Causal Relations Using a Simulated Robotic Arm: Research Report

Group Members: Zeynep Sude İleri Serdar Doğukan Uysal Metehan Şamat Kerem Coşanay

Github: https://github.com/zeynepsudeileri/pybullet\_project

# Introduction

This report details a research project focused on causal learning in robotics, utilizing a simulated 7-degreeof-freedom (DoF) KUKA LBR iiwa robotic arm to model low-level causal relationships in sensorimotor tasks. Drawing inspiration from human causal cognition and Judea Pearl’s work on graphical causal inference, the project aims to enhance the robustness of artificial intelligence (AI) systems by modeling mechanistic relationships at the lowest level of the robotic state space. The study focuses on two categories of robotic causal cognition: sensorimotor self-learning (C1) and learning the consequences of actions on objects (C2).

## Objectives

* Develop and evaluate forward and inverse models to learn causal relations from synthetic datagenerated via motor babbling.
* Investigate sensorimotor self-learning and object interactions in a simulated environment.
* Extract explainable knowledge about environmental behavior using feature attribution methods tosupport dimensionality reduction and causal analysis.
* Lay the foundation for future action planning using causal models.

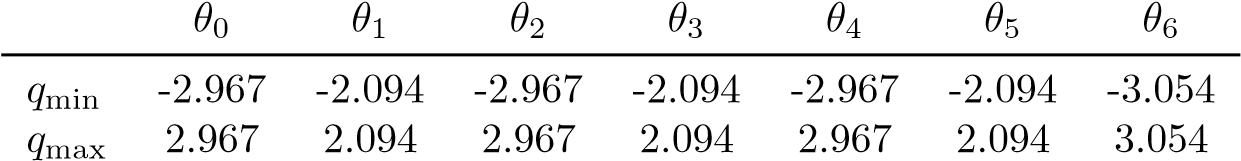
# Methodology

## Data Generation

Data was generated using the myGym toolkit in a simulated environment with a 7-DoF KUKA LBR iiwa robotic arm:

* **Motor Babbling (C1)**: The arm executed 500,000 random actions, sampling joint configurations from a normal distribution within the joint limits specified in Table 1. Each action was executed in 10 substeps, recording joint configurations (θ(*t*) ∈ R7) and Cartesian end-effector positions (ef(*t*) = [*efx,efy,efz*]), forming the state vector s(*t*) = [θ(*t*)*,*ef(*t*)].
* **Object Interaction (C2)**: The arm interacted with a magnetized cube over 4,000 episodes, each lasting 500 iterations. The arm randomly switched the magnet, picked up the cube, maneuvered it, and released it. The state vector was expanded to include object features (position, rotation, color) and magnet state: s(*t*) = [o(*t*)*,*θ(*t*)*,*ef(*t*)*,mgt*(*t*)], where o(*t*) = [*ox,oy,oz,orx,ory,orz,oR,oG,oB*].

Table 1: Joint motion range of KUKA LBR iiwa arm (in radians).



## Model Architectures

Two types of models were developed to capture causal relationships: Forward Models (FM) and Inverse Models (IM), implemented as feed-forward neural networks. The architectures were designed to handle high-dimensional state and action spaces, with specific adaptations for the tasks in Experiments 1 and 2, as described in the referenced paper.

**2.2.1 Forward Model (FM)**

The FM predicts the next state sˆ(*t* + 1) from the current state s(*t*) and action a(*t*), defined as:

FM : [s(*t*)*,*a(*t*)] 7→ sˆ(*t* + 1)*,*

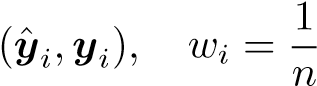
where a(*t*) = θ(*t* + 1) − θ(*t*) represents biologically plausible motor commands. For Experiment 1, s(*t*) = [θ(*t*)*,*ef(*t*)]; for Experiment 2, s(*t*) = [o(*t*)*,*θ(*t*)*,*ef(*t*)*,mgt*(*t*)].

* **Architecture Details**:
  + **Input Layer**: Concatenates s(*t*) and a(*t*). For Experiment 1, the input is 17-dimensional (7 joint angles, 3 end-effector coordinates, 7 action components). For Experiment 2, the input is 27-dimensional (9 object features, 7 joint angles, 6 end-effector pose components, 1 magnet state, 8 action components).
  + **Hidden Layers**: Three fully connected layers (e.g., 256, 128, 64 neurons) with tanh activation functions to capture non-linear relationships. Dropout (rate 0.2) is applied to prevent overfitting.
  + **Output Layer**: Uses separate output heads for state subvectors:

∗ *Experiment 1*: Two heads: joint configuration (θ(*t* + 1), 7 outputs) and end-effector position (ef(*t* + 1), 3 outputs).

∗ *Experiment 2*: Five heads: object position (3 outputs), object rotation (3 outputs), object color (3 outputs), joint configuration (7 outputs), end-effector pose (6 outputs), magnet state (1 output). Each head uses linear activation.

* + **Loss Function**: Mean Squared Error (MSE) computed for each output head:

Loss = ∑*wi* · MSE*,*

*i*

where y*i* are subvectors of s(*t* + 1), and *n* is the number of heads (2 for Experiment 1, 5 for Experiment 2).

* + **Training Process**:

∗ *Experiment 1*: Trained for 60 epochs using Adam optimizer (*η* = 10−3) on 500,000 samples, with 5-fold cross-validation. Inputs and outputs are normalized based on joint limits (Table 1).

∗ *Experiment 2*: Trained for 100 epochs using Adam (*η* = 10−3) on 4,000 episodes.

* + **Mental Simulation**: Supports chained inference:

sˆ(*t*) = FM[sˆ(*t* − 1)*,*a(*t* − 1)]*,* sˆ(0) = s(0)*,*

predicting states up to 10 steps ahead.

* **Advantages**:
  + Multi-head architecture enhances accuracy and interpretability.
  + Enables mental simulation with linear error growth for θ(*t*) and semi-logarithmic for ef(*t*).
  + Scalable to complex tasks by adding output heads.
* **Challenges**:
  + Balancing loss weights across heads requires careful tuning.
  + Assumes reliable state perception, potentially problematic in real-world settings. **–** High-dimensional inputs in Experiment 2 increase computational demands.
* **Implementation Notes**: Implemented in PyTorch or TensorFlow, leveraging GPU acceleration.

The architecture is depicted in the paper’s Figure 1.

**2.2.2 Inverse Model (IM)**

The IM predicts the action aˆ(*t*) to transition from s(*t*) to s(*t* + 1):

IM : [s(*t*)*,*s(*t* + 1)] 7→ aˆ(*t*)*.*

Two approaches address the unavailability of θ(*t*+1) during inference: Monolithic and Pre-Computation.

* **Monolithic Approach**:
  + **Architecture Details**:

- **Input Layer**: Concatenates s(*t*) and s′(*t* + 1) = s(*t* + 1) \ θ(*t* + 1). For Experiment 1, 13-dimensional (10 for s(*t*), 3 for ef(*t* + 1)). For Experiment 2, includes object and magnet features.

- **Hidden Layers**: Three fully connected layers (256, 128, 64 neurons) with tanh activations, dropout (0.2).

- **Output Layer**: Produces a(*t*) (7 outputs for Experiment 1, 8 for Experiment 2) with linear activation.

- **Loss Function**: MSE, computed separately for joint and magnet actions in Experiment

2.

- **Training Process**: Trained for 1,000 epochs using AdamW (*η* = 10−3*,λ* = 0*.*004) for Experiment 1 (MAE 0.0529 rad), and Experiment 2 (MAE 0.0077 rad for joints,

4*.*56 × 10−4 for magnet).

* **Advantages**:

- Simpler architecture (∼100,000 parameters), computationally efficient.

- Robust to missing θ(*t* + 1) during inference.

* **Challenges**:

-Limited incorporation of kinematic priors.

- Performance may degrade with complex state spaces.

* **Pre-Computation Approach**:
  + **Architecture Details**:

**-Pre-Network**: Estimates θˆ(*t* + 1) from [s(*t*)*,*s′(*t* + 1)]. Two hidden layers (128, 64 neurons, tanh) output 7 joint angles. Trained for 4,000 epochs using AdamW (*η* = 10−3*,λ* = 0*.*004), MAE 0.0481 rad.

- **Base Model**: Takes [s(*t*)*,*s′(*t* + 1)*,*θˆ(*t* + 1)], with three hidden layers (256, 128, 64 neurons, tanh) and outputs a(*t*). Trained for 100 epochs using Adam (*η* = 10−3).

- **Loss Function**: MSE for pre-network (joint prediction) and base model (action prediction), trained jointly.

**-Training Process**: Pre-network trained first, then integrated with base model (∼150,000 parameters).

* + **Advantages**:

- Incorporates kinematic priors, improving accuracy for complex tasks. ∗ Modular design allows reuse of pre-network.

* + **Challenges**:

-Higher computational cost.

- Requires synchronized training of pre-network and base model.

* **Comparison**:
  + **Accuracy**: Pre-computation slightly outperforms (MAE 0.0481 rad vs. 0.0529 rad in Experiment 1), but differences are minimal in Experiment 2.
  + **Complexity**: Monolithic is lighter; pre-computation suits precise kinematic tasks.
  + **Use Case**: Monolithic for simplicity, pre-computation for complex transitions.

# Experiments

## Experiment 1: Learning Kinematics

* **Setup**: The KUKA arm performed 500,000 motor babbling steps, recording joint angles and endeffector positions.
* **Results**: The FM achieved MAE of 6.9 mm for ef(*t*) and 3*.*4×10−3 rad for θ(*t*), enabling mental simulation up to 10 steps with linear error growth for θ(*t*) and semi-logarithmic for ef(*t*). The IM (monolithic) achieved MAE of 0.0529 rad; pre-computation at 0.0481 rad.

## Experiment 2: Learning Simple Intuitive Physics

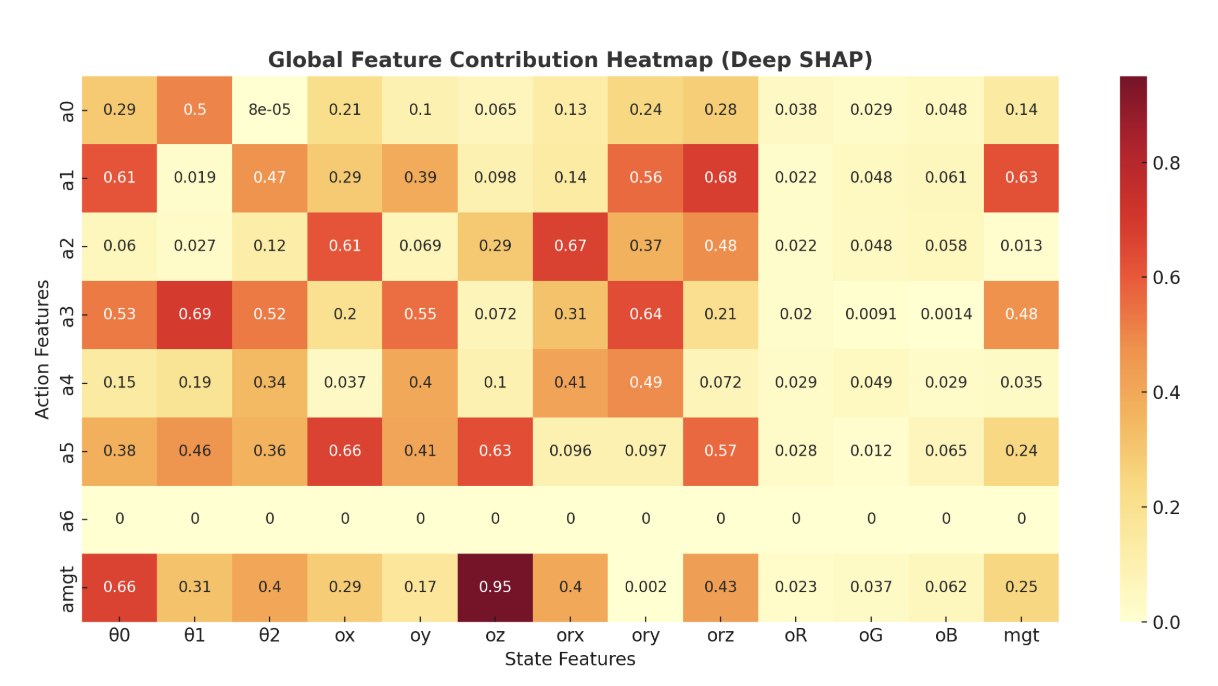
* **Task**: The arm manipulated a magnetized cube over 4,000 episodes, learning object kinematics and interaction effects.
* **Model Performance**: The FM achieved MAEs of 0.0089 m (object position), 0.0721 rad (object rotation), 0.004 (object color), 0.0084 rad (joint configuration), 0.008 m (effector position), 0.0625 rad (effector rotation), and 1*.*3×10−4 (magnet state). The monolithic IM achieved MAEs of 0.0077 rad (joints) and 4*.*56 × 10−4 (magnet).

metin, çizgi, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Knowledge Extraction and Explainability

1. **Deep SHAP Analysis**: Applied to the FM using 200 samples, revealing feature importance. Key findings:
   * Joint 6 (*a*6) was unused, suggesting dimensionality reduction.
   * Object color (*oR,oG,oB*) was irrelevant to actions.
   * Magnet action (*amgt*) significantly affected object position (*oz*).
2. **Partial Dependence Plots (PDPs)**: Visualized correlations. For example, *a*0 strongly impacted *θ*0 and *ox*, while *a*2 had negligible effect on *oB*.



### metin, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, çizgi içeren bir resim Açıklama otomatik olarak oluşturuldu

### **5. Results and Achievements**

#### **5.1 Experiment 1: Learning Kinematics**

In this experiment, the robotic system was subjected to a large-scale motor babbling protocol, totaling 500,000 interaction steps. The Forward Model (FM) was trained to predict both joint configurations and Cartesian end-effector positions from these data. The model achieved a **Mean Absolute Error (MAE)** of **6.9 mm** for the effector position and **3.4 × 10⁻³ rad** for the joint configuration, indicating high precision in low-level kinematic prediction.To evaluate temporal robustness, the FM was tested using **mental simulation**, a technique where the model recursively predicts future states. It showed:

* **Linear growth** in joint configuration error across 10 prediction steps, suggesting stable extrapolation of joint dynamics.
* A **semi-logarithmic trend** in effector position error, which, while increasing, did so at a manageable rate.

The Inverse Model (IM), trained to estimate the required actions for transitioning between two consecutive states, was evaluated in two configurations:

* **Monolithic approach** achieved an MAE of **0.0529 rad**.
* **Pre-computation approach** slightly improved performance with an MAE of **0.0481 rad**, suggesting modest benefits from incorporating intermediate joint estimations.

These results validate the feasibility of learning forward and inverse mappings in continuous robotic control tasks, and confirm the FM's capability as a predictive model for simulation-based planning.

#### **5.2 Experiment 2: Learning Simple Intuitive Physics**

This experiment extended the system's capability by introducing object-level interactions. The robot manipulated a magnetized cube across **4,000 episodes**, allowing it to learn the causal effects of its actions on external objects.The FM was restructured with multiple output heads to accommodate a larger state space, including object position, rotation, color, effector pose, joint angles, and magnet state. Despite the increased dimensionality, the model delivered precise predictions:

* **0.0089 m MAE** for object position.
* **0.0721 rad MAE** for object rotation.
* **0.008 m** and **0.0625 rad MAE** for effector position and rotation respectively.
* **0.0084 rad** for joint configuration.
* **1.3 × 10⁻⁴** for magnet state prediction.

The monolithic IM achieved:

* **0.0077 rad MAE** for joint action prediction.
* **4.56 × 10⁻⁴ MAE** for magnet action, reflecting high reliability in discrete control tasks.

These results demonstrate the model's ability to generalize from motor control to **interaction dynamics**, effectively capturing simple causal laws of motion and contact.

### **6. Conclusions**

This research establishes a robust framework for **low-level causal learning** in robotic systems. Through extensive experimentation with both isolated kinematics and object interactions, several core conclusions can be drawn:

1. **Effectiveness of Causal Models**: The developed Forward and Inverse Models reliably learned the relationships between robot actions and environmental state transitions. The FM enabled accurate mental simulations, while the IM served as a viable tool for inverse reasoning and control synthesis.
2. **Scalability and Modularity**: By using modular output heads and dual model approaches (monolithic and pre-computation), the architecture demonstrated flexibility across varied tasks—scaling from 10 to over 30 input/output features without loss of performance.
3. **Explainability and Knowledge Extraction**: Feature attribution techniques, particularly Deep SHAP and Partial Dependence Plots (PDPs), provided actionable insights:
   * Identified **irrelevant features** (e.g., object color, unused joints) that could be pruned for efficiency.
   * Quantified the **causal impact** of actions on specific state changes, supporting a move toward interpretable and explainable AI (XAI) in robotics.
4. **Simulation as a Ground for Reasoning**: The system was able to simulate the outcomes of sequences of actions—a crucial ability for high-level planning, scenario testing, and reinforcement learning.
5. **Foundation for Future Work**: The methodologies developed in this project set the stage for higher-level robotic cognition, especially in the areas of planning, skill transfer, and real-world deployment.

# Work Table

Zeynep Sude İleri- IM-Deep SHAP-Simulation-Monte Carlo

Kerem Coşanay- IM- Simulation

Serdar Doğukan Uysal- FM-Motor babbling- Simulation

Metehan Şamat- FM -Simulation

**References:**

Cibula, M., Kerzel, M., & Farkaš, I. (2024). Learning low-level causal relations using a simulated robotic arm. arXiv. <https://doi.org/10.48550/arXiv.2410.07751>