2026 IEEE/SICE International Symposium on System Integration (SII)
January 11-14, 2026. Cancún, México

The Role of Real-World Data in Evaluating Causal Bayesian Networks: Data Collection Guidelines and Case Study

Zhitao Liang[1], Maximilian Diehl[1], Nanami Hashimoto[1], Anne Koepken[2], Daniel Leidner[2], Karinne Ramirez-Amaro[1], and Emmanuel Dean[1]

***Abstract*—Causal Bayesian Networks (CBNs) in robotics are often learned in simulation due to the considerable amount of data required for training. However, discrepancies between simulation and the physical world can cause the learned causal relations to fail in real-world scenarios. Thus, the sim-to-real evaluation is a critical step to deploy a simulation-learned CBN in the real-world. The main challenges in this process are the lack of real-robot evaluation datasets that capture the complexity, noise, and variability of physical environments, which are missing in simulation. In this paper, we propose a set of task-agnostic guidelines for real-robot data collection to evaluate Causal Bayesian Networks (CBNs). The guidelines are generalizable and can be applied to collect real-robot datasets across different robot tasks and platforms. To demonstrate this, we apply them to a robotic platform performing one concrete task, e.g., the robot TIAGo performing a two-cube stacking task, and we collect the real-robot dataset from 100 trials. As a case study, we demonstrate how the dataset can be used to evaluate a simulation-trained CBN on real-robot executions, reporting 10% accuracy drop from sim-to-real transfer. We present this as a first step towards standardized and quantifiable sim-to-real evaluation for CBNs.**

I. Introduction

Simulation environments allow faster, repeatable experiments, safer exploration of diverse conditions, and more scalable, lower-cost data acquisition than physical robots [1]. However, models trained in simulation often face a sim-to-real gap when transferred and deployed to real robots, caused by discrepancies in sensing, actuation, and environmental dynamics between simulation and reality [2].

For methods that rely heavily on data, such as causal learning models, this gap is especially critical. Causal methods, for example, Causal Bayesian Networks (CBNs), can model cause-and-effect relationships between environment features and their effects on task execution success [3]. Thus, they have been utilized to explain failures [4] predict and prevent failures [5]. However, CBN learning methods are data-consuming due to their statistical nature [6], [7]. Most causal learning in robotics is performed in simulation [4], [5], [8], leaving the question open of whether the learned causal structures remain valid under real-world conditions (see Fig. 1).

Therefore, sim-to-real evaluation for CBNs is a crucial step before they can be reliably employed on real robots. To conduct such an evaluation, real-robot datasets must be carefully designed and collected; variables should match simulation definitions and semantics, measurements must be time-aligned, and experiments should span diverse conditions to capture realistic variations and noise. In addition, the dataset design must specify the robot platform needed to perform the same task as in simulation, and the sensor modalities required to observe and measure the corresponding variable values. Despite the strong need, there is currently neither a publicly available real-robot dataset purpose-built for CBN learning and evaluation, nor an established methodology guiding the collection of real-robot causality-structured data and sim-to-real evaluation. Existing manipulation datasets [9]–[11] focus on control or perception benchmarking, but do not provide structured cause-and-effect variables that are suitable for causal model learning. Several prior works on causal learning in robotics [4], [5], [7], [8], [12] have evaluated simulation-trained models in real-robot settings. However, real-robot datasets and detailed descriptions of the sim-to-real setups are not currently available, which makes reproducibility and systematic comparison challenging.

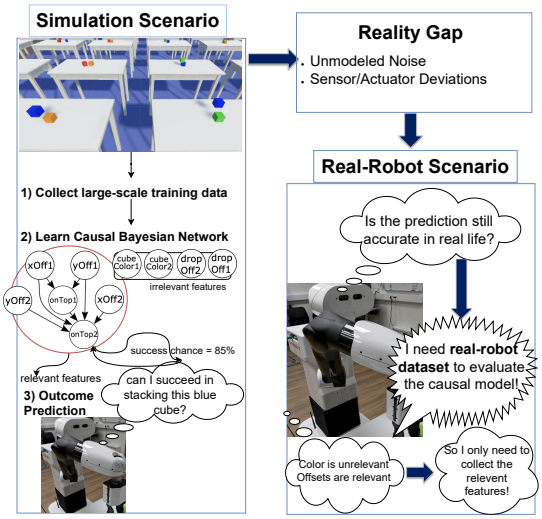


Fig. 1: Simulation-learned CBN requires a real-robot dataset to evaluate whether its learned causal relationships transfer to reality.

To address these research gaps, our contributions in this paper are threefold:

1) We provide a methodology with a set of task-agnostic design guidelines for collecting real-robot data tailored to CBN learning, to guarantee compatibility with simulation data while explicitly capturing noise and variabil-

[1]The authors are with the Faculty of Electrical Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden. Email: {zhitao, diehlm, nanami, karinne, deane}@chalmers.se
[2]The authors are with DLR, Germany. Email: {anne.koepken, daniel.leidner}@dlr.de

ity of real-world operation.
2) Following these guidelines, as a proof-of-concept, we perform the data collection process in a physical robot platform, TIAGo, for one example manipulation task (2-cube stacking). Then, we collect and publish a dataset of 100 real-robot trials.[1]
3) As a case study for the use of the dataset, we apply it in an existing CBN learning pipeline by integrating a sim-to-real evaluation module, demonstrating its practical use.

II. Related work

In this section, we survey existing works that apply causality in robotics with a focus on the datasets in these studies (Table I).

Causality is gaining growing interest in robotics for its ability to uncover causal relations between variables, making it an effective tool to identify task-relevant variables and explain robot behavior. However, a large portion of existing work relies exclusively on simulation data, even though their models are intended to support real-world robot reasoning. For instance, CREST [13] identifies input state variables that influence manipulation policies learned via reinforcement learning through simulated interventions, CAUSALWORLD [14] provides a simulation benchmark that allows controlled causal interventions over environment parameters such as object shape, weight, and appearance to discover causal factors for task success, and ROS-Causal [15] models causal relations among human behaviors, robot behaviors, and human-robot interaction effects within a ROS-based simulation framework.

In addition to purely simulated environments, Uhde et al. [16] incorporate human demonstration data: They inferred causal connections among household activities from 240 virtual demonstrations, but their model was not tested on real robots.

While the above works are limited to simulation-based environments or virtual human demonstrations, Cannizzaro et al. [17] consider modeling real-world noise and uncertainty: They accounted for observation noise and action execution errors when building their causal model, but the noise was simulated and no real-robot data were used.

Other works attempt to incorporate real-world robot executions for training or evaluation. Diehl et al. [4], [5] learned causal models from large-scale simulated data and evaluated their real-world applicability through data collected from physical robots, but the data is not available. Bauer et al. [18] executed 200 real-robot trials to update the probabilistic effect of a robot pick&place action online, and Brawer et al. [12] use 60 human demonstration samples to learn the causal graph underlying tool usage and object manipulation, and 60 real-robot samples for causal intervention and augmentation. However, none of these real-world datasets are publicly available, and a detailed description of the robot setup, real data collection requirements and procedures is missing.

[1]The dataset is released under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. The dataset is intended primarily for research and educational purposes in robotics, causal modeling, and manipulation learning. Users are encouraged to cite this work when using the dataset in published results. Dataset available at https://gitlab.com/craft_lab/causality-robotics/causalrobot_eurobin/eurobin_cubestacking_tiago_dataset.git

III. Preliminaries of Causal Models

In this section, we introduce the existing causal learning process for robot manipulation tasks [5], which served as the foundation for our work presented in this paper.

A. Causal Bayesian Networks (CBNs)

The causal model adopted in this pipeline is the Causal Bayesian Network (CBN). The CBN model is selected because it simultaneously models directed causal relationships and associated probabilistic distributions [3], different from other causal models such as structural equation models [20] and additive noise models [21].

A Causal Bayesian Network is represented by a directed acyclic graph (DAG), $\mathcal{G} = (\mathbf{X}, A)$ [22]. The node set $\mathbf{X}$ comprises N random variables describing the robot and environment states (e.g., object position, grasp success, or target location) in a robot task execution (e.g., pick&place). The arc set A defines the causal relations between the variables. The DAG and Markov property of Bayesian networks factorize the joint probability distribution of $\mathbf{X}$ into a set of local probability distributions, where each random variable X_i depends on its parent variables Π_{X_i}:

$$P(X_1, X_2, ..., X_N) = \prod_{i=1}^{N} P(X_i|\Pi_{X_i}). \quad (1)$$

Learning a Bayesian network from data usually involves two steps: 1) structure learning, where the structure of the DAG (the causal connections between nodes $\mathcal{G} = (\mathbf{X}, A)$) is learned, and 2) parameter estimation, where the local probability distributions $P(X_i|\Pi_{X_i})$ are obtained.

1) Structure Learning: To learn the causal graphical structure of the variables, we employ the PC algorithm [23], which identifies causal connections between variables by applying conditional independence tests. Thus, examining statistical information from discrete data, such as observed frequencies and partial correlations [22]. Some alternative algorithms which could be used with equal eligibility in this step are reviewed in [24].

Many structure learning algorithms, including the PC algorithm, can't learn the causal relations between discrete variables and continuous variables [5]. To ensure that the learned causal graph accurately represents the connections between causes and effects, a preprocessing step is used in this pipeline to discretize all continuous random variables in $\mathbf{X}$ into intervals $\mathbf{X_{int}}$ with an equal number of samples.

TABLE I: Summary of Prior Work on Causal Modeling in Robotics

Used Causal Models	Work	Scenario	Dataset Type & Usage	Public Dataset?
CBN to discover causal connections between environmental/execution features and action outcome	Diehl et al. (2022) [4] [5]	Cube stacking and spheres dropping	Simulated (for CBN learning), Real-Robot (for evaluation)	Yes(sim)
Pyro-based CBN to discover causal connections between environmental/execution features and manipulation effects under uncertainty	Cannizzaro et al. (2024, 2025) [8], [17]	Cube stacking	Simulated (for CBN learning), Real-Robot (5 trials for evaluation)	No
Learn causal graph between execution features and action outcomes for tool affordance	Brawer et al. (2020) [12]	Tool affordance learning	60 human demonstration samples (for structural learning), and 60 real-robot data samples (to infer causal relations from intervention and augmentation)	No
Predict probabilistic effect of robot actions	Bauer et al. (2020) [18]	Pick&Place	200 real-robot trials (to refine the prior effect probabilities online)	No
Infer causal hypotheses among human actions	Uhde et al. (2020) [16]	Household activities	Simulation only, 240 virtual human demonstration samples	Yes (sim) [19]
ROS-Causal, a framework to model causal relations among human behaviors, robot behaviors and interaction outcome	Castri et al. (2024) [15]	HRI scenario (walking)	Simulation only	No
CREST, causal reasoning for robot manipulation policy learning, to determine relevant features for policy reinforcement learning	Lee et al. (2022) [13]	Block stacking and crate opening	No dataset, the causal relations are learned from intervention in simulation	No
Causal structure learning to determine relevant features for policy reinforcement learning	Ahmed et al. (2020) [14]	Building 3d shapes	Simulation-generated tasks with causal variables (causal structures learned from intervention in the simulation)	No

2) Parameter Estimation: Then, parameter learning is performed to estimate the local probability distributions of the factorization in formula 1 (denoted as $\boldsymbol{\theta} = P(X_i|\Pi_{X_i})$). To fit $\boldsymbol{\theta}$ from the data, the Maximum Likelihood Estimation (MLE) is used in the pipeline, estimating all entries in the conditional probability tables $\theta_{x|\boldsymbol{u}}$, $\forall x \in \text{Val}(X_i)$, and $\forall \boldsymbol{u} \in \text{Val}(\Pi_{X_i})$, where $\text{Val}(X_i)$ signifies all possible values of X_i, and $\text{Val}(\Pi_{X_i})$ encompasses all combinations of potential values of Π_{X_i}.

B. Simulation-Based CBN Learning Pipeline

The existing simulation-based CBN learning pipeline from prior work [4], [5] is illustrated in the left part of Figure 1 (blue rectangle). In the process, Simulation data is used to train and evaluate the obtained causal model. A large dataset is generated in a simulation environment (e.g., Unity 3D) by executing task variants with commanded variables. Then, a CBN is learned from simulations. This model is used to predict the success of an action given the current state representation and search for corrective action (variable combination predicted as successful) in case the action is expected to fail.

IV. Guidelines for Real-Robot Dataset Collection

A. Dataset Principles for CBN learning

Datasets for causal models sim-to-real transfer must satisfy the following causal modelling and inference principles: (P1) The dataset should include the task outcome variable and all ancestor variables of it in the DAG. This specification allows the estimation of the corresponding Markov kernels $P(X_i|\Pi_{X_i})$ along all causal paths leading to task outcome using the datasets [3]; (P2) The dataset should include sufficient coverage of parent variable configurations for the estimation of conditional distributions [25]; (P3) Evaluating causal effects requires interventions (i.e., setting cause variables to specific values) to evaluate the effect of causes on outcomes [3]; (P4) CBN variables often abstract dynamic states at specific timepoints. In the dataset, maintaining semantic consistency across trials preserves the meaning of the conditional probability distributions [25]; (P5) The dataset should capture real-world noise and deviations [5].

B. Assumption

Since the causal model is first learned from simulation, we assume that all variables relevant to the task outcomes are defined and observed in the learning process and that there are no unmeasured confounders affecting the learned causal graph. This assumption is reasonable in our sim-to-real setting, as the evaluation focuses on validating whether the simulation-discovered cause-effect relations hold on the real robot, rather than discovering new causal relations. In future work, the assumption could be relaxed by defining measurable metrics to verify the faithfulness of the learned DAG (e.g., measuring the identifiability of causal effects [26]).

C. Task-Agnostic Guidelines

1) **Define relevant cause variables and goal variables based on the causal graph learned from simulation. (R1)** Given the CBN over $\mathbf{X}$ learned from simulation, we extract subsets of variables relevant for evaluation. Specifically, we define cause variables $\mathbf{C} \subseteq \mathbf{X}$ and effect variables $\mathbf{E} \subseteq \mathbf{X}$. Variables outside $\mathbf{C} \cup \mathbf{E}$ are treated as irrelevant for sim-to-real evaluation. The success of an action is specified by a set of goal variables $\mathbf{G} \subseteq \mathbf{E}$ with predefined success conditions $\mathbf{G}_{\text{succ}}$. For example, in a cube-stacking task, a goal variable $\mathbf{G}$ may be `onTop`, with success defined as `onTop = TRUE`.
2) **Assess variable observability and sensor requirements. (R2)** Each variable in $\mathbf{C}, \mathbf{E}, \mathbf{G}$ must be observable from the robot's sensor and perception configuration.
3) **Include diverse data samples and controlled variations. (R3)** To make the CBN realistic about learning the causal relations, the data set should include sufficient variation between the values of the key variables. This can be achieved by systematically commanding different values of the cause variables $\mathbf{C}$ during data collection.
4) **Ensure interventional ability of cause variables. (R3)** To support data diversity, the cause variables $\mathbf{C}$ should be actively manipulable on the real robot, mirroring the interventions possible in simulation. This means that the robot should be able to directly set values of $\mathbf{C}$ independently of other variables, as in the simulation setup.
5) **Capture single-value data with time-aligned observations. (R4)** Rather than using continuous, time-varying sensor streams (e.g., end-effector trajectory), CBNs commonly work on static, single-value variables where each variable has a single value per trial. During real robot execution, these variables are single measurements extracted from raw sensor streams at specific and well-defined moments. For example, instead of storing the whole end-effector trajectory, we should only record the final placement position. To make such single-value observations comparable across trials, the extraction must be time-aligned. The measurement can be triggered by time stamps or sensor signals.
6) **Record real-world noise and execution deviations. (R5)** Since the values of $\mathbf{C}$ are precommanded before execution in the real-robot setup, there might be a deviation between the command and the actual execution. Recording the noisy real-world signals instead of the planned values allows us to evaluate the causal model under realistic conditions.

D. Minimal Robot Platform Requirements

While the guidelines in Section IV-C are task-agnostic, their implementation on a physical robot requires certain hardware and software capabilities. We summarize the minimal requirements below.

- **Degrees of freedom and manipulation ability.** The robot must have sufficient degrees of freedom to perform the actions required by the intended task. For most tabletop manipulation tasks, a 6-DoF arm is sufficient. A reliable end-effector (parallel gripper, suction gripper, or similar) is required to perform grasping and placement consistently.
- **Sensor configuration for observability.** The robot must be equipped with sensors capable of observing the defined variables. For example, RGB-D cameras or fiducial markers (e.g., ArUco) can be used to estimate object poses, while joint encoders and force sensors provide information about robot state and contact events.
- **Interventional control interface.** To enable systematic interventions on cause variables $\mathbf{C}$, the platform must expose interfaces to directly command controllable parameters (e.g., end-effector pose, gripper state). This can be achieved via standard robot control frameworks (ROS, MoveIt) or custom motion controllers, provided that commanded and executed values can be logged.
- **Data logging and synchronization.** The platform must support structured data collection across multiple sensors and subsystems, ensuring that observations are time-aligned. This includes recording both commanded and measured values of $\mathbf{C}$, observed $\mathbf{E}$, and outcome variables $\mathbf{G}$, together with metadata about trial success or failure.

V. Experiments

As a proof-of-concept for the real-robot data collection guidelines, we implement a physical robot experiment in a concrete scenario: a cube-stacking task.

A. Cube-Stacking Environment

As defined in previous work [5], the Cube-Stacking environment consists of three cubes *CubeUp1*, *CubeUp2* and *CubeDown* (see the left part of Fig. 2). The goal is to place *CubeUp2 on top* of *CubeUp1* and *CubeUp1 on top* of *CubeDown*. All cubes have a side length of 5cm. We describe the Cube-Stacking task with the help of ten variables $\mathbf{X} = \{\texttt{xOff1}, \texttt{yOff1}, \texttt{dropOff1}, \texttt{onTop1}, \texttt{cubeColor1}, \texttt{xOff2}, \texttt{yOff2}, \texttt{dropOff2}, \texttt{onTop2}, \texttt{cubeColor2}\}$

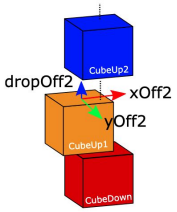
B. CBN Learning in Simulation

Following the simulated CBN learning step III-B, we collect a training dataset of 1,000,000 samples and an evaluation dataset of 80,000 samples. Each data sample takes 6 seconds to generate in the simulation. All task variables $\mathbf{X}$ are sampled according to the right part of Fig. 2 for each 2-Stack experiment in Unity3d simulation environment, where cubes drop according to the sampled parameters without a simulated robot.

Fig. 3 illustrates the graphical structure of the causal model $\mathcal{G} = (\boldsymbol{X}, A)$ learned from the simulation training dataset.

C. Applying the Guidelines in the Real-Robot Setup

1) Define relevant cause variables and goal variables based on simulation-learned causal graph: Based on the obtained causal graph (see Fig. 3), we can define the outcomes $\mathbf{G} = \{\texttt{onTop1}, \texttt{onTop2}\}$, success condition $\mathbf{G}_{\text{succ}} = \{\texttt{onTop1} = 1 \& \texttt{onTop2} = 1\}$ and causes $\mathbf{C} = \{\texttt{xOff1}, \texttt{yOff1}, \texttt{xOff2}, \texttt{yOff2}\}$. We could define



Variable	Meaning
$\texttt{xOff}_2$	$\text{cubeUp}_2.x - \text{cubeUp}_1.x$ (distance between cube centers)
$\texttt{yOff}_2$	$\text{cubeUp}_2.y - \text{cubeUp}_1.y$
$\texttt{dropOff}_2$	$(\text{cubeUp}_2.z - \text{cubeUp}_1.z)$ $-\text{cubeLength}$ (distance between cube surfaces)
$\texttt{onTop}_2$	indicates if cubeUp_2 is on top of cubeUp_1 after the stacking process
$\texttt{cubeColor}_2$	denotes cube color

Variable	Distribution	Range / Notes
`xOff1, xOff2`	$\mathcal{N}(0, 2)$ (in cm)	Truncated to $[-3, 3]$ cm
`yOff1, yOff2`	$\mathcal{N}(0, 2)$ (in cm)	Truncated to $[-3, 3]$ cm
`dropOff1, dropOff2`	$\mathcal{N}(1, 3)$ (in cm)	Truncated to $[0.4, 10]$ cm
`cubeColor1, cubeColor2`	$\mathcal{U}$(Red, Green, Blue, Orange)	Categorical uniform
`onTop1, onTop2`	[TRUE, FALSE]	Automatically determined post-drop

Fig. 2: Left: Defines the causal BN variables for the 2-Stack task (variables that describe the stacking relationship between *CubeUp1* and *CubeDown* are defined analogously) [5]. Right: Distribution of initialization variables in simulation experiments.

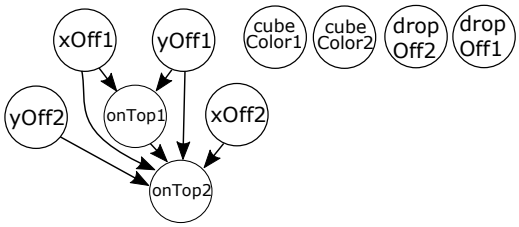


Fig. 3: The obtained causal graph. Directed edges indicate causal dependencies between variables, for example, `onTop1` is causally influenced by `xOff1` and `yOff1`. Conversely, the absence of edges signifies no detected causal relationships; for instance, cube colors were found to have no causal effect on any other variables.

$\{\texttt{dropOff1}, \texttt{dropOff2}, \texttt{cubeColor1}, \texttt{cubeColor2}\}$ as irrelevant features for causal evaluation. We still record `dropOff1` and `dropOff2` in the dataset: While they are not used in our causal evaluation (the simulation-trained model excludes them), they can capture variation in cube placement and may prove useful for future analyses. These variables are therefore included and clearly labeled in the dataset for completeness. For sim-to-real evaluation, they can be optionally omitted to reduce dataset dimensionality.

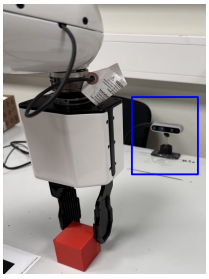


Fig. 4: Positioning of the RealSense camera with respect to the TIAGo robot during data collection.

2) Variable Observability and Sensor Requirements: In this experiment, outcomes `onTop1`, `onTop2` were observed by a human, while the robot has a sufficient perception setup to measure the execution offsets during an action.

Because the robot's onboard camera was often occluded by its own arm during manipulation, we used an external Intel RealSense camera d435i for tracking. This camera was positioned on the opposite side of the table relative to the robot, overlooking the workspace, as shown in Fig. 4. A unique ArUco marker was attached to one face of each cube, allowing cube detection, tracking and pose estimation using RGB images. These settings ensure the variables $\mathbf{X}$ are observable and measurable.

3) Diverse Data Samples and Controlled Variations: During each stack execution, the target stacking positions were determined before moving the robot's arm to each respective goal location (e.g., the position of *CubeDown* was selected before stacking *CubeUp1*). For instance, in Fig. 5 (top row, second image from the left), the stacking position for *CubeUp1* is established before the arm transitions into the pre-stack pose. To ensure a diverse dataset, we vary the commanded stacking offsets for 100 experiment trials, which allows us to include both successful and less optimal stacking scenarios. The full set of commanded stacking positions is presented in the second column of Table II. We selected these commanded positions to cover various stack configurations, including nominal target locations and deliberate positional offsets in different directions and heights. This strategy allowed us to capture both successful stacks and failure cases caused by positioning errors.

4) Interventional Ability of Cause Variables: We use the TIAGo robot, equipped with a 7-DoF arm, an adjustable torso, and a parallel gripper, which is capable of performing the cube stacking task. Motion planning and execution for pick&place actions are conducted using MoveIt!. This setup allows us to stack the cube in a selected gripper position. Before moving to the stacking position, the robot should pre-command the x, y, and z offsets, which requires transforming the measured cube positions from the external camera frame to the robot's frame. To accurately align external camera observations with the robot's kinematic frame, we perform a calibration between the RealSense camera frame and the robot's $base\_link$ frame. This guarantees consistency between detected cube positions and robot motion planning. TIAGo's internal joint encoders and forward kinematics are used to estimate the end-effector pose during the plan and execution of each stacking action.

5) Capture Single-Value, Time-aligned Measurements: In our setup, the high-level motion planner defines a symbolic sequence of actions (e.g., *reach*, *take*, *stack*, *release*) that the robot must execute to complete the stacking tasks. For instance, the *stack* action is predefined in the planner as that the end-effector moves to the target cube-dropping position with the gripper closed to 5cm holding *CubeUp1*, and the *release* action corresponds to opening the gripper to 7cm to drop the holding cube. We use the action sequence to trigger time-aligned measurement. For example, we measure

and record `xOff1`, `yOff1`, `dropOff1` after the first *stack* event is finished and just before the first *release* event is executed (Fig. 5, top row, third image from the left, timestamp t_2). The measurement for `xOff2`, `yOff2`, `dropOff2` is similar (Fig. 5, timestamp t_6).

6) Record Real-World Noise and Execution Deviations: In addition to commanded $\mathbf{C}$, we store the actual measured offsets from the perception system. This allows us to capture discrepancies caused by motion errors, perception noise, or slippage. The deviations between commanded and measured values are shown in Table II.

D. Collected Real-Robot Dataset

Following our proposed guidelines, we successfully collected a real-robot dataset of 100 trials for the cube-stacking task, where each row represents an execution and each column contains a single measured value for one of the variables in $\mathbf{C}$ and $\mathbf{E}$.

The variables distributions show that the stacking offsets `xOff1`, `yOff1`, `xOff2`, `yOff2` span the intended range of variations and show deviations between the commanded and executed positions, as shown in Table II and Fig. 6. This demonstrates that the dataset captures real-world noise arising from actuation errors, perception inaccuracies, and inherent execution variability. Such diversity is critical for causal model evaluation, as it allows the model to encounter realistic variations and learn robust cause-effect relationships.

The guidelines ensure that the data collection and labeling process is largely automated: each trial executes predefined commands, and all variable values are captured by the sensors and extracted automatically.

VI. Case Study: Using the Dataset for Sim-to-Real Evaluation

After collecting the real-robot dataset, we demonstrate its use by performing sim-to-real evaluation of a simulation-trained causal Bayesian Network (CBN). A sim-to-real evaluation module is integrated into the existing CBN learning pipeline (Sec. III-B), as illustrated in Fig. 7.

A. Evaluation Metrics

• **Mean Error:** The **Mean Error** measures the average absolute difference between the conditional probabilities of CBN learned from the simulation-trained data and those estimated from the real-robot dataset. Particularly, it compares the learned parameters $\theta^{\text{train}}_{x|\boldsymbol{u}}$ with their empirical counterparts $\theta^{\text{test}}_{x|\boldsymbol{u}}$ over all configurations of the variable X_i and its parent variables Π_{X_i}. This metric could assess the parameter robustness of a simulation-trained causal model when deployed in real-world conditions.

$$\frac{1}{|\text{Val}(X_i)| \cdot |\Pi_{X_i}|} \sum_{x \in \text{Val}(X_i)} \sum_{\boldsymbol{u} \in \Pi_{X_i}} \left| \theta^{\text{test}}_{x|\boldsymbol{u}} - \theta^{\text{train}}_{x|\boldsymbol{u}} \right| \tag{2}$$

• **Accuracy, Precision, Recall, F1-Score:** These metrics are used to assess the predictive performance of the simulation-trained model in real-robot practice. The ground-truth outcome is given by the binary success variables G, while the model provides a probability distribution $P(\mathbf{G}_{\text{succ}}|\mathbf{C})$ for each data sample in the test set. Thus, a threshold $\epsilon = 0.5$ should be applied to the predicted probability of a successful outcome. A prediction is considered successful (positive) if it exceeds this threshold, resulting in a binary classification. These binary classifications are then compared between the training data and the test data to obtain accuracy, precision, recall, and the F1 score. The existing pipeline also uses Accuracy to evaluate the predictive performance.

B. Obtained Causal Model

To select the optimal number of discretization intervals for the continuous variables $\{$`xOff1`, `yOff1`, `dropOff1`, `xOff2`, `yOff2`, `dropOff2`$\}$, we tested the number of bins as a hyperparameter by performing a 10-fold cross-validation over a range of candidate values. Based on the mean validation performance, we chose to use 7 equal-width discretization intervals within the value ranges (See the right part of Fig. 2).

Subsequently, we assessed the model's performance on the separate simulation evaluation dataset (Section V-A), using the metrics defined at Section VI-A. The resulting performances are presented in the second column of Table III. A failure prediction accuracy of 0.9425 is achieved.

C. Sim-to-Real Evaluation

To evaluate how well the causal model reflects the real-world scenarios, we assessed the obtained causal model on the 100 real-world samples (Sec. V-D) using the same metrics (Section VI-A) to evaluate the probability distribution difference and the predictive performance. During the prediction, the stacking offsets recorded from the real-world experiments were discretized using the same discretization intervals learned during the training.

The sim-to-real performance of the cube-stacking task (Tab. III, third column) shows a mean error of 0.18 between the probability distribution from real-robot experiments and the causal model learned in simulation, corresponding to a sim-to-real accuracy of 0.82, which shows a 12% decrease from the simulated evaluation result. To the best of our knowledge, while prior work has focused on causal learning in either simulation or real settings, the average sim-to-real performance decrease has not been quantified in the literature. We attribute the observed sim-to-real performance difference primarily to real-world perception noise (e.g., camera-to-robot calibration noise), small mechanical deviations in end-effector pose, and unmodeled contact dynamics during cube placement. Mitigation strategies include (i) injecting structured noise during simulation training, and (ii) using a small amount of real data for parameter fine-tuning.

VII. Discussion & Conclusions

In response to the lack of real-robot datasets for causal learning, we propose six task-agnostic guidelines for generating structured real-robot datasets tailored to causal learning and capturing real-world variability and noise. Following

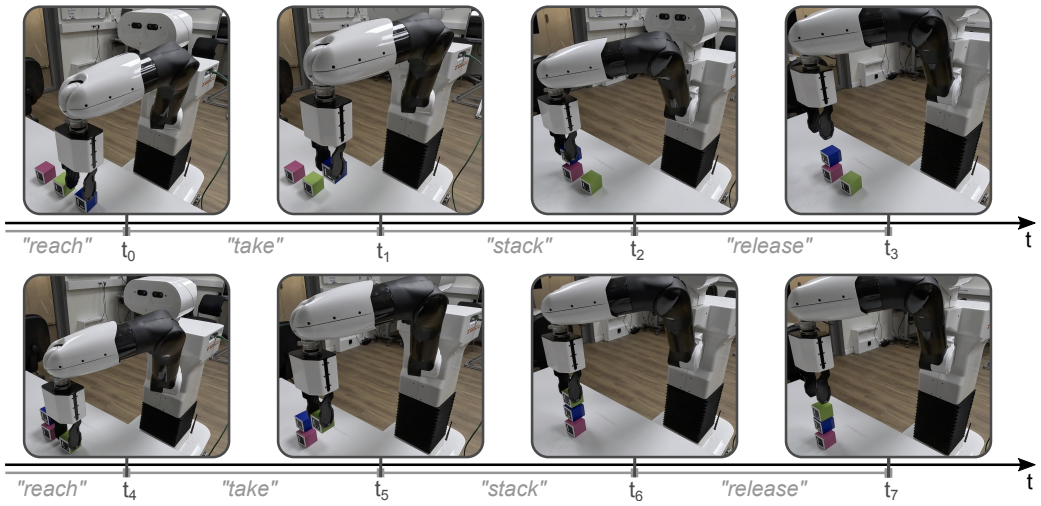


Fig. 5: The stacking process performed by the TIAGo robot.

TABLE II: Summary statistics (mean ± SD) for each trial group.

Trials	Commanded	xOff1 (cm)	yOff1 (cm)	dropOff1 (cm)	xOff2 (cm)	yOff2 (cm)	dropOff2 (cm)
1–5	(0, 0, 0.5), (0, 0, 0.5)	-0.503 ± 0.229	-0.249 ± 0.134	0.586 ± 0.051	-0.858 ± 0.329	-0.269 ± 0.104	0.592 ± 0.065
6–10	(1, 0, 0.5), (1, 0, 0.5)	-0.530 ± 0.389	-0.164 ± 0.058	0.625 ± 0.088	-0.436 ± 0.495	-0.143 ± 0.074	0.590 ± 0.040
11–15	(2, 2, 0.5), (2, 2, 0.5)	1.56 ± 0.427	1.80 ± 0.012	0.670 ± 0.100	1.83 ± 0.157	1.84 ± 0.034	0.591 ± 0.051
16–20	(1, -1, 0.5), (1, -1, 0.5)	0.570 ± 0.302	-1.41 ± 0.070	0.610 ± 0.044	0.370 ± 0.257	-1.43 ± 0.100	0.595 ± 0.070
21–30	(1.5, -1.5, 0.5), (-1.5, 1.5, 0.5)	0.583 ± 0.425	-1.79 ± 0.068	0.638 ± 0.081	-2.32 ± 0.568	1.31 ± 0.102	0.601 ± 0.044
31–40	(0, -1.5, 0.5), (0, 0, 0.5)	-0.968 ± 0.622	-2.00 ± 0.068	0.655 ± 0.067	-0.743 ± 0.328	-0.459 ± 0.058	0.596 ± 0.060
41–45	(0, 1.0, 0.5), (0, 1.0, 0.5)	-0.807 ± 0.583	0.854 ± 0.319	0.651 ± 0.073	-0.430 ± 0.325	0.916 ± 0.337	0.577 ± 0.048
46–50	(0, 0, 1.0), (0, 0, 1.0)	-0.254 ± 0.367	-0.440 ± 0.040	1.11 ± 0.037	-0.422 ± 0.263	-0.440 ± 0.055	1.09 ± 0.061
51–55	(1, 0, 1.0), (1, 0, 1.0)	0.540 ± 0.251	-0.514 ± 0.045	1.11 ± 0.046	0.834 ± 0.166	-0.482 ± 0.041	1.03 ± 0.068
56–60	(2, 2, 1.0), (2, 2, 1.0)	1.72 ± 0.313	1.54 ± 0.031	1.11 ± 0.029	1.81 ± 0.311	1.71 ± 0.110	1.01 ± 0.046
61–65	(1, -1, 1.0), (1, -1, 1.0)	0.921 ± 0.361	-1.44 ± 0.091	1.05 ± 0.057	0.690 ± 0.240	-1.51 ± 0.092	1.09 ± 0.081
66–75	(1.5, -1.5, 1.0), (-1.5, 1.5, 1.0)	1.21 ± 0.309	-2.07 ± 0.042	0.949 ± 0.061	-1.84 ± 0.253	0.924 ± 0.042	1.20 ± 0.052
76–85	(0, -1.5, 1.0), (0, 0, 1.0)	-0.384 ± 0.262	-1.97 ± 0.039	1.08 ± 0.078	-0.533 ± 0.210	-0.471 ± 0.038	1.12 ± 0.056
86–90	(0, -1.0, 1.0), (0, -1.0, 1.0)	-0.373 ± 0.256	-1.38 ± 0.163	1.06 ± 0.082	-0.380 ± 0.173	-1.39 ± 0.132	1.08 ± 0.073
91–95	(0, -1.5, 0.5), (1.0, -1.0, 0.5)	-0.48 ± 0.385	-1.91 ± 0.114	0.649 ± 0.0406	0.649 ± 0.488	-1.41 ± 0.0649	0.53 ± 0.0474
96–100	(0, 1.5, 0.5), (1.0, 1.0, 0.5)	-0.437 ± 0.395	1.20 ± 0.0305	0.679 ± 0.0518	0.586 ± 0.126	0.705 ± 0.0504	0.595 ± 0.0890

Note: All values are in centimeters. Commanded offsets represent the target position of the object stack as $(\texttt{xOff1}, \texttt{yOff1}, \texttt{dropOff1}), (\texttt{xOff2}, \texttt{yOff2}, \texttt{dropOff2})$. Due to sensor and motor inaccuracies, the actual positions were different from the originally commanded ones.

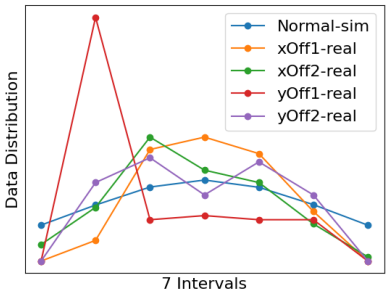


Fig. 6: Distribution of stacking offsets in the collected real-robot dataset compared to the simulation. Each variable is discretized into 7 equal-width intervals. Real-robot measurements capture execution variability and noise, while the simulation assumes an idealized normal distribution for all variables (blue line).

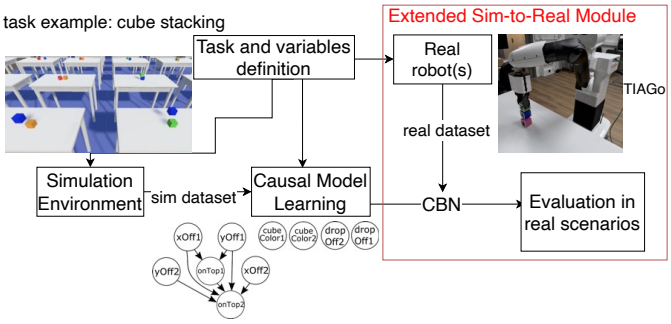


Fig. 7: The pipeline for collecting and utilizing robotic data to learn and evaluate Causal Bayesian Networks in task-agnostic settings.

these guidelines, we collected and released a real-robot dataset for a cube-stacking task, providing a concrete example of the methodology in practice. The approach is generalizable and can be applied to other platforms and causal tasks.

As a case study, we applied the dataset to perform sim-to-real evaluation of a simulation-trained CBN. The experimental results indicate that the dataset accommodates a simulation-trained CBN and enables the assessment of CBN transfer from simulation to reality, with quantitative evidence of potential sim-to-real performance. The measured performance drop from simulation to real-world trials underlines the importance of capturing real-world noise and variability,

TABLE III: Failure Prediction Results (Averaged over All Samples) for Simulation and Real Robot.

Metric	Simulation	Real Robot
Mean Error	0.04	0.18
Accuracy	0.94	0.82
Precision	0.93	0.97
Recall	0.95	0.79
F1 Score	0.94	0.89

confirming that the guidelines effectively produce a dataset that exposes CBN to realistic variations. As a potential future extension, the quantitative sim-to-real performance evaluation can allow researchers to utilize a causal model in failure explanation and prevention in real-robot execution by predicting the success chance of a given policy in future work. For example, if the predicted success probability of the current action is low, the model can identify alternative actions (feature combinations) with higher predicted success. Comparing these combinations yields an explanation of potential failure, while executing the counterfactual action with a higher success chance prevents it. While our results suggest that the dataset accommodates the needs of CBNs and supports sim-to-real evaluation, the dataset's limited size and single-task scenario mean that broader generalization to other models and tasks remains to be verified in the future work.

Overall, our work provides a first step toward the standardization of real-robot datasets for causal learning. By explicitly linking dataset principles, data collection guidelines, robot platform requirements and experimental validation, we demonstrate a reproducible methodology that other researchers can adopt, test, and refine. The case study illustrates the value of such datasets in supporting quantitative sim-to-real evaluation, opening a discussion on norms and best practices for creating structured datasets tailored to causal models in robotics.

ACKNOWLEDGMENT

This work was supported by Chalmers Gender Initiative for Excellence (Genie) and co-financed by the European Union's Horizon Europe grant euRobin (GA no. 101070596).

REFERENCES

[1] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[2] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8973–8979.

[3] J. Pearl, *Causality: Models, Reasoning and Inference*, 2nd ed. USA: Cambridge University Press, 2009.

[4] M. Diehl and K. Ramirez-Amaro, "Why Did I Fail? A Causal-Based Method to Find Explanations for Robot Failures," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8925–8932, Oct. 2022.

[5] ——, "A causal-based approach to explain, predict and prevent failures in robotic tasks," *Robotics and Autonomous Systems*, vol. 162, p. 104376, 2023.

[6] H. Wang, I. Rish, and S. Ma, "Using sensitivity analysis for selective parameter update in bayesian network learning," *Assoc. Adv. Artif. Intell.*, 01 2002.

[7] M. Diehl and K. Ramirez-Amaro, "Generating and transferring priors for causal bayesian network parameter estimation in robotic tasks," *IEEE Robotics and Automation Letters*, 2024.

[8] R. Cannizzaro, M. Groom, J. Routley, R. O. Ness, and L. Kunze, "COBRA-PPM: A Causal Bayesian Reasoning Architecture Using Probabilistic Programming for Robot Manipulation Under Uncertainty," Jun. 2025, arXiv:2403.14488 [cs]. [Online]. Available: http://arxiv.org/abs/2403.14488

[9] A. Hundt, V. Jain, C.-H. Lin, C. Paxton, and G. D. Hager, "The costar block stacking dataset: Learning with workspace constraints," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1797–1804.

[10] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay *et al.*, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," in *Conference on Robot Learning*. PMLR, 2018, pp. 879–893.

[11] A. Khazatsky *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," *arXiv preprint arXiv:2403.12945*, 2024.

[12] J. Brawer, M. Qin, and B. Scassellati, "A causal approach to tool affordance learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 8394–8399.

[13] T. E. Lee, J. A. Zhao, A. S. Sawhney, S. Girdhar, and O. Kroemer, "Causal reasoning in simulation for structure and transfer learning of robot manipulation policies," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4776–4782.

[14] O. Ahmed, F. Träuble, A. Goyal, A. Neitz, Y. Bengio, B. Schölkopf, M. Wüthrich, and S. Bauer, "Causalworld: A robotic manipulation benchmark for causal structure and transfer learning," *arXiv preprint arXiv:2010.04296*, 2020.

[15] L. Castri, G. Beraldo, S. Mghames, M. Hanheide, N. Bellotto *et al.*, "Ros-causal: A ros-based causal analysis framework for human-robot interaction applications," in *Workshop on Causal Learning for Human-Robot Interaction (Causal-HRI), ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2024.

[16] C. Uhde, N. Berberich, K. Ramirez-Amaro, and G. Cheng, "The robot as scientist: Using mental simulation to test causal hypotheses extracted from human activities in virtual reality," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 8081–8086.

[17] R. Cannizzaro, M. Groom, J. Routley, R. O. Ness, and L. Kunze, "Physics-based causal reasoning for safe & robust next-best action selection in robot manipulation tasks." *CoRR*, 2024.

[18] A. S. Bauer, P. Schmaus, F. Stulp, and D. Leidner, "Probabilistic effect prediction through semantic augmentation and physical simulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9278–9284.

[19] K. Ramirez-Amaro, C. Uhde, T. Bates, and G. Cheng, "A benchmarking dataset for automatic symbolic grounding from virtual demonstrations," in *2nd International Workshop on Computational Models of Affordance in Robotics (ICRA)*, 2019.

[20] K. A. Bollen, *Structural Equation Models with Observed Variables*, 1989, ch. Four, pp. 80–150. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118619179.ch4

[21] J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf, "Causal discovery with continuous additive noise models," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2009–2053, 2014.

[22] M. Scutari, "Learning bayesian networks with the bnlearn r package," *Journal of Statistical Software*, vol. 35, no. 3, p. 1–22, 2010. [Online]. Available: https://www.jstatsoft.org/index.php/jss/article/view/v035i03

[23] D. Colombo and M. H. Maathuis, "Order-independent constraint-based causal structure learning," *J. Mach. Learn. Res.*, vol. 15, no. 1, p. 3741–3782, Jan. 2014.

[24] M. J. Vowels, N. C. Camgoz, and R. Bowden, "D'ya like dags? a survey on structure learning and causal discovery," *ACM Comput. Surv.*, Nov. 2022. [Online]. Available: https://doi.org/10.1145/3527154

[25] J. Peters, D. Janzing, and B. Schölkopf, *Elements of causal inference: foundations and learning algorithms*. The MIT press, 2017.

[26] D. Kong, S. Yang, and L. Wang, "Identifiability of causal effects with multiple causes and a binary outcome," *Biometrika*, vol. 109, no. 1, pp. 265–272, 2022.