performance of AI software controllers. In particular, we introduce four evaluation metrics and present the evaluation results.

5.1 Benchmark Construction

Our benchmark includes eight typical and diverse robotics manipulation tasks (see Fig. 4). The task selection is based on dominant topics and needs within the manipulation field [8]. Our aim is to cover a diverse range of tasks with varying characteristics, such as rigid body manipulation, soft/deformable object manipulation, and non-prehensile manipulation [49]. Each task requires a unique set of control strategies and presents a varying level of difficulty, ranging from simple to complex, to examine the performance and adaptivity of AI controllers.

The manipulator model used in all tasks is the Franka Emika Panda [21], a state-of-the-art robotic arm widely used for various manipulation tasks in research and industry [26, 47, 71]. A detailed introduction to the presented benchmark is given as follows.

Tasks. The benchmark contains a total of eight tasks, each designed to evaluate different aspects of a robot's manipulation capabilities.

- *Point Reaching (PR):* The robot needs to reach a specific point in 3D Cartesian space by using its end-effector. This task is a fundamental and essential functionality that a manipulator should possess.
- *Cube Stacking (CS):* In this task, the robot needs to grasp a cube-shaped object and accurately place it on top of another cube-shaped object that serves as the target. This pick-and-place like task is a common task in industrial settings, e.g., assembly lines.
- *Peg-in-Hole (PH):* The robot needs to accurately insert a cylindrical object into a corresponding hole. This task is commonly used in manufacturing processes, such as circuit board assembly.

- *Ball Balancing (BB):* The objective of this task is to balance a ball at the center of a tray that is held by the robot's end-effector. This task is useful for applications such as stabilizing a moving platform.
- *Ball Catching (BC):* The robot needs to catch a ball that is thrown to it using a tool. This task demands the capability to handle moving objects.
- *Ball Pushing (BP):* The robot is required to push a ball towards a target hole on a table. As a typical non-prehensile manipulation, this task is important in many industrial applications, such as material handling and conveyor systems.
- *Door Opening (DO):* The robot attempts to open a door using its gripper. This is a more advanced manipulation task and requires a multi-stage control process.
- *Cloth Placing (CP):* The robot has to move and place a piece of cloth onto a target table. As a soft object manipulation task, it often requires a more complex controller than rigid body object manipulations.

AI software controllers. We employ various DRL algorithms, such as *Trust Region Policy Optimization* (TRPO) [50], *Deep Deterministic Policy Gradient* (DDPG) [37], *Soft Actor-Critic* (SAC) [27], *Proximal Policy Optimization* (PPO) [51], and *Twin-Delayed Deep Deterministic* (TD3) [24], as the AI software controller of the robotic manipulator. These algorithms are based on the implementation given in the SKRL library [53]. The AI controller takes the states of the manipulator and the object to be manipulated as input and generates the control command for each robot joint. For each task, we design a specific reward function for training the AI controllers, which is detailed on our website.

Learning Environments. To facilitate training and evaluation, we wrap all tasks within the Omniverse Isaac Gym Reinforcement

Table 1: Performance evaluation of AI controllers in robotics manipulation tasks (better results are highlighted in gray).

Task	Initial Configuration	Controller	TT	No noise			With noise		
				SR	DBR	TCT	SR	DBR	TCT
PR	Position of the target point (x, y, z-coordinates) $x \in [0.3, 0.7]$ $y \in [-0.4, 0.4]$, $z \in [0.4, 0.8]$	PPO	10500	93%	0.26%	29.29	87%	0.27%	29.83
		TRPO	26700	100%	0.07%	16.21	98%	0.12%	18.39
CS	Position of the target cube (x, y-coordinates) $x \in [0.4, 0.8]$, $y \in [-0.1, 0.3]$	PPO	11300	100%	7.13%	75.66	98%	24.77%	75.74
		TRPO	12600	99%	8.98%	66.44	98%	15.62%	66.80
PH	Position of the target hole (x, y-coordinates) $x \in [0.3, 0.7]$, $y \in [-0.2, 0.2]$	PPO	154000	85%	3.59%	16.79	78%	4.41%	17.83
		TRPO	259600	92%	1.90%	43.53	90%	2.41%	46.57
BB	Initial position of the ball (x, y-coordinates) $x \in [0.2, 0.5]$, $y \in [-0.15, 0.15]$	PPO	11500	98%	0.47%	6.41	89%	3.44%	10.24
		TRPO	12900	100%	0.01%	5.91	93%	2.23%	6.01
BC	Initial position of the ball (x, y-coordinates) $x \in [1.05, 1.15]$, $y \in [-0.05, 0.05]$	PPO	20800	100%	6.34%	23.14	99%	6.96%	23.39
		TRPO	28100	97%	8.69%	21.60	93%	9.14%	21.86
BP	Initial position of the ball (x, y-coordinates) $x \in [0.4, 0.6]$, $y \in [-0.1, 0.1]$	PPO	60300	98%	9.28%	53.53	97%	13.78%	55.45
		TRPO	175100	100%	6.85%	37.12	100%	7.88%	38.27
DO	Position of the door (x, y-coordinates) $x \in [0.75, 0.85]$, $y \in [-0.1, 0.1]$	PPO	124400	89%	36.60%	136.62	81%	39.22%	154.63
		TRPO	128300	97%	31.11%	126.36	94%	32.74%	127.06
CP	Position of the target table (x, y-coordinates) $x \in [0.45, 0.75]$, $y \in [-0.35, 0.35]$	PPO	34700	100%	2.44%	12.15	99%	2.85%	14.87
		TRPO	22400	99%	14.38%	27.45	99%	15.08%	29.25

Learning Environment [46], which is built on top of the OpenAI Gym framework [10]. This provides better compatibility with Isaac Sim simulations and other DRL libraries and allows for easy extensions in the future.

Initial Configurations. By treating each manipulation task as an independent entity, we consider the initial configuration of either the object to be manipulated or the target object as the input signal to the system. During the training process of the AI controller, we vary the initial configuration to evaluate its performance. The allowable range of initial configurations for each task used in this paper is specified in Table 1. All position values represent the relative Cartesian distances to the base of the manipulator.

5.2 Evaluation Metrics

With the developed benchmark of robotics manipulation tasks, we are able to answer RQ2. For this purpose, we perform the evaluation based on four categories of evaluation metrics that consider different aspects of the manipulation tasks:

- *Success Rate (SR):* This metric measures the percentage of successful task completions among all attempted trials. Successful task completion is defined as when the controlled system behavior satisfies a predefined task-relevant STL specification (see Table 2 for the STL specifications used in our benchmark). These STL specifications are grounded in existing literature and benchmark problems [8, 12, 19]. A high SR indicates that the AI controller is capable of effectively completing the task.
- *Dangerous Behavior Rate (DBR):* DBR is computed as the percentage of time steps, i.e., control intervals, where the manipulator is close to failing the task, among all simulated time steps. A high DBR indicates that the AI controller is prone to generating unsafe or unstable control commands, which can potentially cause task failures. The STL specifications used in our benchmark for describing dangerous behaviors are given in Table 2.
- *Task Completion Time (TCT):* This metric measures the time steps needed by the manipulator to successfully complete the task. A

shorter TCT indicates that the AI controller is capable of generating more efficient control commands to complete the task.
- *Training Time (TT):* We use TT to measure the training time steps required to train the AI controller, i.e., the policy reward converges to a constant level. A shorter training time indicates that the used DRL algorithm is more efficient in learning the control policy for the given task.

It is worth mentioning that, in real-world applications, system noises and uncertainties, such as sensor noises or model inaccuracies, are often inevitable. Thus, the ability of manipulators to remain robust under such conditions is crucial, especially for those designed for real-world tasks. Therefore, we consider *robustness* as our final critical evaluation metric. For this, we introduce action noises to the trained AI controllers and measure how the SR, DBR, and TCT are impacted. The influences of the noise indicate the potential of AI controllers in bridging the simulation-to-reality gap when implemented in real-world manipulators. Details about our evaluation results are presented in the following subsection.

5.3 Experimental Evaluations

Experiment Settings. To evaluate the performance of AI-enabled robotics manipulation, we first use various DRL algorithms, i.e., TRPO, PPO, SAC, TD3, and DDPG, to train multiple AI software controllers for each manipulation task. However, our experiments reveal that except for TRPO and PPO, the other DRL algorithms fail to produce a working controller that is capable of solving the manipulation task. One potential explanation for this could be that, compared to off-policy algorithms, on-policy algorithms are better suited for Isaac Sim, which employs parallel running of a large number of environments [46]. In such a case, the collected samples are strongly correlated in time, and a considerable amount of information is accumulated in each time step, posing challenges for off-policy algorithms. Further research is needed to investigate whether this is the root cause of the issue. Consequently, we use

**Table 2: STL specifications for different robotics manipulation tasks (□: Globally. ◇: Eventually).**

Task	Successful Task Completion	Dangerous Behavior
PR	$\square_{[0,30]}(\|\mathsf{finger_{pos}} - \mathsf{point_{pos}}\| \leq 0.3)$	$\square_{[0,30]}(\|\mathsf{finger_{pos}} - \mathsf{point_{pos}}\| \leq 0.12)$
CS	$\diamondsuit_{[0,30]}(\|\mathsf{cube_{pos}} - \mathsf{target_{pos}}\| \leq 0.024 \wedge$ $\|\mathsf{cube_{pos\_z}} - \mathsf{target_{pos\_z}}\| \geq 0)$	$\square_{[0,30]}(\|\mathsf{cube_{pos}} - \mathsf{target_{pos}}\| \leq 0.35 \vee$ $\|\mathsf{cube_{pos\_z}} - \mathsf{target_{pos\_z}}\| \geq 0.02)$
PH	$\square_{[25,30]}(\|\mathsf{obj_{pos}} - \mathsf{hole_{pos}}\| \leq 0.12)$	$\square_{[25,30]}(\|\mathsf{obj_{pos}} - \mathsf{hole_{pos}}\| \leq 0.37)$
BB	$\square_{[5,20]}(\|\mathsf{ball_{pos}} - \mathsf{tray_{pos}}\| \leq 0.25)$	$\square_{[5,20]}(\|\mathsf{ball_{pos}} - \mathsf{tray_{pos}}\| \leq 0.2)$
BC	$\square_{[5,30]}(\|\mathsf{ball_{pos}} - \mathsf{tool_{pos}}\| \leq 0.1)$	$\square_{[5,30]}(\|\mathsf{ball_{pos}} - \mathsf{tool_{pos}}\| \leq 0.2)$
BP	$\diamondsuit_{[0,30]}(\|\mathsf{ball_{pos}} - \mathsf{hole_{pos}}\| \leq 0.3)$	$\diamondsuit_{[0,30]}(\|\mathsf{ball_{pos}} - \mathsf{hole_{pos}}\| \leq 0.5)$
DO	$\diamondsuit_{[0,30]}(\mathsf{door_{yaw}} \geq 20)$	$\diamondsuit_{[0,30]}(\mathsf{door_{yaw}} \geq 0.1)$
CP	$\diamondsuit_{[0,30]}(\|\mathsf{cloth_{pos}} - \mathsf{table_{pos}}\| \leq 0.25)$	$\square_{[0,30]}(\|\mathsf{cloth_{pos}} - \mathsf{table_{pos}}\| \leq 0.3 \vee$ $\|\mathsf{cloth_{pos\_z}} - \mathsf{ground_{pos\_z}}\| \geq 0.02)$

only the AI controllers trained by PPO and TRPO in our performance evaluation and the falsification experiment presented in the next section, as they are the only functional ones.

For each task and AI controller, we conduct 100 trials separately under two different conditions: without and with action noise, where the action noise is a white Gaussian noise with a variance of 0.25. For each trial, the initial configuration is generated randomly according to Table 1, and the simulation length is set to 300 time steps. The values of DBR and TCT are averaged among all trails.

Experiment Results. The results are presented in Table 1. It can be observed that both TRPO and PPO are able to accomplish most of the manipulation tasks with an SR over 90%, except for PPO in PH (85%) and DO (89%). While TRPO maintains a comparable SR to PPO in tasks like CS, BC, and CP, it has a better performance in other tasks, particularly for those that require precise control (e.g., PH) or a multi-stage control process (e.g., DO). As a result, TRPO generally outperforms PPO in terms of SR. Similar trends can also be observed in the metrics DBR and TCT, where TRPO usually has a better performance with the exception that for the task CP, PPO has a more clear advantage. However, TRPO often requires a longer TT, especially for tasks PR, PH, and BP, where as much as twice of the TT is required compared to PPO.

As expected, the introduction of action noise leads to a decrease in the performance of AI controllers across all metrics. However, both PPO and TRPO are still able to accomplish the tasks with a satisfactory SR, where the decrease is less than 5% for most of the tasks. Tasks that require accurate execution of actions, such as PH, BB, and DO, exhibit a more noticeable decrease in SR. In terms of the DBR, the action noise has a strong impact on the CS and BB tasks. One potential reason could be that these tasks have less tolerable space for manipulating objects, e.g., the size of the target cube in CS or the tool in BB, making imprecise actions more dangerous. The impact of the action noise on the DBR of other tasks as well as on the TCT is marginal. Overall, both PPO and TRPO show good performance in terms of robustness against action noise.
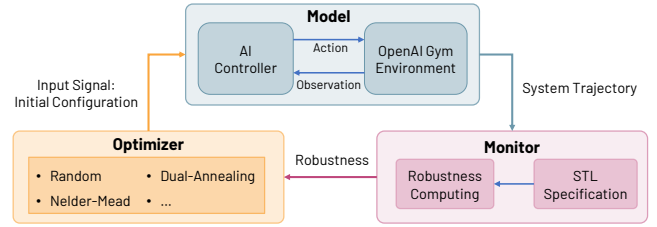
Figure 5: Structure of the proposed falsification framework.

> **Answer to RQ2:** The AI controllers trained using PPO and TRPO exhibit satisfactory performance in robotics manipulation tasks. In general, TRPO outperforms PPO in terms of SR, while PPO requires less TT. Moreover, both controllers demonstrate a good level of robustness against action noise.

6 FALSIFICATION FOR DEFECT DETECTION IN ISAAC SIM

As discussed in Sec. 4.2, there is an urgent need to provide testing support for Isaac Sim. However, to the best of our knowledge, a dedicated testing tool specifically designed for physical simulators is currently lacking. To bridge this gap, we focus on falsification, a widely-used testing technique specified for CPS, and develop the first Python-based falsification framework that can be seamlessly integrated with Isaac Sim. By leveraging this framework, we compare the effectiveness of three commonly used optimization methods, i.e., random, Nelder-Mead, and dual annealing, in falsifying subject tasks, as an answer to **RQ3: How effective are different optimization methods in falsifying physics engine-based robotics manipulation tasks?**

6.1 Falsification Framework

Based on the proposed benchmark in Isaac Sim, we develop accordingly an extendable falsification framework to evaluate the performance of AI controllers in physics engine-based simulations. Fig. 5 shows the structure of the falsification framework. It consists of three parts: the system model, the monitor, and the optimizer.

Table 3: Performance of different optimization methods in the falsification experiment (best results are highlighted in gray).

Task	Controller	Random			Nelder-Mead			Dual Annealing		
		#Suc. Fals.	Avg. Time (s)	Avg. #Sim	#Suc. Fals.	Avg. Time (s)	Avg. #Sim	#Suc. Fals.	Avg. Time (s)	Avg. #Sim
PR	PPO	30	15.10	8.57	7	15.03	8.43	30	27.76	16.37
	TRPO	0	-	-	2	206.28	128.50	23	173.26	107.83
CS	PPO	1	482.68	201	2	411.56	172.00	10	436.81	182.20
	TRPO	12	429.94	171.17	12	408.05	167.08	16	317.58	126.31
PH	PPO	26	222.09	117.19	17	249.76	132.59	30	117.35	93.97
	TRPO	27	223.78	120.41	13	189.63	102.00	26	220.29	118.27
BB	PPO	3	199.22	101.33	7	187.56	96.14	9	98.85	50.22
	TRPO	5	225.87	111.60	5	256.52	126.6	6	246.68	121.67
BC	PPO	5	510.06	195.00	4	400.12	153.00	6	390.68	146.50
	TRPO	16	164.51	56.69	12	264.35	90.50	25	185.03	64.32
BP	PPO	26	222.09	117.19	17	249.76	132.59	30	117.35	93.97
	TRPO	27	223.78	120.41	13	189.63	102.00	26	220.29	118.27
DO	PPO	30	25.31	11.13	30	21.78	9.50	30	17.09	7.50
	TRPO	30	58.88	25.83	30	56.21	24.30	30	54.60	23.67
CP	PPO	16	121.53	100.00	2	158.08	127.00	28	62.40	50.43
	TRPO	21	1211.48	107.52	14	1677.16	99.64	30	754.18	83.13

Model: The model is responsible for simulating the physical behavior in different robotics manipulation tasks. For each input signal, i.e., the initial configuration, a trained AI software controller attempts to control the manipulator to accomplish the task. This results in a sequence of observations, i.e., the system trajectory, that will be used in the monitor to verify the completeness of the task. As aforementioned, in our benchmark, each task is wrapped as an Omniverse Isaac Gym environment. To facilitate compatibility with other physical simulators, we further wrap all task environments into standard OpenAI Gym environments by using the wrapper provided in the SKRL library [53]. This provides the developed falsification framework with the potential to be used not only with Isaac Sim but also with other physical simulators that utilize OpenAI Gym environments, e.g., Mujoco [59] or PyBullet [20].

Monitor: The monitor takes the system trajectory generated from the simulation as input and computes the robustness of the AI controller's behavior against predefined STL specifications. We use RTAMT [43], a Python library for monitoring STL specifications, to compute the robustness values. The STL specifications employed in our falsification experiments are identical to those used to define successful task completions, as detailed in Table 2.

Optimizer: The optimizer searches for worst-case scenarios where the AI controller fails to meet the STL specifications, i.e., the initial configurations resulting in a system trajectory with minimal robustness. Such a search process is considered an optimization problem where the objective function is the combination of the model and the monitor, which returns the corresponding robustness value for a given input signal. We implement three optimization algorithms, namely random, Nelder-Mead, and dual annealing, in the proposed falsification framework. The latter two algorithms are based on the Scipy library [61].

The proposed falsification framework provides a rigorous and systematic approach for evaluating the reliability of various AI controllers in robotics manipulation tasks. Based on it, we conducted a falsification experiment using our benchmark. The results are detailed in the next subsection.

6.2 Experimental Evaluations

To answer RQ3, i.e., how effective are different optimization methods in falsifying physics engine-based robotics manipulation tasks, we falsify the trained AI controllers from Sec. 5.3 with the proposed falsification framework. For each AI controller, following the typical setting in the falsification community [72, 74–76], we conduct 30 falsification trials, each consisting of a maximum of 300 task simulations.

The number of successful falsifications, and the average time and number of task simulations used for finding the first falsification in each trail are presented in Table 3. The results of the falsification experiment show that dual annealing outperforms other methods. While achieving only one less successful falsification than the random approach in the BP and PH tasks with the TRPO controller, it has a noticeable advantage in other tasks. In contrast, Nelder-Mead performs poorly in robotics manipulation tasks. A possible reason for this could be that, due to the highly nonlinear function of the robustness of STL specifications, a large number of local optima exist. In such cases, heuristic direct search methods like Nelder-Mead are prone to get stuck in local optima and, therefore, may fail to falsify these tasks.

It is also worth mentioning that achieving high rewards in the training process does not necessarily indicate that the AI controller will reliably accomplish its desired task. As mentioned in [78], success in reward does not guarantee success in task completion. Our experiment also shows that even well-trained AI controllers can still be falsifiable by state-of-the-art falsification techniques. This highlights the necessity of incorporating falsification techniques into the testing framework of AI-enabled robotics manipulation.

> **Answer to RQ3:** The effectiveness of different optimization methods varies in falsifying physics engine-based robotics manipulation tasks. Dual annealing shows satisfactory falsification results, while Nelder-Mead has a relatively poor performance. Considering the task-specific characteristics may be crucial when developing a highly effective falsification method for AI-enabled robotics applications.

7 DISCUSSION AND FUTURE DIRECTION

Discussion. Based on the findings of our survey, we recognize that a versatile environment, e.g., a benchmark, is crucial for the development of AI-enabled robotics systems. Practitioners seek an intuitive pipeline and plug-and-play solutions to streamline their efforts and avoid getting entangled in inconsequential steps. In response to this demand, we construct our benchmark with an emphasis on ease of use, i.e., developers can easily deploy their methods and techniques on our benchmark without the need for extensive low-level adjustments. Such a benchmark could greatly facilitate the development of AI-CPSs in industrial applications with Isaac Sim, and also aid the creation of innovative AI solutions to address industry challenges. Moreover, our evaluation provides insights into the challenges and potential opportunities involved in developing AI software controllers with Isaac Sim. Notably, existing DRL algorithms may neglect the impact of the parallel running of learning environments associated with Isaac Sim. Further research is needed to design and develop AI methods that effectively account for and exploit this characteristic.

We also notice that, while AI controllers demonstrate good performance in manipulation tasks, their reliability often cannot be guaranteed. Hence, before the actual real-world deployment, a thorough falsification is crucial for AI-CPSs. To address this need, we developed the first Python-based falsification framework that is compatible with Isaac Sim. This could greatly improve the reliability of falsification and pave the way for smoother future deployments of AI-CPSs in real-world industrial scenarios. However, the performance of different optimization techniques in falsification can be influenced by the properties of the control task. Therefore, taking into account the characteristics of the task is important for performing effective falsification, and different methods may be necessary for different tasks.

Future direction. For applying AI controllers in real-world industrial applications, an analysis of the simulation-to-reality gap is critical. Therefore, one of our future research directions is to investigate the accuracy of Isaac Sim's simulations in relation to real-world results. Such an analysis could facilitate the deployment of AI-CPSs in real-world tasks. Another possible direction is to expand our benchmark to encompass other AI-CPSs, such as AI-enabled autonomous vehicles or UAVs. This would amplify the adaptability of our benchmark, catering to a broader range of diverse AI-CPSs. Moreover, we also aim to incorporate more state-of-the-art optimization methods, such as those available in Breach [17] or S-Taliro [2] libraries, into our falsification framework. This would increase the versatility of our framework, making it useful not only for robotics manipulation tasks but also for other types of AI-CPSs.

8 THREATS TO VALIDITY

To address the *construct validity* concern, we recognize that the evaluation metrics may not fully capture the performance of AI controllers. Therefore, we adopt multiple evaluation metrics that cover various perspectives of the task to measure and analyze the performance of AI-enabled robotics manipulation. In terms of *internal validity*, one potential threat is that the behavior of the AI controller can differ when using different training parameters. To mitigate this threat, we try different sets of parameters and choose the one that results in the best performance in our evaluation. Regarding the *external validity*, we acknowledge that our evaluation results may not be generalized to other robotics manipulation tasks. To address this concern, we include a diverse set of tasks with unique challenges and requirements in our benchmark to consider a wide range of manipulation scenarios.

9 RELATED WORK

Due to the page limit, we focus on presenting the most pertinent related work in this section. For a more detailed review, we kindly direct readers to our website.

Benchmark of CPS and AI-CPS. CPSs are highly integrated systems that collaborate diverse disciplines, e.g., mechanical, electrical, and particularly software engineering, to tackle challenges in real-world applications. Ernst et al. [22] provided a benchmark of traditional CPSs and compared the performance of various testing tools on these systems. However, this benchmark only focuses on traditional CPSs instead of AI-CPSs. Song et al. [55] proposed a first benchmark for AI-CPSs with nine tasks from different domains. However, this benchmark is built on MATLAB, which relies on accurate mathematical models to describe the system behaviors. In addition, other existing literature [9, 19, 33] includes AI-CPSs, but most of them are simplified systems or game scenarios such as *Cart-Pole* and *Inverted Pendulum* that are not appropriate for industrial-level applications.

Benchmark of robotics. In the field of robotics, multiple benchmarks have been proposed for general robotics applications, e.g., [1, 11, 23]. However, these benchmarks often have limitations, as they either focus on providing different robot, object, or sensor models [42, 77] or are designed only for specific topics of robotics control, such as manipulating deformable objects [12] or real-time robotics [5]. Moreover, these benchmarks often ignore important components of the software development lifecycle, e.g. testing. Considering these limitations, a unified benchmark that covers a broad range of tasks and effectively supports the software development lifecycle is desirable. By proposing a benchmark based on Isaac Sim, we aim to address this need and take a first step towards building a development platform for AI-enabled robotics applications.

AI-CPS testing and falsification. Testing is a non-trivial topic in CPSs as it provides the quality assurance to deploy reliable and robust CPSs in safety-critical applications. There is an increasing trend of research devoted in this direction [3, 6, 7, 28, 31, 64, 66–68, 73]. For instance, Zolfagharian et al. [78] proposed a search-based testing approach that leverages a genetic algorithm to generate testing cases for DRL agents, and Stocco et al. [56] delved deeper into the simulation-to-reality gap in the context of autonomous driving vehicles. Falsification [15, 55, 74] is one of the widely used testing techniques to examine the safety of AI-CPS. Zhang et al. [76] leveraged the temporal behaviors of DNN controllers and introduced a falsification framework for AI-CPSs. In [58], a Python-based falsification toolbox for CPSs is presented. However, like other MATLAB falsification tools, it still requires an accurate system model defined as, e.g., ordinary differential equations. Moreover, it is still unclear the effectiveness of existing falsification approaches on modern

physical simulators. Our work develops the first Python-based extendable falsification framework with various optimization methods for Isaac Sim. We believe that our framework can greatly enhance the flexibility of conducting testing for AI-CPS practitioners and motivate further research in this direction.

10 CONCLUSION

By using Isaac Sim, this paper presents a public benchmark of AI-enabled robotics manipulation that includes eight representative manipulation tasks. Multiple AI software controllers are trained with various DRL algorithms, and an evaluation of their performance is conducted. The results show that AI controllers are able to successfully complete the tasks with satisfactory performance and a good level of robustness against action noise. To further test the AI controllers, we also developed the first Python-based falsification framework that is compatible with Isaac Sim. Three different optimization methods are employed to falsify AI controllers in robotics manipulation tasks. The results of the falsification reveal the effectiveness of state-of-the-art falsification techniques in identifying system defects, making them useful for analyzing the reliability of AI controllers. Our work establishes a foundation as well as a systematic pipeline for evaluating and testing AI-enabled robotics systems with modern physical simulators, which is an important step towards understanding and developing trustworthy AI-CPSs for critical real-world domains.

ACKNOWLEDGMENT

REFERENCES

[1] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. 2020. Robel: Robotics benchmarks for learning with low-cost robots. In *Conference on Robot Learning*. PMLR, 1300–1313.

[2] Yashwanth Annpureddy, Che Liu, Georgios Fainekos, and Sriram Sankaranarayanan. 2011. S-TaLiRo: A Tool for Temporal Logic Falsification for Hybrid Systems. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 254–257.

[3] Aitor Arrieta, Maialen Otaegi, Liping Han, Goiuria Sagardui, Shaukat Ali, and Maite Arratibel. 2022. Automating Test Oracle Generation in DevOps for Industrial Elevators. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 284–288.

[4] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine* 34, 6 (2017), 26–38.

[5] Mohammad Bakhshalipour, Maxim Likhachev, and Phillip B Gibbons. 2022. Rtr-bench: A benchmark suite for real-time robotics. In *2022 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 175–186.

[6] Matteo Biagiola and Paolo Tonella. 2022. Testing the plasticity of reinforcement learning-based systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31, 4 (2022), 1–46.

[7] Matteo Biagiola and Paolo Tonella. 2023. Testing of Deep Reinforcement Learning Agents with Surrogate Models. *arXiv preprint arXiv:2305.12751* (2023).

[8] Aude Billard and Danica Kragic. 2019. Trends and challenges in robot manipulation. *Science* 364, 6446 (2019), eaat8414.

[9] David Braganza, Darren M Dawson, Ian D Walker, and Nitendra Nath. 2007. A neural network controller for continuum robots. *IEEE Transactions on Robotics* 23, 6 (2007), 1270–1277.

[10] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540

[11] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. 2015. Benchmarking in manipulation research: Using the Yale-CMU-Berkeley object and model set. *IEEE Robotics & Automation Magazine* 22, 3 (2015), 36–52.

[12] Konstantinos Chatzilygeroudis, Bernardo Fichera, Ilaria Lauzana, Fanjun Bu, Kunpeng Yao, Farshad Khadivar, and Aude Billard. 2020. Benchmark for bimanual robotic manipulation of semi-deformable objects. *IEEE Robotics and Automation Letters* 5, 2 (2020), 2443–2450.

[13] Hong Chen. 2017. Applications of cyber-physical system: a literature review. *Journal of Industrial Integration and Management* 2, 03 (2017), 1750012.

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

[15] Jyotirmoy Deshmukh, Marko Horvat, Xiaoqing Jin, Rupak Majumdar, and Vinayak S Prabhu. 2017. Testing cyber-physical systems through bayesian optimization. *ACM Transactions on Embedded Computing Systems (TECS)* 16, 5s (2017), 1–18.

[16] Nilanjan Dey, Amira S Ashour, Fuqian Shi, Simon James Fong, and João Manuel RS Tavares. 2018. Medical cyber-physical systems: A survey. *Journal of Medical Systems* 42 (2018), 1–13.

[17] Alexandre Donzé. 2010. Breach, a toolbox for verification and parameter synthesis of hybrid systems.. In *CAV*, Vol. 10. Springer, 167–170.

[18] Alexandre Donzé and Oded Maler. 2010. Robust satisfaction of temporal logic over real-valued signals. In *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 92–106.

[19] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. 2016. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*. PMLR, 1329–1338.

[20] Benjamin Ellenberger. 2018–2019. PyBullet Gymperium. https://github.com/benelot/pybullet-gym.

[21] Franka Emika. 2023. Franka Emika Panda. https://www.franka.de/

[22] Gidon Ernst, Paolo Arcaini, Ismail Bennani, Alexandre Donze, Georgios Fainekos, Goran Frehse, Logan Mathesen, Claudio Menghi, Giulia Pedrinelli, Marc Pouzet, et al. 2020. Arch-comp 2020 category report: Falsification. *EPiC Series in Computing* (2020).

[23] Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei. 2018. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In *Conference on Robot Learning*. PMLR, 767–782.

[24] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*. PMLR, 1587–1596.

[25] Fuchang Gao and Lixing Han. 2012. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. *Computational Optimization and Applications* 51, 1 (2012), 259–277.

[26] Claudio Gaz, Marco Cognetti, Alexander Oliva, Paolo Robuffo Giordano, and Alessandro De Luca. 2019. Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization. *IEEE Robotics and Automation Letters* 4, 4 (2019), 4147–4154.

[27] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*. PMLR, 1861–1870.

[28] Liping Han, Shaukat Ali, Tao Yue, Aitor Arrieta, and Maite Arratibel. 2023. Uncertainty-aware Robustness Assessment of Industrial Elevator Systems. *ACM Transactions on Software Engineering and Methodology* 32, 4 (2023), 1–51.

[29] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Florian Golemo, Melissa Mozifian, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. 2020. Perspectives on sim2real transfer for robotics: A summary of the r: Ss 2020 workshop. *arXiv preprint arXiv:2012.03806* (2020).

[30] Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. 2021. Sim2Real in robotics and automation: Applications and challenges. *IEEE transactions on automation science and engineering* 18, 2 (2021), 398–400.

[31] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. 2017. Safety verification of deep neural networks. In *Computer Aided Verification: 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I 30*. Springer, 3–29.

[32] Nasser Jazdi. 2014. Cyber physical systems in the context of Industry 4.0. In *2014 IEEE International Conference on Automation, Quality and Testing Robotics*. IEEE, 1–4.

[33] Taylor T Johnson, Diego Manzanas Lopez, Patrick Musau, Hoang-Dung Tran, Elena Botoeva, Francesco Leofante, Amir Maleki, Chelsea Sidrane, Jiameng Fan, and Chao Huang. 2020. ARCH-COMP20 category report: artificial intelligence and neural network control systems (AINNCS) for continuous and hybrid systems plants. *EPiC Series in Computing* 74 (2020).

[34] Artúr István Károly, Péter Galambos, József Kuti, and Imre J Rudas. 2020. Deep learning in robotics: Survey on model structures and training strategies. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 51, 1 (2020), 266–279.

[35] Shinpei Kato, Eijiro Takeuchi, Yoshio Ishiguro, Yoshiki Ninomiya, Kazuya Takeda, and Tsuyoshi Hamada. 2015. An open approach to autonomous vehicles. *IEEE Micro* 35, 6 (2015), 60–68.

[36] N. Koenig and A. Howard. 2004. Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, Vol. 3. 2149–2154 vol.3. https://doi.org/10.1109/IROS.2004.1389727

[37] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[38] Yongle Luo, Kun Dong, Lili Zhao, Zhiyong Sun, Chao Zhou, and Bo Song. 2020. Balance between efficient and effective learning: Dense2sparse reward shaping for robot manipulation with environment uncertainty. *arXiv preprint arXiv:2003.02740* (2020).

[39] Ruofei Ma, Hsiao-Hwa Chen, Yu-Ren Huang, and Weixiao Meng. 2013. Smart grid communication: Its challenges and opportunities. *IEEE transactions on Smart Grid* 4, 1 (2013), 36–46.

[40] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. 2021. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning. arXiv:2108.10470 [cs.RO]

[41] Matthew T Mason. 2018. Toward robotic manipulation. *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), 1–28.

[42] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, et al. 2023. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters* (2023).

[43] Dejan Ničković and Tomoya Yamaguchi. 2020. RTAMT: Online robustness monitors from STL. In *Automated Technology for Verification and Analysis: 18th International Symposium, ATVA 2020, Hanoi, Vietnam, October 19–23, 2020, Proceedings.* Springer, 564–571.

[44] NVIDIA. 2021. NVIDIA Isaac Sim. https://developer.nvidia.com/isaac-sim

[45] NVIDIA. 2023. NVIDIA Developer Forum. https://forums.developer.nvidia.com

[46] NVIDIA. 2023. Omniverse Isaac Gym Reinforcement Learning Environment. https://github.com/NVIDIA-Omniverse/OmniIsaacGymEnvs

[47] Razvan Andrei Budau Petrea, Massimiliano Bertoni, and Roberto Oboe. 2021. On the interaction force sensing accuracy of franka emika panda robot. In *IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 1–6.

[48] Petar Radanliev, David De Roure, Max Van Kleek, Omar Santos, and Uchenna Ani. 2021. Artificial intelligence in cyber physical systems. *AI & society* 36 (2021), 783–796.

[49] Fabio Ruggiero, Vincenzo Lippiello, and Bruno Siciliano. 2018. Nonprehensile dynamic manipulation: A survey. *IEEE Robotics and Automation Letters* 3, 3 (2018), 1711–1718.

[50] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International Conference on Machine Learning.* PMLR, 1889–1897.

[51] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[52] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. 2018. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), 187–210.

[53] Antonio Serrano-Muñoz, Nestor Arana-Arexolaleiba, Dimitrios Chrysostomou, and Simon Bøgh. 2022. skrl: Modular and Flexible Library for Reinforcement Learning. *arXiv preprint arXiv:2202.03825* (2022).

[54] Jiayang Song, Deyun Lyu, Zhenya Zhang, Zhijie Wang, Tianyi Zhang, and L. Ma. 2021. When Cyber-Physical Systems Meet AI: A Benchmark, an Evaluation, and a Way Forward. *ArXiv* abs/2111.04324 (2021).

[55] Jiayang Song, Deyun Lyu, Zhenya Zhang, Zhijie Wang, Tianyi Zhang, and Lei Ma. 2022. When cyber-physical systems meet AI: a benchmark, an evaluation, and a way forward. In *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice*. 343–352.

[56] Andrea Stocco, Brian Pulfer, and Paolo Tonella. 2022. Mind the gap! a study on the transferability of virtual vs physical-world testing of autonomous driving systems. *IEEE Transactions on Software Engineering* (2022).

[57] Pixar Animation Studio. 2023. Introduction to USD. https://openusd.org/release/index.html

[58] Quinn Thibeault, Jacob Anderson, Aniruddh Chandratre, Giulia Pedrielli, and Georgios Fainekos. 2021. Psy-taliro: A python toolbox for search-based test generation for cyber-physical systems. In *Formal Methods for Industrial Critical Systems: 26th International Conference, FMICS 2021, Paris, France, August 24–26, 2021, Proceedings 26*. Springer, 223–231.

[59] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5026–5033.

[60] Maria Lorena Tuballa and Michael Lochinvar Abundo. 2016. A review of the development of Smart Grid technologies. *Renewable and Sustainable Energy Reviews* 59 (2016), 710–725.

[61] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[62] Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C Karen Liu. 2021. Fast and feature-complete differentiable physics engine for articulated rigid bodies with contact constraints. In *Robotics: Science and Systems*.

[63] Wayne Wolf. 2009. Cyber-physical systems. *Computer* 42, 03 (2009), 88–89.

[64] Weiming Xiang, Hoang-Dung Tran, and Taylor T Johnson. 2018. Output reachable set estimation and verification for multilayer neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 29, 11 (2018), 5777–5783.

[65] Yang Xiang, DY Sun, W Fan, and XG Gong. 1997. Generalized simulated annealing algorithm and its application to the Thomson model. *Physics Letters A* 233, 3 (1997), 216–220.

[66] Xuan Xie, Jiayang Song, Zhehua Zhou, Fuyuan Zhang, and Lei Ma. 2023. Mosaic: Model-based Safety Analysis Framework for AI-enabled Cyber-Physical Systems. *arXiv preprint arXiv:2305.03882* (2023).

[67] Qinghua Xu, Shaukat Ali, and Tao Yue. 2023. Digital Twin-based Anomaly Detection with Curriculum Learning in Cyber-physical Systems. *ACM Transactions on Software Engineering and Methodology* (2023).

[68] Tao Yue, Shaukat Ali, Paolo Arcaini, and Fuyuki Ishikawa. 2022. Towards requirements engineering for digital twins of cyber-physical systems. In *International Symposium on Leveraging Applications of Formal Methods*. Springer, 9–21.

[69] Deniz Yuret, Michael De La Maza, et al. 1993. Dynamic hill climbing: Overcoming the limitations of optimization techniques. In *The second Turkish Symposium on Artificial Intelligence and Neural Networks*. Citeseer, 208–212.

[70] Baohua Zhang, Yuanxin Xie, Jun Zhou, Kai Wang, and Zhen Zhang. 2020. State-of-the-art robotic grippers, grasping and control strategies, as well as their applications in agricultural robots: A review. *Computers and Electronics in Agriculture* 177 (2020), 105694.

[71] Kevin Zhang, Mohit Sharma, Jacky Liang, and Oliver Kroemer. 2020. A modular robotic arm control stack for research: Franka-interface and frankapy. *arXiv preprint arXiv:2011.02398* (2020).

[72] Zhenya Zhang, Paolo Arcaini, and Ichiro Hasuo. 2020. Hybrid system falsification under (in) equality constraints via search space transformation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 11 (2020), 3674–3685.

[73] Zhenya Zhang, Paolo Arcaini, and Xuan Xie. 2022. Online Reset for Signal Temporal Logic Monitoring. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41, 11 (2022), 4421–4432.

[74] Zhenya Zhang, Gidon Ernst, Sean Sedwards, Paolo Arcaini, and Ichiro Hasuo. 2018. Two-layered falsification of hybrid systems guided by monte carlo tree search. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37, 11 (2018), 2894–2905.

[75] Zhenya Zhang, Deyun Lyu, Paolo Arcaini, Lei Ma, Ichiro Hasuo, and Jianjun Zhao. 2021. Effective hybrid system falsification using monte carlo tree search guided by QB-robustness. In *International Conference on Computer Aided Verification*. Springer, 595–618.

[76] Zhenya Zhang, Deyun Lyu, Paolo Arcaini, Lei Ma, Ichiro Hasuo, and Jianjun Zhao. 2022. FalsifAI: Falsification of AI-Enabled Hybrid Control Systems Guided by Time-Aware Coverage Criteria. *IEEE Transactions on Software Engineering* 01 (2022), 1–17.

[77] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. 2020. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293* (2020).

[78] Amirhossein Zolfagharian, Manel Abdellatif, Lionel C Briand, Mojtaba Bagherzadeh, and S Ramesh. 2023. A Search-Based Testing Approach for Deep Reinforcement Learning Agents. *IEEE Transactions on Software Engineering* (2023).