CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY

LICENTIATE THESIS

MAXIMILIAN DIEHL • Explainable and Interpretable Decision-Making for Robotic Tasks • 2022

Explainable and Interpretable Decision-Making for Robotic Tasks

MAXIMILIAN DIEHL

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY

THESIS FOR THE DEGREE OF LICENTIATE OF ENGINEERING

Explainable and Interpretable Decision-Making for Robotic Tasks

Maximilian Diehl

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2022

Explainable and Interpretable Decision-Making for Robotic Tasks

Maximilian Diehl

This thesis has been prepared using LaTeX.

Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg, Sweden
Phone: +46 (0)31 772 1000
www.chalmers.se

Printed by Chalmers Digitaltryck
Gothenburg, Sweden, December2022

To my family and friends!

Abstract

Future generations of robots, such as service robots that support humans with household tasks, will be a pervasive part of our daily lives. The human's ability to understand the decision-making process of robots is thereby considered to be crucial for establishing trust-based and efficient interactions between humans and robots. In this thesis, we present several interpretable and explainable decision-making methods that aim to improve the human's understanding of a robot's actions, with a particular focus on the explanation of why robot failures were committed.

In this thesis, we consider different types of failures, such as task recognition errors and task execution failures. Our first goal is an interpretable approach to learning from human demonstrations (LfD), which is essential for robots to learn new tasks without the time-consuming trial-and-error learning process. Our proposed method deals with the challenge of transferring human demonstrations to robots by an automated generation of symbolic planning operators based on interpretable decision trees. Our second goal is the prediction, explanation, and prevention of robot task execution failures based on causal models of the environment. Our contribution towards the second goal is a causal-based method that finds contrastive explanations for robot execution failures, which enables robots to predict, explain and prevent even timely shifted action failures (e.g., the current action was successful but will negatively affect the success of future actions). Since learning causal models is data-intensive, our final goal is to improve the data efficiency by utilizing prior experience. This investigation aims to help robots learn causal models faster, enabling them to provide failure explanations at the cost of fewer action execution experiments.

In the future, we will work on scaling up the presented methods to generalize to more complex, human-centered applications.

Keywords: Failure explanation, Explainability, Interpretability, Causality

List of Publications

This thesis is based on the following publications (numbered in chronological order):

[A] **Diehl Maximilian**, Paxton Chris, Ramirez-Amaro Karinne, "Automated Generation of Robotic Planning Domains from Observations". *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, Online, Oct. 2021.

[B] **Diehl Maximilian**, Ramirez-Amaro Karinne, "Why did I fail? A Causal-based Method to Find Explanations for Robot Failures". *IEEE Robotics and Automation Letters*, vol. 7, no. 4, Oct. 2022.

[C] **Diehl Maximilian**, Ramirez-Amaro Karinne, "A Causal-based Approach to Explain, Predict and Prevent Failures in Robotic Tasks". Conditionally accepted with minor revisions to *Robotics and Autonomous Systems (RAS)*, Elsevier, 2022.

[D] **Diehl Maximilian**, Ramirez-Amaro Karinne, "Transferable Priors for Bayesian Network Parameter Estimation in Robotic Tasks". Submitted to *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

Other publications by the author, not included in this thesis, are:

[E] **Diehl Maximilian**, Ramirez-Amaro Karinne, "Augmented Reality Interface to Verify Robot Learning". *Proc. 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, Naples, Italy, Online, 2020.

[F] **Diehl Maximilian**, Paxton Chris, Ramirez-Amaro Karinne, "Optimizing robot planning domains to reduce search time for long-horizon planning". *5th Workshop on Semantic Policy and Action Representations for Autonomous Robots (SPAR) at IROS 2021*, Prague, Czech Republic, Online, 2021.

[G] **Diehl Maximilian**, Ramirez-Amaro Karinne, "Work in Progress - Automated Generation of Robotic Planning Domains from Observations". *18th International conference on Ubiquitious Robots, Organized Session - Robots in the household: A review of task knowledge acquisition, planning and execution*, Gangwon-do, South Korea, Online, 2021.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Prof. Karinne Ramirez-Amaro. Thank you for always having my back and the constant encouragement, which has been vital to building up my confidence and belief in myself. Thank you for all the hours of discussion and feedback, which helped me improve significantly over the last three years. And finally, thank you for always pushing me to give my most, without which I would not have achieved what I have so far.

Acronyms

AI:	Artificial Intelligence
AP:	Automated Planning
AR:	Augmented Reality
BFS:	Breadth-First Search
BN:	Bayesian Network
LfD:	Learning from Demonstration
MAP:	Maximum a Posteriori
MLE:	Maximum Likelihood Estimation
PDDL:	Planning Domain Definition Language
RL:	Reinforcement Learning
VR:	Virtual Reality
XAI:	Explainable Artificial Intelligence

Contents

Part I

Overview

CHAPTER 1

Introduction

Future generations of robots will be a pervasive part of our daily lives [1], [2]. Service robots are envisioned to help humans in our homes with daily activities such as folding laundry [3] or washing the dishes [4], transporting laboratory specimens in hospitals [5], [6], or even providing care for the elderly [7] and rehabilitation for stroke patients [8]. There is also an increasing interest in collaborative robots in assembly lines and workshops that should support human workers in strenuous tasks with negative consequences on the workers' health [9], [10]. To establish trust-based and efficient interactions between humans and robots, one of the emerging challenges is the human's ability to understand the decision-making process of the robots. Robots have different physiology and move differently than humans, which makes it more difficult to interpret the robots' intentions [11]. For that reason, the field of explainable artificial intelligence (XAI) has experienced a significant boost of interest over the last few years [2], [12].

Two essential keywords in XAI are interpretability and explainability. Interpretable systems generate decisions based on humanly understandable rules, whereas explainability refers to explicit explanations and justifications of the decisions and actions [13]. These objectives are important for various reasons:

it has been shown that humans tend to assume that robots have mental states and, therefore, naturally try to understand the rationale for the robot's actions from a human perspective [2]. If the robots' actions are not explainable, there could be an incoherence between the human's explanation and the internal stance of the robot. This incoherence raises the risk of self-deception, may degrade the interaction quality, and, in the worst case, the user's safety could be at risk [2]. Furthermore, robots that autonomously perform complex tasks in unstructured environments such as homes or hospitals are prone to mistakes, just as any human being. However, humans have the capability to reflect upon mistakes and reason about what went wrong, which helps them to learn and improve in the future. Moreover, humans can communicate and justify their actions. That is a crucial part of human interactions and is particularly important when failures have occured [13], [14]. Explainability was shown to be fundamental to fostering trust, acceptance, and motivation to engage with robots [15]–[18], whereas misunderstanding the intentions of robots creates discomfort and confusion [2]. Finally, robots that have failed might need the assistance of everyday users to recover, which is difficult without an understanding of why the robot has failed [19]–[21]. The objectives of interpretability and explainability are of particular importance in the light that most of the people envisioned to interact with robots are not robot experts but everyday users such as the elderly, nurses, or simply people at home [19].

In this thesis, we present interpretable and explainable learning and decision-making methods that should accommodate different types of robot failures. In the beginning, we present an interpretable approach to learning from human demonstration (LfD). LfD is considered to be an essential tool for autonomous robots to continuously extend their capabilities by learning new tasks when required [22]. Moreover, interpretability is crucial since we have a human in the loop (as a teacher and demonstrator of the task). In the second part of this thesis, we focus on robot execution failures. We present methods that allow robots to predict, explain and prevent execution failures.

1.1 Research goals and contributions

Enabling humans to better understand the robots' decision-making process and actions led to our objective to investigate methods that allow for **explainable and interpretable decision-making for robotic tasks**. In the following paragraphs, I derive a series of goals and introduce our contributions to solving these challenges.

Goal 1: Life-long learning (or learning from demonstration)

Robots that are deployed in environments such as homes are likely to encounter situations that require them to learn new tasks, such as setting the table or cleaning the kitchen. Thus, my first goal is to provide robots with the functionality to learn how to achieve these tasks, while being able to connect and reuse previous experiences to the newly collected knowledge. This knowledge should be shareable among robots with different embodiments, and it should be possible to flexibly apply this knowledge in different situations. There are multiple ways to learn, but LfD is essential for humans of all ages to acquire new tasks without the time-consuming process of trial-and-error learning [22]. We, therefore, aim to provide robots with similar learning capabilities. Since we have a human teacher in the loop, it will be crucial for humans to understand what the robot has learned from the demonstrations, which motivates the objective of using interpretable learning methods. We, therefore, formulate the following research question:

RQ1: How can a robot learn a task in a flexible, robot-agnostic, and interpretable manner by observing a human demonstration of the task?

Goal 2: Explanation of a robot's actions

Explaining one's actions is one of the most fundamental interactions between humans [23]. This ability is particularly crucial when we encounter failures and plays a crucial role in achieving trust, acceptance, and efficient interactions with robots. Explaining failures is typically linked to the concept of causality. Specifically challenging are situations where a current action is considered successful but could negatively affect future actions, which we refer to as timely shifted action effects. Therefore another goal of this thesis is to collect causal knowledge from the environment with the purpose of understanding and explaining robot task failures. This goal motivated the formulation of the following research questions:

RQ2: How can we reliably detect cause-effect relationships in robot tasks involving timely shifted and erroneous action effects?

RQ3: How can robots predict, explain and prevent failures?

RQ4: How can robots transfer knowledge in the form of causal models from one task to another?

Methods and contributions

Semantic-based method for automated generation of robotic planning domains from human demonstrations

End-to-end learning from human demonstration approaches are typically not robot agnostic or robust to unexpected events during the task execution [24] (e.g., the carried object drops out of the robot gripper). Instead, our approach is, therefore, to break down a task demonstration into several sub-steps and infer the intention behind each step by observing the user's hand activities. Each detected action is then collected in the form of planning operators. Planning operators describe under which circumstances the action can be successfully executed and what the resulting outcome will look like. Then, based on the collection of these planning operators, the robot can generate high-level, symbolic action plans, which will guide it to solve the demonstrated tasks. The collection of planning operators can be continuously expanded and used to plan for novel tasks which have not been demonstrated. Moreover, since the description of these planning operators is based on symbols, the actions and generated action plans are human-understandable. For example, one condition for being able to execute the action *Take* might be *objectIsGraspable*.

- Our main contribution is to associate the timely segmented and classified hand activities from the human demonstrations with changes in the environment and capture the obtained actions in the form of high-level planning operators. This method is presented in Paper A.

Causal-based method for explaining and preventing robot failures

Papers B and C discuss our approach to explaining and preventing robot failures. In Paper B, we present our novel causal-based method to find contrastive explanations of robot execution failures. We demonstrate how causal Bayesian networks can be learned from simulations, exemplified in a cube-stacking and sphere-dropping scenario, and provide real-world experiments that show that the obtained causal models are transferable from simulation to reality without any retraining. Our method is agnostic to various robot platforms with different embodiments and scales over various tasks and scenarios. We, thus, show that the simulation-based model serves as an excellent prior experience for the explanations, making them more generally applicable. The causal model allows robots to predict how likely an action will succeed, however, it does not provide explanations for why the failure has occurred.

- Our main contribution of Paper B is a method that finds contrastive failure explanation upon task failure. The explanation is based on setting the failure state in contrast with the closest state that would have allowed for successful execution. This state is found through breadth-first search and is based on success predictions from the learned causal model. For example, after failing to stack a tower of cubes, the robot explains that it failed because “the upper cube was stacked too high and too far to the right of the lower cube”.

In Paper C, we then introduce an extension to Paper B that utilizes these prediction capabilities to find corrective actions which will allow the robot to prevent failures from happening.

- Our main contribution of Paper C is an algorithm that proposes a solution to the complex challenge of timely shifted action effects, which are cases where the current action on its own cannot be considered as a failure, but nevertheless might have negative consequences on later actions and the overall task goal.

For example, when building a tower of several cubes, offsets in between the lower cubes can have detrimental effects on the overall stability of the tower. By detecting causal links over the history of several actions, the robot can effectively predict and prevent failures, even if the root of a failure lies in a previous action.

One of the constraints of Papers B and C is the data efficiency of learning larger causal models. The larger the number of causally relevant parent variables, the more data is required to learn the causal structure and conditional probability distributions of the variables.

- Our main contribution in Paper D is a method that constructs parameter priors from previous experience and transfers it to related but different tasks. We conduct a detailed comparison between learning a parameter model from scratch and learning from a prior, mainly with respect to data efficiency.

We test and compare the outcome of the two estimation methods for the use-case of failure prevention from paper B. A special focus is thereby set on cases where no or only a few data samples of the new task are available.

1.2 Research journey

All the papers presented in this thesis are built on top of each other. To guide the reader through my research journey so far, I will attempt a short elaboration of my thinking process throughout that time.

I began my research journey when I started to explore Augmented Reality (AR) as means to communicate the robot's state of mind for the purpose of human-robot interaction. AR, e.g., in the form of Head-Mounted-Displays like the Hololens, allows the superimposition of virtual objects over the real environment and thus opens up many possibilities to visualize information in the environment. However, virtual objects can obviously not physically interact with the environment, making it more challenging to detect potential errors during the robot task execution, e.g., collisions with the environment (unless they are also virtually enhanced). The particular use-case we investigated was a learning-from-demonstration scenario, where we assume that the user has already taught the robot a new skill, e.g., opening a drawer. However, before executing the task on the real robot, the user would like to verify if the robot has learned the correct behavior. We, therefore, conducted an experiment to investigate the capability of humans to detect robot execution failures in robot simulations performed in AR. We compared the different visualization modes of 1) AR on Hololens, 2) AR on a tablet, and 3) RVIz-like simulation on a tablet [25] (Paper E). The users generally enjoyed the experience and possibilities that AR was able to provide. However, solely showing the robot task executions was not enough for the users to reliably detect failures since they often looked at the wrong locations (e.g., focused on the gripper, even though a collision happened between the robot arm and its base). Even though Paper E is not further discussed in this thesis, we drew inspiration to conduct research in two directions: 1) We wanted to close the gap between high-level action learning and low-level execution, and 2) we wanted to investigate other methods which would allow the robot to explain and prevent failures itself.

Following that, I started to work on Paper A, which deals with the automated generation of robotic planning domains from human demonstrations, with the goal of narrowing the gap between high-level task learning and low-level task execution. During this investigation, I was very curious about the causality aspect of the demonstrations. In our proposed framework, we keep track of a variety of categorical variables, some of them describing the environment, and some of them describing the hand activity. The challenge we faced was how to causally connect hand activities with changes in the environment. In other words, which hand activities causes which effect on the environment? We, humans, are naturally capable of understanding even complex causal chains. Imagine the following example: a human picks up a bottle of water from the table with his right hand while reaching towards the bottle with the left hand, with the intention to eventually unscrew the bottle. During this process, the `onTop` predicate, describing the relationship between the bottle and the table, turns from true to false. Both hands perform an activity that involves the bottle, but nevertheless, for the human, it is instantly clear that it was the right hand that was responsible for this change since it had the bottle `inHand`. The robot, however, first needs to learn these causal relations. For paper A, we have the assumption that there are no overlapping hand activities and no external environment changes, such that we can assign the environment changes to the currently executed hand activity. However, already simple examples such as the one mentioned above would break this assumption. Another problem are timely shifted action effects. In such cases, an effect cannot be ascribed to the currently executed task but, in fact, was caused by a prior action. Imagine a situation where the robot would stack cubes but does not place them centered on top of each other. After the third cube, the tower falls. The robot should be able to understand that the issue was with the positioning of the first cube, which was badly placed.

Detecting causal-effect relations in human demonstrations led me to investigate the statistical/mathematical framework of causality as introduced by Judea Pearl [23]. This opened up the possibility of learning causal relations from experiments in an environment; however, it requires a lot of data. For that reason, we collected data from simulations. Interestingly, causality also allowed us to reconsider the problem of failure detection and explanation from Paper E. However, instead of requiring a human operator to verify the learned task, the robot is able to explain potential failures itself or might even be able to prevent them from happening in the first place. Additionally, causality allows us to tackle the problems of timely shifted action effects and, in the future, potentially other problems, like environment changes that have been caused by a third party or even learning relevant operator predicates. These problems are investigated in papers B and C. While simulations are a great tool to obtain a large number of data necessary for these kinds of statistical learning methods, there will always be a sim-to-real gap. However, human demonstrations or physical robot experiments are expensive. Therefore we have investigated new ways of transferring prior knowledge about similar tasks in Paper D, e.g., from dropping a sphere into a plate to dropping a sphere into a bowl or from stacking one cube to stacking two cubes.

1.3 Thesis outline

Chapter 2: Starts with a short introduction to automated planning, which is a preliminary for understanding the concept of planning operators. Then, the chapter gives a brief overview of our contribution in Paper A, which is the automated generation of robotic planning domains from human demonstrations.
Chapter 3: Starts with a background section on failure explanations as well as causality and causal (Bayesian) Network learning. Then, our causal approach to explaining Robot action failures (Paper B) and prediction and prevention of robotic failures (Paper C) are summarized. Finally, we give a short summary of our results of paper D, which deals with transferring prior knowledge for Bayesian Network parameter estimation.
Chapter 4: Provides a brief summary of all the papers that are included in this thesis.
Chapter 5: Discusses future work.

CHAPTER 2

Automatic generation of robot planning operators from human demonstrations

When robots are instructed to complete complex and long horizon tasks such as setting the table, they need to execute a long chain of actions, e.g., navigate to the kitchen, open the drawer and pick up plates. Some important information, however, might not be available at the task execution's start, leading to occasional action failures. For example, the robot might be unaware that the plates that are usually stored inside the kitchen drawer are currently in the dishwasher. Once the robot does not find the plates in the kitchen drawer, it needs to quickly adapt its plan and look for an alternative plate location, such as the dishwasher. Automated planning (AP), which is also known as Artificial Intelligence (AI) Planning, plays an essential role in achieving this type of deliberation of autonomous robots [26] by providing the tools to generate action plans to reach the desired goal or replan when unexpected situations occur. More formally, AP is described as "the study of computational models and methods of creating, analyzing, managing and executing plans" [27]. In the following sections, we introduce and illustrate the most important AP concepts based on a classic planning example. Then the reader will be led through the complete pipeline of planning task definition and formalization.

One prerequisite for the application of AI planning is the definition of a list of all possible actions that can be used by the agent (e.g., robot) to achieve its goal. These actions are typically referred to as planning operators and need to be manually defined, which is a time-intensive process that often requires a task-domain expert. Therefore, we introduce our method to automatically generate planning operators from human demonstrations from Paper A at the end of this chapter. The human demonstrations are recognized based on interpretable decision trees. Furthermore, a big advantage of AP is that actions are typically defined in terms of human-understandable symbols that describe each action's preconditions and effects, which positively affects the interpretability of the whole task plan and its execution.

2.1 Automated Planning (AP)

The goal of AP is to solve *planning problems*, which formally refers to obtaining a sequence of transformations for moving a system from an initial state to a goal state, given a description of possible transformations [28]. Planning problems are solved by utilizing problem-solving techniques like heuristic search or propositional satisfiability to find an (optimal) sequence of actions to reach a desired objective [27]. Such techniques are referred to as *planning algorithms*. However, as a prerequisite to applying planning algorithms, all possible world states transformations (actions) need to be defined. States and actions are defined in the so-called `planning domain` [29], which is formally defined as a triple $\Sigma = (S, A, \gamma)$ or a 4-tuple $\Sigma = (S, A, \gamma, cost)$, where S is a finite set of *states*, A is a finite set of actions, $\gamma : S \times A$ a partial function called the *state-transition function* and $cost : S \times A \rightarrow [0, \inf)$ the cost function. Then a planning algorithm solves Π to produce a plan $\pi =< a_1, a_2, ..., a_n >$ that transforms the current state $I \in S$ of the agent to its goal $G \in S$. In the remainder of the thesis, we refer to a particular combination of current state I and goal state G as a *planning task*.

The world state S is formally defined in terms of a set of state variables or predicates that can either be true or false. The set of Actions A of the planning domain is provided in terms of planning operators O. Operators are blueprints of actions that, if applicable, change the world state in a specific way. Each planning operator has an associated `name`, a set of `objects` constituting its arguments, a set of `preconditions` governing what must be true about the

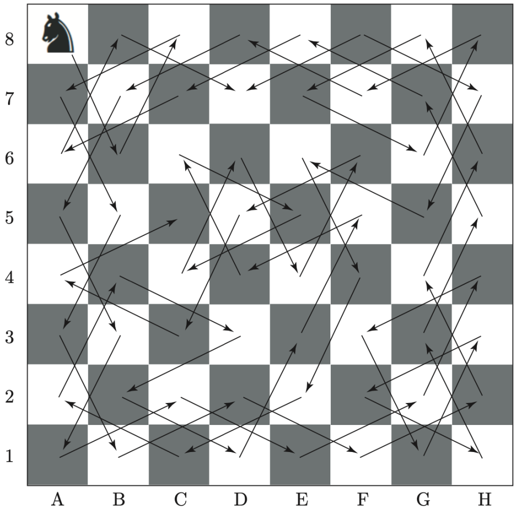


Figure 2.1: Illustrates the Knight's tour puzzle. The objective is to find a sequence of actions to move the knight to any desired location on the field. Source: [27]

world for the operator to be used, and a set of `effects` describing how the world will change after we use this operator [30].

A classical demonstrative example for AP is the *Knight's tour* puzzle (visualized in Fig. 2.1). The objective of this game is to move the knight to any desired location on the checkboard. However, the knight can only move in a particular way: either two squares in any direction vertically followed by one square horizontally or two squares in any direction horizontally followed by one square vertically. Finding the optimal sequence of moves is non-trivial for humans but can be easily solved with the help of AI planning. First, it requires the definition of an adequate set of world-states (e.g., the position of the knight), a set of actions (e.g., moving the knight from the current field two squares horizontally and one square vertically), and a formalization that allows plugging this definition into a planning algorithm. The planning algorithm is then responsible for finding the optimal sequence of actions that will solve the game.

Formalization of a planning problem - PDDL

In order to facilitate the development of planning algorithms, the planning community has developed a de-facto standard for formalizing planning tasks, which is called PDDL (Planning Domain Definition Language). Initially, PDDL was only able to express planning in its purest form: actions are modeled with preconditions and positive/negative effects, expressed as sets of atomic facts. This is commonly referred to as STRIPS but has been gradually expanded with new functionality like numeric and temporal planning. It is important to note that there are other ways of formalizing a planning task, e.g., as a Markov Decision Process (MDP) or Hierarchical Task Networks (HTN). However, since we view the operator generation process from human demonstrations as a classical planning problem (discrete, deterministic, and finite), we use PDDL.

We now show how the Knight's tour puzzle can be formalized in PDDL. First we take a look at the `domain` specification [27]:

```
(define (domain knights-tour)
    (:requirements :negative-preconditions)

    (:predicates
        (at ?square)
        (visited ?square)
        (valid_move ?square_from ?square_to)
    )

    (:action move
     :parameters (?from ?to)
     :preconditions (and (at ?from)
                         (valid_move ?from ?to)
                         (not (visited ?to)))
     :effect (and (not (at ?from))
                  (at ?to)
                  (visited ?to))
    )
)
```

In the knight-tour example, the world state S is defined in terms of the predicates `at(?square)`, which describes the current position of the knight and takes a square as an argument, `visited(?square)`, which describes which square has been visited, and `valid_move(?square_from ?square_to)`, which denotes if a move from *?square_from* to *?square_to* is valid. For example, `valid_move(A1 A2) = False`, but `valid_move(A1 C2) = True`. We only defined one operator with the name `move`, which has two squares, *?from* and *?to* as input arguments. It can only be used on a pair of two locations *?from* and *?to*, when the knight is currently located at the *?from* location, it is a valid move for the knight to jump from *?from* to *?to* location, and *?to* has not been visited yet. As a result of applying the `move` operator, the knight is located at the new location *?to* and *?to* is added to the list of visited locations.

Note that `domain` does not contain any information about the current position of the knight or the goal, nor any specific instances of objects that are contained in our environment. These things are defined in a separate `problem` specification:

```
(define (problem knights-tour-problem-8x8)
    (:domain knights-tour)

    (:objects
        A1 A2 A3 A4 A5 A6 A7 A8
        B1 B2 B3 B4 B5 B6 B7 B8
        ...
        H1 H2 H3 H4 H5 H6 H7 H8
    )

    (:init
     ; The Knight's starting square is arbitrary
     ; here, we have
     ; chosen the upper right corner.
     (at A8)
     (visited A8)
     ; We have to list all valid moves:
     (valid_move A8 B6)
     (valid_move B6 A8)
     (valid_move A8 C7)
     (valid_move C7 A8)
     (valid_move B8 A6)
     (valid_move A6 B8)
     (valid_move B8 C6)
     ...
    )

    (:goal
        (and (visited A1)
             (visited A2)
             ...
             (visited H8))
    )
)
```

All objects that could play a role in finding the solution for the particular planning task are defined in `:objects`. In this example, we need to define all the possible square instances (locations) that the knight can move to. Note that in order to generalize the definition of predicates and actions, these instances were just denoted as *?square* in the `domain` file. In `:init`, the current state is defined in terms of predicates, applied to the object instances (squares like A1). Finally, the `:goal` denotes the goal configuration of that particular planning task, again in terms of predicates (similar to init). Note that one only needs to define one `domain` per planning environment but needs to define multiple `problem` files in case of different initial- or goal configurations.

Automated planning in robotics

From the beginnings of AI, AP has played an essential role in achieving the deliberation of autonomous robots [26]. The STRIPS planning system, for example, was pioneered in the early 70s for use on the robot Shakey [31], but since then found its way into countless robotic applications, like autonomous spacecrafts [32], exploration and rescue robots [33] or autonomous areal vehicles (AUVs) [34]. Recently planning has also been used for collaborative robots in an assembly line [35], leveraging the advantages of replanning in case of unexpected situations or plan changes, and with ROSPlan [36] an interface between PDDL and plan execution through ROS (Robot Operating System) has been developed.

The core principle of planning is the abstraction of lower-level task execution descriptions (e.g., joint trajectories or sensor readings) into a set of high-level actions. It has been shown that such an abstraction is particularly advantageous for complex and long-horizon tasks, even in continuous real-world-domains [37]. For example, the task of setting the table. Even humans break down such a long task into individual actions that are executed one after another. Other popular methods, like Reinforcement Learning, will have difficulties in learning such tasks end-to-end [38]. Another advantage is that symbolic, high-level plans are shareable among different robots as long as the operators do not contain any robot-specific predicates. Due to the high-level, symbolic descriptions of planning operators, which are human-understandable, also the resulting plans are interpretable by humans. Finally, planning domains can be easily adapted, or additional knowledge can be continuously added, and it is possible to use the operators even in different tasks. Therefore, we believe that automated planning is the ideal tool for our goal of learning/extracting knowledge from human demonstrations in a flexible and robot-agnostic way.

2.2 Paper A: Our approach for automated generation of planning operators

As discussed in the previous sections, using planning operators to store, continuously learn and reuse experience for new tasks is advantageous. However, from the planning perspective, a major bottleneck is the requirement of an accurate description of the planning task in the form of the planning domain. Generating large domains by hand is very time-consuming or even infeasible and often requires domain experts who are also knowledgeable of the PDDL language or an alternative system to formulate the planning constraints [39]. For example, suppose we are in a household situation where the robot should be able to learn from any member of the family. In such a case, we could not expect them to formulate the knowledge in the form of planning operators. This motivated us to develop a method for automatically generating these operators from human demonstrations.

Unsurprisingly, we are not the first to discover the need for automated tools for constructing planning domains. In fact, the planning community dedicated the field of Knowledge Engineering to the acquisition and formulation of planning knowledge, with the domain model being the desired output [40]. Various approaches like LOCM2 [41] or AMAN [42] try to generate operators from plan traces. However, plan traces are usually constructed from benchmark domains, like the ones defined by the International Planning Competition (IPC). In other words, plan traces are generated from domains, which are then recovered. They do not consider traces from real-world demonstrations, e.g., human task demonstrations, and the resulting challenges of detecting human actions and timewise segmenting the observation.

There are also some works from the robotics community that aim to learn planning operators. Many approaches do this directly on their respective robotic platforms. In [43], a framework is presented that learns probabilistic action effects. As input, the framework requires the motor commands associated, for example, with the stacking activity, and possible outcomes are simulated in an environment with different objects like cubes and spheres. In another approach [37], state representations and operators are learned while the robot is exploring its environment. In [44], and [45], the technique of kinesthetic teaching is used to generate operators for skills like reaching, grasping, pushing, and pulling. Recent work also deals with the question of how to abstract the essence behind lower-level actions [21], [37], [46]. In [46], a parameterized model of a pushing activity is learned, which relates the applied force to the final block location after sliding. The authors of [21] abstract tasks from the sensorimotor level to state variables like `inFrontOf`, `behind`, or `above`. The task of opening a drawer and a door is learned based on these spatial and temporal features. The biggest difference is that our method learns from human demonstrations. Additionally, by utilizing our activity recognition framework [47] we can derive semantic operator descriptions and, most importantly, meaningful segmentation of the observed task rather than considering every state change as a potential new operator.

Challenges and contributions

Several challenges must be addressed to generate a useful planning domain. First, human demonstrations are costly compared to simulations, meaning we typically can only collect a handful of demonstration samples per task. As a result, it is not easy to get any statistically significant data, which in turn does not allow the use of any statistical learning methods. We, therefore, need to assume that every recognized action within a task demonstration was correct and led to the expected predicate transitions, despite the possibility of noisy or erroneous demonstrations. For this reason, we introduce an operator cost that prioritizes more commonly observed operators during plan generation. Another issue is that we cannot look behind the motor commands of a human in a similar fashion as we can do with robots. In other words, the only way of comparing two human actions is by investigating the preconditions and effects of the actions. If they have the exact same set of preconditions or effects, we can assume the actions to be similar; otherwise, we need to assume that they are two different actions, which will result in the creation of two different planning operators. One consequence is that we stick to so-called *classical planning*, when we automatically generate planning operators from observations. Classical planning is constrained to planning tasks with a finite set of deterministic actions and a static world. The latter means that changes to the world state can not be caused by external forces but only by actions defined in the planning domain. The third problem is that the demonstration environments can contain many objects or relations that are irrelevant to a specific activity. For example, when picking up a specific `Cube_green`, the other cubes have no bearing on the execution of the action, which should reflect in the predicate selection for preconditions and effects of the generated operator. Lastly, operators should generalize activities performed with a specific hand or objects (e.g., `Left_hand`, `Cube_blue`) to types (e.g., *Hand*, *Wooden_cube*). Another advantage of our system is that the collection of operators can continuously grow with new demonstrations. As a result, the planning system was able to create plans for unseen goals in our experiments. While demonstrations only covered the stacking of one or two cubes in a row, our system was able to create plans for more complex tasks such as stacking four cubes or two separate towers.

To summarize, our main contributions of Paper A are:

- High-level operator generation from noisy demonstrations, including the omission of irrelevant preconditions/effects and generalization from objects to classes.

- Integration of the operator generation process with activity recognition from human demonstrations, plan generation, and execution procedure.

- Operator collection, which is automatically extended with each new demonstration and prioritizes more often observed operators during the plan generation.

CHAPTER 3

Causal approaches to predict, explain and prevent failures in robotic tasks

A crucial component in human interactions is the ability to explain one's actions, especially when failures occur [13], [14]. It has been argued that this ability is also vital for future robots that act in environments such as homes or hospitals [23], as it can increase trust and transparency of robots [13], [14]. Moreover, explainability facilitates the diagnosis capabilities of a robot, which is crucial for correcting its behavior [21]. In this chapter, we will first provide an overview of the current state of the art in failure explanations. Then we will provide a short introduction to the mathematical framework of causality, which has been identified to be an essential part of human failure explanations [13], and how it has been primarily used in the field of robotics. Finally, we will present an overview of how we use causality to enable robots to find contrastive explanations for robot failures and predict and prevent future failures in robotic tasks.

3.1 Background on failure explanation and causality

Failure explanation

The field of explainable AI (XAI) has recently been experiencing a significant increase in engagement because people started to worry about the transparency and comprehensibility of black box machine learning algorithms that decide on important matters concerning people's lives, such as loan applications or university admission [48]. Otherwise, how should we trust algorithms with provable biases, which occasionally make fundamental mistakes [49]? For many years, XAI has been mainly addressing issues of interpretability and explainability of black-box deep-learning systems [50], [51]. However, with the increasing inclusion of robots in human-centered environments and human-robot collaborative applications, research in explainable agency is gaining importance.

The literature on XAI has established many different reasons and advantages for explainable robot agents, such as an increase in trust [52], transparency [53], comprehensibility or predictability [54]. This, in turn, results in less discomfort and less confusion towards the robot [51] and allows the human to have safer interactions with the robot [55]. Explainable robots furthermore lead to increased efficiency of human-robot team performances [56]. Typical applications for explainable agents are collaborative robot tasks such as working in a factory environment [53] or teaming for military missions [57]. Other applications contain gaming, education, e-health, search and rescue scenarios, or debugging [2]. In these applications, robots are typically equipped with the ability to explain their intentions or can explain their functional capabilities and limitations.

Besides XAI, which mainly addresses the explainability of different Machine Learning algorithms, the trend to develop explainable and interpretable methods has also emerged in the planning community. This research direction is called XAIP (explainable AI planning) [12] and aims to provide more detailed explanations of task plans. These explanations might be required for inference reconciliation (the inference capabilities of humans do not suffice to understand the plan) or model reconciliation (in case the human mental model does not coincide with the robot model/domain) [12]. Typical questions that require explanations are *Why is a particular action in this plan?*, *Why did* *the planning algorithm come up with plan A and not plan B* or *Why is this planning problem not solvable?* [58], [59].

In this thesis, we focus on explanations that are generated by a robot (the explainer) and directed toward a human (the explainee). It was shown that humans tend to assign human traits to machines and often expect explanations that are similar to the ones that humans use when talking to each other [13]. Therefore, when agents are creating explanations, they need to adhere to some norm such that humans will be able to understand them. Human explanations have been extensively studied in areas like philosophy, cognitive science, and social psychology, and [13] summarizes some of the most important features that human explanations possess:

- Contrastive: Humans often provide explanations in terms of counterfactuals, e.g., would I have arrived at work on time if I had left the highway to avoid the traffic jam and taken the small side roads, or more abstractly, what would have happened if we had chosen A instead of event B?

- Selected: humans focus on a subset of relevant causes instead of the entire event chain. For example, explanations for being late to work involve the traffic jam but not the color of the car or things that have happened a week prior.

- Probabilities are typically not included in humans' explanations: people are typically not aware of the exact action outcome probabilities but rather of an approximation. For example, humans failing to build a tower of cubes don't explain this failure by stating that the success probability was 12%, but in terms of more abstract features such as the cube being stacked too far to the right.

- Explanations are social: Explanations take into account the explainer's beliefs about the explainee's mental model of the environment.

Furthermore, in [13], an explanatory agent is able to point out the underlying causes of its decision-making. Therefore the concept of *Causality* plays a fundamental role in the generation of explanations and is thus reviewed more deeply in the next subsection.

Statistical and causal models

Humans are exceptionally good at detecting causal patterns even from limited observations, so it might seem surprising that most statisticians have, for more than the first half of the 20th century, strictly opposed the idea of deriving causal relations from data or observations [23]. The predominant mantra was that correlations do not imply causation. Correlation is a satistical measure that indicates the size and direction of a relationship between several random variables [60]. A correlation between two random variables does not automatically imply that changing one of the variables will also cause a change in the other. This fact can be illustrated in various comical examples where completely unrelated variables correlate without an actual causal relation (see Fig. 3.1). For example, the number of people who drowned by falling into a pool correlates with the number of movies that Nicolas Cage appeared in (see Fig. 3.1-a), or the divorce rate in Maine correlates with the per capita consumption of margarine (see Fig. 3.1-b). Such correlations are also denoted as spurious. Because of this strong opposition towards the derivation of causal relations from data, the mathematical language to express causal and counterfactual relationships between variables has only been developed in the later half of the 20th century [23], [60].

Traditional statistical methods, such as regression or classification, operate on the level of associations/correlations, meaning they fit a model around a set of passive observations and use this model to answer questions about this data. This is how simpler methods such as linear regression but also more advanced methods such as deep neural networks operate: the goal is to learn a set of weights or parameters that fits training data in a way that allows you to predict the output given a set of inputs. For example, as a cafe owner, you could train a deep neural network to predict the probability that a customer that orders coffee also adds milk on top (P(Milk=1|Coffee=1)) or how many customers typically drop by in the morning. One of the prerequisites of reliably applying deep neural networks is that the training and test data are independent and identically distributed (i.i.d). As a result, the data generation process, or in other words, the situations in which the data has been collected, is an important factor that determines the prediction reliability in test data, which is illustrated in the following example. Assume that we, as cafe shop owners, want to know how much revenue we would generate if we doubled the prices of chocolate brownies. Note that this is not a simple observational

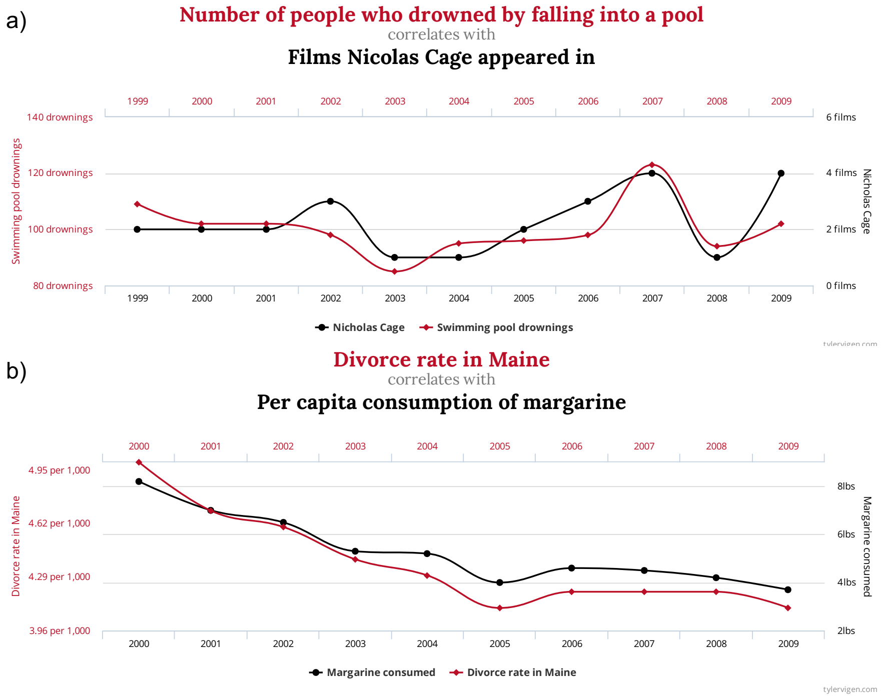


Figure 3.1: Two different examples when *correlation does not imply causation.* Source: `https://matt-rickard.com/correlation-vs-causation`

query anymore since doubling the prices requires an intervention. We actively modify the price of our brownies and want to predict the effect on our revenue while keeping all the other factors as they are right now. Trying to answer this question based on previous observations yields correct predictions only if the data generation process, which means the circumstances and reasons for the higher brownie prices now are the same as in the previous observations. For example, imagine that we had doubled the brownie prices two years ago. However, during that time, there was a chocolate shortage, and other cafe shops ran out of chocolate brownies. As a result, we were able to raise the prices and still attract a lot of customers. Surely, customers today would not be willing to pay double the price on brownies, given there are now cheaper options in other cafe shops available. So instead of asking a conditional probability P(Revenue|DoubledBrowniePrices), which queries data with doubled brownie prices but for different reasons than today, we need to decouple the price DoubledBrowniePrices from all other potentially relevant variables like product availability or production cost, while keeping the rest of the variables as they are. This is what is referred to as intervention. By decoupling the price from some of the relevant factors, we change the distribution of the test data, which in turn violates the i.i.d. assumptions. As a result, the deep neural network won't provide accurate revenue predictions in the discussed example. This example should illustrate the need to formalize causal concepts, such as interventions that go beyond traditional statistics that mainly deals with associations/correlations.

One prominent formalization of the concept of interventions is the `do` operator P(Revenue|`do`(DoubledBrowniePrices)) [23]. In itself, the introduction of the do operator does not solve the problem at hand; namely, how can we predict the revenue if we would only change the price but keep all other factors as they are? The simplest solution would be to start an experiment and offer brownies at different prices for a limited amount of time and then go with the price that optimizes our revenue. Unfortunately, not all problems allow such experiments, e.g., because they are unethical (e.g., we should not force people to smoke to prove that smoking causes lung cancer or force people to buy expensive products), or we might not have the resources to do it. In such cases, however, we can prove in which cases we can answer interventional queries just based on observational data or show that interventional queries cannot be answered no matter how many additional data samples are collected. We are going to consider examples for both of these cases in the following paragraphs.

A large contribution of causality is the establishment of graphical causal models. Such graphical models are very helpful in visualizing the prediction problem (e.g., the effect of doubling brownie prices on revenue). Furthermore, graphical models are crucial for the correct analysis of the data. For example, one big issue for clinical trials aiming to measure a new treatment's effect on a sick patient is confounding. Confounders are variables that have an impact on both the treatment and the effect variable. As an example, let's look at the famous kidney stone recovery data set [60], where the size of kidney stones Z causally impacted both the choice of treatment T as well as the recovery R (Fig. 3.2). In this dataset, there are two types of possible treatments T

for kidney stones, a) open surgery (denoted as $T = a$) and b) percutaneous nephrolithotomy (denoted as $T = b$). Treatment b removes kidney stones by a small puncture wound and would therefore be preferable to open surgery in terms of invasiveness and, judging from the data in Fig. 3.3 provides a seemingly larger recovery rate (78% for $T = a$ vs. 83% for $T = b$). In the available dataset containing 700 patients, half of the patients have been treated with each method. Additionally, we can see that both treatments achieve a larger recovery rate for small kidney stones, which opens up the conclusion that smaller kidney stones are less severe. However, surgery has been applied much more often to severe cases of kidney stones (192/263), whereas treatment b has been more often applied to small, less severe kidney stones (234/270). As a result, the recovery rate (conditional probability $P(R = 1|T = a)$ and $P(R = 1|T = b)$) is not an accurate measure to judge which treatment method is better. Instead, we would like to know the treatment success if both methods had been applied to patients with small and large stones at similar rates. This, again, would result in a different distribution than the data we are presented with. In terms of the `do` operator we would like to know $P(R = 1|\texttt{do}(T = a))$ and $P(R = 1|\texttt{do}(T = b))$. We can prove that this probability is different from the conditional probabilities but nevertheless solve this problem despite only having observational data at hand [60]:

$$\begin{aligned}
P(R = 1|\texttt{do}(T = a) &= \sum_{z=0}^{1} P(R = 1, \texttt{do}(T = a), Z = z) && (3.1)\\
&= \sum_{z=0}^{1} P(R = 1|\texttt{do}(T = a), Z = z)P(T = a, Z = z) && (3.2)\\
&= \sum_{z=0}^{1} P(R = 1|\texttt{do}(T = a), Z = z)P(Z = z) && (3.3)\\
&= \sum_{z=0}^{1} P(R = 1|T = a, Z = z)P(Z = z) && (3.4)\\
&= 0.93\frac{357}{700} + 0.73\frac{343}{700} = 0.832. && (3.5)
\end{aligned}$$

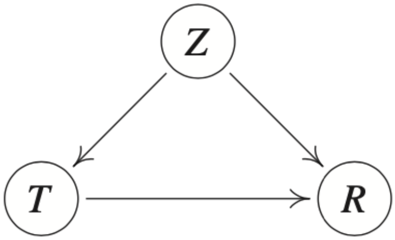
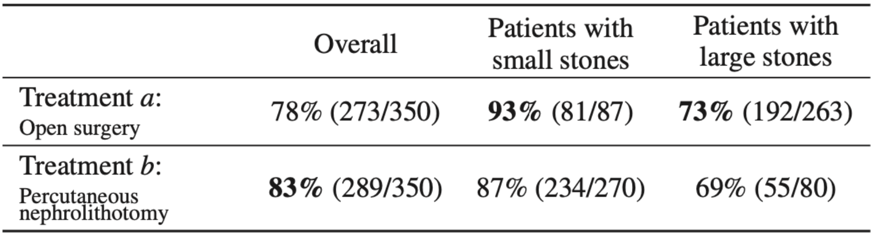


Figure 3.2: Kidney Stone dataset: Graphical Causal Model [60]. Z represents the size of kidney stones, T the treatment (either open surgery or nephrolithotomy and R indicates if the kidney stones were cured or not.



	Overall	Patients with small stones	Patients with large stones
Treatment *a*: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment *b*: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

Figure 3.3: Kidney Stone dataset [60].

Analogously,

$$P(R = 1|\mathtt{do}(T = b) = 0.87\frac{357}{700} + 0.69\frac{343}{700} = 0.782. \tag{3.6}$$

Comparing these two numbers, we can see that if the two treatments were applied similarly to severe and non-severe cases, treatment a would be more effective in treating patients than b. The calculations performed in 3.1 are called adjusting. In the kidney stones example, we adjusted for the size of the kidney stones. However, when we have to choose a proper adjustment set (which is the choice of variables that need to be adjusted for) from a large list of variables, the correct choice is not always as straightforward as in the kidney stone example. Then, the graphical representation provides simple visual features (confounding variabels have an outgoing edge towards the treatment variable (cause) and effect variable) which facilitates the choice of the adjustment set.

The graphical structure can also help decide when an interventional query is not answerable. Without the knowledge of the kidney stone size, we cannot adjust for it and thus would not be able to obtain the interventional probabilities P(R|do(T)). Importantly, this query cannot be answered, even if we collected an infinite amount of data (as long as this data does not contain a measure of the kidney stone size Z). In such cases, we either would need to refine the model or simplify assumptions, e.g., assuming that the effect of the kidney stone size Z on treatment outcome R is negligible.

A third use case of graphical models concerns the adaptability of models. For example, consider an agent or human learning a deep neural network of the effect L (number of years that the patient will survive) of treating patients with a drug D, solely from data. Now, suppose we want to transfer the model to a different part of the world, where diet, hygiene, and other variables like work habits are different. In that case, the model must be retrained, despite these new characteristics merely modifying the numerical relationships among the variables recorded. A structural causal model, however, could be transferred, and only the population-specific prediction functions would need to be relearned.

One way to avoid the danger of unmeasured confounders is to collect data based on randomized controlled trials, which are also referred to as the gold standard for causal inference [23]. By randomizing the variable inputs, the incoming variable edges are deleted since the variable value does not depend on any other input. If, for example, the treatment had been randomized, the dependence from kidney stone size Z to treatment T would be non-existent, and thus, it would not be required to adjust for Z.

To summarize the previous section, we motivated the need for causal concepts such as interventions, e.g., in the form of the `do` operator. We also saw that a graphical representation of causal relations played a major role in analyzing causal problems. In the next section, we are going to formalize causal models and discuss under which assumptions and how causal models can be learned from data. In the upcoming sections, we will discuss how causal models can be used in the robotics domain and, in particular, how we use causal models to enable robots to find contrastive explanations of their action execution failures.

Learning cause-effect models: causal discovery

Formally, Structural Causal Models are defined via a graphical structure $\mathcal{G} = (\mathbf{V}, A)$, which is a *directed acyclic graph* (DAG), where $\mathbf{V} = \{X_1, X_2, ..., X_n\}$ represents the set of nodes, and A is the set of arcs [61]. Each node $X_i \subseteq \mathbf{X}$ represents a random variable. Based on the dependency structure of the DAG and the *Markov property*, the *joint probability distribution* of a Bayesian network can be factorized into a set of *local probability distributions*, where each random variable X_i only depends on its direct parents Π_{X_i}:

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i | \Pi_{X_i}) \tag{3.7}$$

Learning a structural causal model is typically split into the two steps of structure learning (retrieving the graphical representation) and parameter learning (estimating the local probability distributions).

Structure Learning

The purpose of this step is to learn the graphical representation of the network $\mathcal{G} = (\mathbf{V}, A)$. There are several different categories of algorithms that can be used to retrieve the causal structure from data [61], [62]. *Constraint-based* methods like Grow-and-Shrink [63], test for conditional independencies in order to construct a graph that reflects these conditional independencies. *Score-based* algorithms, like Hill-Climbing, validate how well different candidate graphs fit the data based on some scoring function such as Minimum Description Length or Bayesian Dirichlet equivalent score [61]. Continuous optimization-based approaches, like NOTEARS [64], regard structure learning as a purely continuous optimization problem. An extensive survey on this class of algorithms is found in [62].

One of the biggest obstacles to learning the causal structure is that the mapping from joint distribution to a graph is not unique. Without further assumptions, it is, in general, only possible to retrieve a group of possible structures which are in the same Markov equivalence classes. It was shown that two DAGs are Markov equivalent iff they have the same skeleton and v-structure [65], where skeleton means that two graphs have the same edges (without taking into account the direction of the graphs). For example, the three graphs in Fig. 3.4. The exact graph can be established either by exploit-

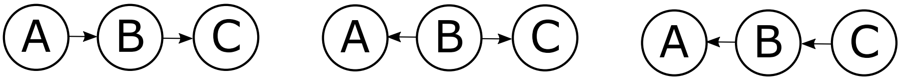


Figure 3.4: One example of a Markov Equivalence Class. All these three graphs have the same joint probability distribution.

ing various forms of interventions [62] or additional assumptions regarding the model (e.g., linear) or the noise variables (e.g., additive noise) [60].

Typically, most structure learning algorithms require the sufficiency assumption. Causal Sufficiency means that all common causes of all pairs of measurable variables in a graph are also measured. In other words, this assumption makes sure that all possible confounders are included in the dataset that we use to learn the causal network [62].

Parameter Learning

The goal of this step is the estimation of the conditional probability distributions of all analyzed variables. In the remainder of the thesis, we typically discretize all random variables. Therefore the conditional probability distribution can be expressed in the form of a probability table. Each parameter θ of this table represents one of the probabilities of all possible outcomes of a variable given all possible parametrizations of its parents. For example, if we had a binary variable with two parent variables that have 5 possible categories each, we would need to estimate 25 parameters $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_{25}]$. There are different ways to estimate these variables, like Maximum Likelihood Estimation (MLE) or Bayesian Estimators. In the case of MLE, it can be shown that the optimal value for a single $\hat{\theta}$ evaluates to

$$\hat{\theta}_{ML} = \frac{1}{n}\Sigma_{i=1}^{n} x_i = \frac{N_1}{N_1 + N_0} = \frac{N_1}{N}, \tag{3.8}$$

where N_1 represents the number of positive samples, N_0 the number of negative samples and N the total number of observations.

Unlike the ML estimator, MAP belongs to the group of bayesian approaches which incorporate our belief about θ in the form of a prior:

$$p(\theta|X) = \frac{p(X|\theta)p(\theta)}{p(X)} \tag{3.9}$$

For a likelihood that is Bernoulli distributed and a Beta prior, the MAP estimation of θ is

$$\hat{\theta}_{MAP} = \frac{N_1 + \alpha - 1}{N_1 + N_0 + \alpha + \beta - 2}. \tag{3.10}$$

where N_1 represents the number of positive samples, N_0 the number of negative samples and α, β represent the parameters of the Beta prior.

Learning Bayesian Networks through the two steps of structure learning and parameter learning is the basis for our causal-based contrastive failure explanation method that we are going to present in the upcoming sections.

Causality in robotics

Despite being acknowledged as an important concept, causality is relatively underexplored in the robotics domain [14], [66]. Some works explore causality to distinguish between task-relevant and -irrelevant variables [67]. For example, CREST [68] uses causal interventions on environment variables to discover which of the variables affect an RL policy. They find that excluding irrelevant variables positively impacts generalizability and sim-to-real transfer. In [69] a set of causal rules is defined to learn to distinguish between unimportant features in physical relations and object affordances. A humanoid iCub robot learns through cumulative experiences that dropping heavy objects into a jar of water will increase the water level, and other variables like color are irrelevant. Brawer et al. present a causal approach to tool affordance learning [66]. Some works explore Bayesian networks to learn statistical dependencies between object attributes, grasp actions, and a set of task constraints from simulated data [70]. While the main objective is to use graphical models to generalize task executions, these works don't look into the question of how these models can be utilized for failure explanations. A different paper [71] investigates the problem of learning causal relations between actions in household-related tasks. They discover, for example, that there is a causal connection between opening a drawer and retrieving plates from human demonstrations. The learning is based on data that was obtained from human expert demonstrations, which were instructed, for example, to clean the table or wash the dishes in a virtual reality environment, but only causal links between actions are retrieved. We, on the other hand, focus on causal relations between different environment variables, like object features and the action outcome. On top of that, we utilize such causal models to enable robots with the ability to find explanations for task execution failures. In the planning domain, cause-effect relationships are represented through (probabilistic) planning operators [72]. Mitrevksi et al. [21] propose the concept of learning task execution models, which consists of learning symbolic preconditions of a task and a function approximation for the success model based on Gaussian Process models. They noted that a simulated environment could be incorporated for a faster and more extensive experience acquisition, as proposed in [70]. Human virtual demonstrations have been used to construct planning operators to learn cause-effect relationships between actions and observed state-variable changes [72]. However, even though symbolic planning operators are considered human-understandable, they are not explanations in themselves, thus requiring an additional layer to interpret the models and generate failure explanations.

Some other works also aim to learn probabilistic action representations experience to generalize the acquired knowledge. For example, learning probabilistic action effects of dropping objects into different containers [73]. Again, the main objective is to find an intelligent way of generalizing the probability predictions for a variety of objects, e.g., bowl vs. bread box, but their method does not include any understanding of why there is a difference in the dropping success probabilities between these different objects. In our work, we not only discuss how to learn cause-effect models but utilize them to explain robot execution failures in a contrastive manner.

Contrastive explanations are deeply rooted in the human way of generating explanations [13]. This also has a significant impact on explanation generation in other fields like Explainable AI Planning (XAIP) [12]. In XAIP, typical questions that a machine should answer are *why a certain plan was generated vs. another one?* or *why the plan contains a particular action* a_1 *and not action* a_2*?* [12], [74]. We, however, are mostly interested in explaining why specific actions failed based on environment variables like object features. A method for explaining synthesis failures of high-level robot task specifications (encoded through Linear temporal logic formulae) is presented in [75]. However, the causes need to be explicitly modeled (violations of specification constraints), while, in our approach, the causes are automatically detected during the BN learning process. Das et al. generate verbal failure explanations [19], by learning an encoder-decoder network that maps current information about the robot and environment state into a vector of words. However, the method does not scale well since it requires data with annotations about each failure cause. Our approach only requires annotations regarding the action success, which can be binary and are generally easier obtainable. Additionally, we encode the explanations directly in the causal structure of the different state variables instead of learning a black-box model. In a follow-up study [20], the authors use MOTIFNET [76] to autonomously detect spatial relationships and object attributes in a given scene. Then, pairwise ranking is used to filter out the subset of relevant relations for a particular explanation. Annotations for pairwise preferences of one relation over another need to be provided for training an SVM, which cannot be easily automated since they require human input.

3.2 Paper B: Our causal approach to explain robot action failures

In Paper B, we present our novel method for generating causal explanations of failures based on a causal model that provides robots with a partial understanding of their environment. First, we learn a causal Bayesian network from simulated task executions, tackling the problem of knowledge acquisition. We also show that the obtained model can transfer the acquired knowledge (experience) from simulation to reality and is agnostic to several real robots with different embodiments. Second, we propose a new method to generate explanations of execution failures based on the learned causal knowledge. Our method is based on a contrastive explanation comparing the variable parametrization associated with the failed action with its closest parametrization that would have led to a successful execution, which is found through breadth-first search (BFS). Finally, we analyze the benefits of this method in two different scenarios: I) stacking cubes and II) dropping spheres into a container.

To summarize, our contributions to Paper B are as follows:

- We present a novel method to generate contrastive causal explanations of action failures based on causal Bayesian networks.

- We demonstrate how causal Bayesian networks can be learned from simulations, exemplified in a cube stacking and sphere dropping scenario, and provide extensive real-world experiments that show that the obtained causal models are transferable from simulation to reality without any retraining. Our method is agnostic to various robot platforms with different embodiments and scales over multiple tasks and scenarios. We, thus, show that the simulation-based model serves as an excellent prior experience for the explanations, making them more generally applicable.

3.3 Paper C: Our causal approach to predict and prevent failures in robotic tasks

The causal relations obtained by the Bayesian networks can be used to predict how likely a particular parametrization of causes will produce failures. In Paper C, we propose an extension of Paper B, which makes use of the prediction capabilities of the learned BNs to prevent failures from happening. When the prediction of a failure has a high probability given the current state, our method finds an alternative execution state, which is expected to result in a successful action execution. This alternative state is found through BFS in a similar fashion as in Paper B, which allows the agent not only to prevent failures but, at the same time, to provide explanations for its corrective actions.

Predicting and preventing errors is particularly difficult if the effects of an action are not immediately flawed but become problematic in future actions [77]. For example, the error was produced on the first action, but the consequence is only observed after the third action (in the future). We call these cases timely shifted action errors. In such cases, the models need to consider the history of the previous actions. For example, imagine the task of building a tower of 4 cubes, and the second cube is not stacked entirely centered with respect to the bottom cube. Even if this particular stack can be considered successful on its own, it negatively impacts the overall stability of the tower, which might become a problem later after the second or third stack. This challenging problem is also addressed in this paper, and we show that our causal-based method scales to these complex cases by detecting causal links over the history of several actions, effectively predicting and preventing action failures.

To summarize, our contributions of Paper C are as follows:

- We propose a causal-based method that allows robots to understand possible causes for errors and predict how likely an action will succeed.
- We then introduce a novel method that utilizes these prediction capabilities to find corrective actions which will allow the robot to prevent failures from happening.
- Our algorithm proposes a solution to the complex challenge of timely shifted action effects. By detecting causal links over the history of several actions, the robot can effectively predict and prevent failures even if the root of a failure lies in a previous action.

3.4 Paper D: Transferable priors for Bayesian Network parameter estimation in robotic tasks

One of the major challenges in robotics is the ability to transfer prior experience to new domains and tasks. In that regard, causal Bayesian Networks (BNs) are a great tool to model and estimate the outcome of robotics tasks which has lead to an increased interest in the robotics community [14], [66], [78]. However, due to the statistical nature of BN learning methods, constructing a BN can be very data-consuming [79]. This is particularly problematic when we attempt to learn BNs from real robot experiments, where collecting ten- or even a hundred of thousands of data samples is expensive and time-consuming. One solution could be using prior knowledge and transferring it to related tasks. For example, we would like to use the knowledge of stacking one cube for the task of stacking two cubes. Stacking two cubes is more complex since the successful stacking outcome of the second cube highly depends on the first stack. Nevertheless, both tasks are closely related and share similar variables.

Learning a Bayesian network consists of two steps: A) learning a graphical representation of the variable relations (structure learning) and B) learning the conditional probability distributions for each variable (parameter learning). Often, the variable relations are intuitively transferable to related problems. Let us consider, for example, the task of dropping a sphere onto plates. The success of such a task might depend on parameters like x- (right/left) and y-Offset (up/down) between the sphere and the center of the plate or diameter and the height of the plate. This set of relations is also true when we try to drop spheres into bowls. However, the precise success probabilities are likely to change. Therefore an open question that we investigate in this paper is under which circumstances such prior knowledge might aid the parameter estimation process, which is step B) of the BN learning pipeline. We specifically focus on the case of binary and categorical (discrete) variables. Thus the outcome of the parameter estimation steps is a conditional probability table.

A common parameter estimation method is Maximum-Likelihood estimation (ML), which does, however, not allow the usage of any prior information. In Paper D, we, therefore, consider the usage of the Maximum-a-Posteriori (MAP) estimator, which can incorporate prior knowledge into the estimation process. The contributions of this paper can be summarized as follows:

- We propose a method that constructs parameter priors from previous experience and transfers it to related but different tasks.
- We conduct a detailed comparison between learning a parameter model from scratch (ML) and learning from a prior (MAP), mainly regarding data efficiency.
- We test and compare the outcome of the two estimation methods for the use-case of failure prevention as proposed in [80]. A special focus is thereby set on cases where no or only a few data about the new task is available.

CHAPTER 4

Summary of included papers

This chapter provides a summary of the included papers.

4.1 Paper A

Diehl Maximilian, Paxton Chris, Ramirez-Amaro Karinne
Automated Generation of Robotic Planning Domains from Observations
IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, Online, Oct. 2021 .

This paper presents an interpretable approach to learning from human demonstrations, which addresses our proposed solution to RQ1 (How can we extract and learn a task in a flexible, robot-agnostic, and interpretable manner by observing humans demonstrating the task?). This paper investigates the automatic generation of the planning domain, which enables robots to find plans for achieving complex, long-horizon tasks given a planning domain. This planning domain consists of a list of actions, with their associated preconditions and effects, and is usually manually defined by a human expert, which is very time-consuming or even infeasible. In this paper, we introduce a novel method for generating this domain automatically from human demonstrations. First, we automatically segment and recognize the different observed actions from human demonstrations. From these demonstrations, the relevant preconditions and effects are obtained, and the associated planning operators are generated. Finally, a sequence of actions that satisfies a user-defined goal can be planned using a symbolic planner. The generated plan is executed in a simulated environment by the TIAGo robot. We tested our method on a dataset of 12 demonstrations collected from three different participants. The results show that our method is able to generate executable plans from using one single demonstration with a 92% success rate, and 100% when the information from all demonstrations are included, even for previously unseen stacking goals.

4.2 Paper B

Diehl Maximilian, Ramirez-Amaro Karinne
Why did I fail? A Causal-based Method to Find Explanations for Robot Failures
IEEE Robotics and Automation Letters, vol. 7, no. 4, Oct. 2022 .

This paper presents contributions towards providing robots with a causal understanding of an action (RQ2: reliable detection of cause-effect relationships) and our method that allows robots to find explanations of action failures (RQ3: prediction and explanation of failures): robot failures in human-centered environments are inevitable. Therefore, the ability of robots to explain such failures is paramount for interacting with humans to increase trust and transparency. To achieve this skill, the main challenges addressed in this paper are I) acquiring enough data to learn a cause-effect model of the environment and II) generating causal explanations based on the obtained model. We address I) by learning a causal Bayesian network from simulation data. Concerning II), we propose a novel method that enables robots to generate contrastive explanations upon task failures. The explanation is based on setting the failure state in contrast with the closest state that would have allowed for successful execution. This state is found through breadth-first search and is based on success predictions from the learned causal model. We assessed our method in two different scenarios I) stacking cubes and II) dropping spheres into a container. The obtained causal models reach a sim2real accuracy of

70% and 72%, respectively. We finally show that our novel method scales over multiple tasks and allows real robots to give failure explanations like “the upper cube was stacked too high and too far to the right of the lower cube.”

4.3 Paper C

Diehl Maximilian, Ramirez-Amaro Karinne
A Causal-based Approach to Explain, Predict and Prevent Failures in Robotic Tasks
Conditionally accepted with minor revisions to *Robotics and Autonomous Systems (RAS)*, Elsevier, 2022 .

This paper is an extension to Paper B that focuses on the explanation and prevention (RQ3: How can a robot predict, explain and prevent failures?) of timely shifted action effects (RQ2: detection of cause-effect relationships involving timely shifted, erroneous action effects): robots working in real environments need to adapt to unexpected changes to avoid failures. This is an open and complex challenge that requires robots to timely predict and identify the causes of failures to prevent them. In this paper, we present a causal method that will enable robots to predict when errors are likely to occur and prevent them from happening by executing a corrective action. First, we propose a causal-based method to detect the cause-effect relationships between task executions and their consequences by learning a causal Bayesian network (BN). The obtained model is transferred from simulated data to real scenarios to demonstrate the robustness and generalization of the obtained models. Based on the causal BN, the robot can predict if and why the executed action will succeed or not in its current state. Then, we introduce a novel method that finds the closest state alternatives through a contrastive Breadth-First-Search if the current action was predicted to fail. We evaluate our approach for the problem of stacking cubes in two cases; a) single stacks (stacking one cube) and; b) multiple stacks (stacking three cubes). In the single-stack case, our method was able to reduce the error rate by 97%. We also show that our approach can scale to capture multiple actions in one model, allowing to measure timely shifted action effects, such as the impact of an imprecise stack of the first cube on the stacking success of the third cube. For these complex situations, our model was able to prevent around 75% of the stacking errors, even for the challenging multiple-stack scenario. Thus, demonstrating that our method is able to explain, predict, and prevent execution failures, which even scales to complex scenarios that require an understanding of how the action history impacts future actions.

4.4 Paper D

Diehl Maximilian, Ramirez-Amaro Karinne
Transferable Priors for Bayesian Network Parameter Estimation in Robotic Tasks
Submitted to *IEEE International Conference on Robotics and Automation (ICRA)*, 2023 .

This paper analyses the data requirements for learning causal BNs with a particular focus on the estimation of the conditional probability distributions and investigates the problem of transferring prior experience with the goal of improving data efficiency (RQ4: knowledge transfer in the form of causal models from one task to another): one of the major challenges in robotics is the ability to transfer prior experience to new domains. In that regard, causal Bayesian Networks are an effective tool for modeling the outcome of various robotic tasks. Learning a Bayesian network from data consists of learning a graphical representation that captures the relations of analyzed variables and learning the conditional probability distributions for each variable (parameter learning). While the obtained graphical representation can often be transferred between tasks, the learned parameters (distributions) must be re-learned. This represents a problem in robotics where real robot data is not as easily available as simulation data. Therefore, the transferability of the learned models is a challenging problem and is the focus of this paper. We analyze different possibilities to abstract priors from simulations through bayesian estimation methods like Maximum a posteriori (MAP). We investigate the transferability capability of the learned prior in two cases 1) learn a model from a single stack and transfer it to execute a second stack; 2) learn and transfer priors for the task of dropping a sphere into containers (plates to bowls). Finally, we compare the data efficiency between MAP and Maximum-Likelihood (ML) estimation, which implies learning the task parameters from scratch. We show that the ML estimation converges multiple times faster towards the true parameters compared to MAP, which makes ML generally preferable; however, in very sparse data cases, the MAP achieves a substantially smaller variance than ML.

CHAPTER 5

Concluding Remarks and Future Work

In this thesis, we presented interpretable and explainable learning and decision-making methods that should accommodate different types of robot failures.

- First, we presented an interpretable approach to learning tasks from human demonstration (LfD) based on a decision tree enhanced with Knowledge representation. LfD is considered to be an essential tool for autonomous robots to continuously extend their capabilities by learning new tasks [22]. Moreover, interpretability is crucial since we have a human in the loop (as a teacher and demonstrator of the task). Our approach automatically generates symbolic, human-readable planning operators from the demonstration, which can then be used to flexibly generate plans for robot tasks.

- In the second part of this thesis, we focused on the analysis of the challenging problem of robot execution failures. We presented methods that allow robots to provide contrastive explanations of execution failures based on causal models of the action environment. The ability to explain why failures have occurred is important to foster trust and also allows robots to prevent future failures from happening. With an extension of this causal-based method, we were even able to explain and prevent timely shifted action failures, which cover cases where current actions are not considered failures but will negatively affect the success of future actions.

- Finally, we analyzed the problem of transferring prior experience to improve the data efficiency of Bayesian Network learning, with a particular focus on estimating the conditional probability distributions. This investigation aims to help robots learn causal models faster, enabling them to provide failure explanations at the cost of fewer action execution experiments.

In this final chapter, I will shortly recapitulate the research questions and how they were addressed in this thesis. Furthermore, I will expand on future work and open questions for each of the research questions respectively.

Goal 1: Life-long learning (or learning from demonstration)

RQ1: How can we extract and learn a task in a flexible, robot-agnostic, and interpretable manner by observing humans demonstrating the task?

As opposed to end-to-end learning of the demonstration [24], our approach from Paper A segments the demonstrated task into a series of actions. This is done by segmenting and classifying the continuous hand motions into a minimal set of activities, like `IdleMotion`, `Reach`, `Put`, `Take`, and `Stack`. One advantage of the semantic-based recognition method is that it can be easily enhanced with a First-order-Logic reasoning method and an ontology system [47] to improve the generalization of the obtained models. Another advantage is its ability to segment and recognize continuous data without a new training step. Furthermore, as hand motion recognition is based on a decision tree, the operator generation process is interpretable by humans. The preconditions of a planning operator are based on the effects of the last state before the activity transition, and effects are based on the last frame of the current activity. Both preconditions and effects are expressed in terms of symbolic states and can be therefore used with any robotic system or robot task. Since each planning operator is based on one of the previously detected and classified activities, operators can be named automatically with human-understandable labels, thus providing a semantic description of the under-laying functionality.

Moreover, operators are defined in terms of human-understandable symbols, which allows humans to understand the obtained task plan and its execution more easily.

Currently, we have only focused on high-level task abstraction. Nevertheless, if we want to be able to execute the resulting high-level action plans, we need to bridge the gap to the low-level execution functions. One possibility could be to learn robot-specific execution policies for each planning operator. These policies could be learned through methods like Reinforcement learning (RL) and should be rearrangeable and re-usable as instructed by the high-level action plan. Similarly, feedback from the RL training could be used to update the high-level planning operators, e.g., by removing/adding preconditions/effects or adding information about action execution costs and success probabilities. By enriching RL policies with symbolic descriptions of what each policy aims to achieve (through the precondition and effect description of the operator), action executions and, specifically, failures could be more easily understandable by humans. This is considered future work.

Goal 2: Explanation of a robot's actions

RQ2: How can we reliably detect cause-effect relationships in robot tasks involving timely shifted and erroneous action effects?

In Paper B and C, we have presented a pipeline to learn causal models of robot actions. A causal understanding of how actions impact and are impacted by the environment is crucial knowledge to be able to explain actions and, in particular, why failures have occurred. Initially, we define a set of random variables that describe the task of interest, including potential cause and outcome variables. We then collect data through randomized controlled simulations of the action. Based on the obtained data, we first learn the causal structure of the random variables through methods like Grow-Shrink [63] or the PC algorithm [81]. Necessary assumptions are discussed in more detail in Sec.3.1. Then given the causal structure, we can apply parameter learning methods like MLE or Bayesian Estimators to estimate the conditional probability distributions of the variables. In Paper C, we demonstrated the robustness of our proposed pipeline toward more complex tasks with more random variables. By representing several actions together in one model, we can handle cases where the observed effect was not immediate.

Future research directions include BN learning under incomplete data samples and hybrid BN learning algorithms (mixing discrete and continuous random variables), which could potentially open up the application of our failure explanation method to different industrial robotic systems.

RQ3: How can a robot predict, explain and prevent failures?

In Paper B, we presented a novel method that generates contrastive failure explanations based on the obtained causal Bayesian Networks. These BNs can be used to generate task success predictions; however, simply stating that the task had to fail because the success chance was low is not an explanation of why the failure has occurred. Humans typically use contrastive explanations, and we generate this contrast by conducting BFS, starting from the current failure parametrization. By changing one variable value at a time, our proposed algorithm searches until it finds the closest state, which has a significant success change (e.g., over 90%). Then, a contrastive failure explanation can be provided by comparing the changing variables between these two parametrizations. In Paper C, we showed that, by learning one causal BN over several actions, we could address the challenging issue of explaining timely-shifted action failures and prevent future action failures through adjustments of the current action.

In the future, we want to use the proposed methods as a bridge between high-level task planning (e.g., in the form of PDDL action plans [27]) and the low-level execution functions since they would allow a planner to flexibly insert high-level corrective actions, given low-level action imperfections or failures, while at the same time providing explanations for the required corrections. Furthermore, we want to explore different search heuristics to find contrastive failure explanations. Our current approach (BFS) provides the closest success parametrization, following Occam's Razor principle. However, while providing a simple explanation, it might not necessarily be the most human-like explanation, which will require a user study for evaluation.

RQ4: How can a robot transfer knowledge in the form of causal models from one task to another?

Learning causal BNs is based on statistical methods, which are data-intensive. In Paper D, we, therefore, analyzed the data requirements of the parameter estimation step (estimation of conditional probability distributions given the knowledge of the causal graph) more closely. We proposed a method to transfer parameter priors obtained from related but different tasks, e.g., from stacking one cube to two cubes or from dropping a sphere from a plate to a new container (bowl). We could show that, in terms of data efficiency, the ML estimation (learning the task from scratch) quickly outperforms the MAP method (using prior information). However, in particular, for the sparse data case, the priors can represent valuable experience and have the advantage of a smaller variance. In the future, we would like to investigate if and how effective priors could also be used for learning the causal network structure.

References

[1] S. Schaal, "The New Robotics—towards human-centered machines," *HFSP Journal*, vol. 1, no. 2, pp. 115–126, Jul. 2007, ISSN: 1955-2068.

[2] S. Anjomshoae, A. Najjar, D. Calvaresi, and K. Främling, "Explainable agents and robots: Results from a systematic literature review," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '19, Montreal QC, Canada: International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1078–1088, ISBN: 9781450363099.

[3] P.-C. Yang, K. Sasaki, K. Suzuki, K. Kase, S. Sugano, and T. Ogata, "Repeatable folding task by humanoid robot worker using deep learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 397–403, 2017.

[4] J. Kinugawa, H. Suzuki, J. Terayama, and K. Kosuge, "Underactuated robotic hand for a fully automatic dishwasher based on grasp stability analysis," *Advanced Robotics*, vol. 36, no. 4, pp. 167–181, 2022.

[5] R. E. Bloss, "Mobile hospital robots cure numerous logistic needs," *Ind. Robot*, vol. 38, pp. 567–571, 2011.

[6] J. Hu, A. Edsinger, Y. Lim, *et al.*, "An advanced medical robotic system augmenting healthcare capabilities - robotic nursing assistant," in *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*, IEEE, 2011, pp. 6264–6269.

[7] E. Broadbent, “Interactions with robots: The truths we reveal about ourselves,” *Annual Review of Psychology*, vol. 68, no. 1, pp. 627–652, 2017, PMID: 27648986.

[8] H. Krebs and B. Volpe, “Chapter 23 - rehabilitation robotics,” in *Neurological Rehabilitation*, ser. Handbook of Clinical Neurology, M. P. Barnes and D. C. Good, Eds., vol. 110, Elsevier, 2013, pp. 283–294.

[9] A. Grau, M. Indri, L. L. Bello, and T. Sauter, “Industrial robotics in factory automation: From the early stage to the internet of things,” in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 6159–6164.

[10] J. Guiochet, M. Machin, and H. Waeselynck, “Safety-critical advanced robots: A survey,” *Robotics and Autonomous Systems*, vol. 94, pp. 43–52, 2017, ISSN: 0921-8890.

[11] Z. Han, E. Phillips, and H. A. Yanco, “The need for verbal robot explanations and how people would like a robot to explain itself,” *J. Hum.-Robot Interact.*, vol. 10, no. 4, Sep. 2021.

[12] T. Chakraborti, S. Sreedharan, and S. Kambhampati, “The emerging landscape of explainable automated planning & decision making,” *International Joint Conference on Artificial Intelligence*, C. Bessiere, Ed., pp. 4803–4811, 2020.

[13] T. Miller, “Explanation in artificial intelligence: Insights from the social sciences,” *Artificial Intelligence*, vol. 267, pp. 1–38, 2019.

[14] T. Hellström, “The relevance of causation in robotics: A review, categorization, and analysis,” *Paladyn, Journal of Behavioral Robotics*, vol. 12, no. 1, pp. 238–255, 2021.

[15] A. Mohseni-Kabir, C. Rich, S. Chernova, C. L. Sidner, and D. Miller, “Interactive hierarchical task learning from a single demonstration,” in *2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2015, pp. 205–212.

[16] K. Kawamura, P. Nilas, K. Muguruma, J. Adams, and C. Zhou, “An agent-based architecture for an adaptive human-robot interface,” in *36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the*, 2003.

[17] K. Drnec, G. Gremillion, D. Donavanik, *et al.*, "The role of psychophysiological measures as implicit communication within mixed-initiative teams," in *Virtual, Augmented and Mixed Reality: Interaction, Navigation, Visualization, Embodiment, and Simulation*, Springer International Publishing, 2018, pp. 299–313.

[18] C. Kardos, Z. Kemény, A. Kovács, B. E. Pataki, and J. Váncza, "Context-dependent multimodal communication in human-robot collaboration," *Procedia CIRP*, vol. 72, pp. 15–20, 2018, 51st CIRP Conference on Manufacturing Systems, ISSN: 2212-8271.

[19] D. Das, S. Banerjee, and S. Chernova, "Explainable ai for robot failures: Generating explanations that improve user assistance in fault recovery," *ACM/IEEE International Conference on Human-Robot Interaction*, pp. 351–360, 2021.

[20] D. Das and S. Chernova, "Semantic-based explainable ai: Leveraging semantic scene graphs and pairwise ranking to explain robot failures," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3034–3041, 2021.

[21] A. Mitrevski, P. G. Plöger, and G. Lakemeyer, "Representation and experience-based learning of explainable models for robot action execution," in *IROS*, 2020, pp. 5641–5647, ISBN: 978-1-7281-6212-6.

[22] H. Bekkering, A. Wohlschläger, and M. Gattis, "Imitation of gestures in children is goal-directed," *The Quarterly Journal of Experimental Psychology Section A*, vol. 53, no. 1, pp. 153–164, 2000.

[23] J. Pearl and D. Mackenzie, *The Book of Why: The New Science of Cause and Effect.* USA: Basic Books, Inc., 2018, ISBN: 046509760X.

[24] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009, ISSN: 0921-8890.

[25] M. Diehl, A. Plopski, H. Kato, and K. Ramirez-Amaro, "Augmented reality interface to verify robot learning," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, 2020, pp. 378–383.

[26] F. Ingrand and M. Ghallab, “Deliberation for autonomous robots: A survey,” *Artificial Intelligence*, vol. 247, pp. 10–44, 2017, Special Issue on AI and Robotics, ISSN: 0004-3702.

[27] M. Ghallab, C. Knoblock, D. Wilkins, *et al.*, “Pddl - the planning domain definition language,” Aug. 1998.

[28] O. Giménez and A. Jonsson, “The complexity of planning problems with simple causal graphs,” *Journal of Artificial Intelligence Research*, vol. 31, pp. 319–351, 2008.

[29] M. Ghallab, D. Nau, and P. Traverso, *Automated planning. Theory & practice.* May 2004, ISBN: 978-1-55860-856-6.

[30] A. Lindsay, J. Read, J. Ferreira, T. Hayton, J. Porteous, and P. Gregory, “Framer: Planning models from natural language action descriptions,” in *ICAPS*, 2017.

[31] R. E. Fikes and N. J. Nilsson, “STRIPS: A new approach to the application of theorem proving to problem solving,” *Artificial Intelligence*, vol. 2, no. 3, pp. 189–208, 1971.

[32] Jonsson, Morris, Muscettola, Rajan, and Smith, “Planning in interplanetary space: Theory and practice,” in *International Conference on Artificial Intelligence Planning Systems (AIPS 2000)*, Breckenridge, CO, Apr. 2000.

[33] F. Ingrand, S. Lacroix, S. Lemai-Chenevier, and F. Py, “Decisional autonomy of planetary rovers,” *Journal of Field Robotics*, vol. 24, 2007.

[34] P. Doherty, J. Kvarnström, and F. Heintz, “A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems,” *Autonomous Agents and Multi-Agent Systems*, vol. 19, pp. 332–377, Dec. 2009.

[35] E. Erős, M. Dahl, A. Hanna, P.-L. Götvall, P. Falkman, and K. Bengtsson, “Development of an industry 4.0 demonstrator using sequence planner and ros2,” in *Robot Operating System (ROS): The Complete Reference (Volume 5)*, A. Koubaa, Ed. Cham: Springer International Publishing, 2021, pp. 3–29, ISBN: 978-3-030-45956-7.

[36] M. Cashmore, M. Fox, D. Long, *et al.*, “Rosplan: Planning in the robot operating system,” in *ICAPS*, Jerusalem, Israel: AAAI Press, 2015, pp. 333–341, ISBN: 9781577357315.

[37] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "From skills to symbols: Learning symbolic representations for abstract high-level planning," *J. Artif. Int. Res.*, vol. 61, no. 1, pp. 215–289, Jan. 2018, ISSN: 1076-9757.

[38] J. Gu, D. S. Chaplot, H. Su, and J. Malik, "Multi-skill mobile manipulation for object rearrangement," *arXiv preprint arXiv:2209.02778*, 2022.

[39] S. Jiménez, T. De La Rosa, S. Fernández, F. Fernández, and D. Borrajo, "A review of machine learning for automated planning," *The Knowledge Engineering Review*, vol. 27, no. 4, pp. 433–467, 2012.

[40] A. Arora, H. Fiorino, D. Pellier, M. Métivier, and S. Pesty, "A review of learning planning action models," *The Knowledge Engineering Review*, vol. 33, Nov. 2018.

[41] S. Cresswell and P. Gregory, "Generalised domain model acquisition from action traces.," in *ICAPS*, Jan. 2011.

[42] H. H. Zhuo and S. Kambhampati, "Action-model acquisition from noisy plan traces," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI '13, Beijing, China: AAAI Press, 2013, pp. 2444–2450, ISBN: 9781577356332.

[43] A. Ahmetoglu, M. Y. Seker, A. Sayin, *et al.*, "Deepsym: Deep symbol generation and rule learning from unsupervised continuous robot interaction for planning," *CoRR*, vol. abs/2012.02532, 2020.

[44] N. Abdo, H. Kretzschmar, L. Spinello, and C. Stachniss, "Learning manipulation actions from a few demonstrations," in *ICRA*, May 2013, pp. 1268–1275, ISBN: 978-1-4673-5641-1.

[45] S. R. Ahmadzadeh, A. Paikan, F. Mastrogiovanni, L. Natale, P. Kormushev, and D. G. Caldwell, "Learning symbolic representations of actions from human demonstrations," in *ICRA*, 2015, pp. 3801–3808.

[46] L. P. Kaelbling and T. Lozano-Pérez, "Learning composable models of parameterized skills," in *ICRA*, 2017, pp. 886–893.

[47] T. Bates, K. Ramirez-Amaro, T. Inamura, and G. Cheng, "On-line simultaneous learning and recognition of everyday activities from virtual reality performances.," in *IROS*, 2017, ISBN: 978-1-5386-2682-5.

[48] C. O'Neil, *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy.* USA: Crown Publishing Group, 2016, ISBN: 0553418815.

[49] D. S. Char, N. H. Shah, and D. Magnus, "Implementing machine learning in health care—addressing ethical challenges," *The New England journal of medicine*, vol. 378, no. 11, p. 981, 2018.

[50] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau, "Visual analytics in deep learning: An interrogative survey for the next frontiers," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 8, pp. 2674–2693, 2019.

[51] O. Biran and C. Cotton, "Explanation and justification in machine learning: A survey," in *IJCAI-17 workshop on explainable AI (XAI)*, vol. 8, 2017, pp. 8–13.

[52] M. W. Boyce, J. Y. Chen, A. R. Selkowitz, and S. G. Lakhmani, "Effects of agent transparency on operator trust," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*, 2015, pp. 179–180.

[53] B. Hayes and J. A. Shah, "Improving robot controller transparency through autonomous policy explanation," in *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, 2017, pp. 303–312.

[54] A. D. Dragan, S. Bauman, J. Forlizzi, and S. S. Srinivasa, "Effects of robot motion on human-robot collaboration," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '15, Portland, Oregon, USA: Association for Computing Machinery, 2015, pp. 51–58, ISBN: 9781450328838.

[55] T. Hellström and S. Bensch, "Understandable robots - what, why, and how," *Paladyn, Journal of Behavioral Robotics*, vol. 9, no. 1, pp. 110–123, 2018.

[56] S. Li, W. Sun, and T. Miller, "Communication in human-agent teams for tasks with joint action," in *Proceedings of the 2015 International Conference on Coordination, Organizations, Institutions, and Norms in Agent Systems XI*, ser. COIN@AAMAS/IJCAI'15, Buenos Aires, Argentina: Springer, 2015, pp. 224–241, ISBN: 9783319426907.

[57] R. W. Wohleber, K. Stowers, J. Y. Chen, and M. Barnes, "Effects of agent transparency and communication framing on human-agent teaming," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 3427–3432.

[58] M. Göbelbecker, T. Keller, P. Eyerich, M. Brenner, and B. Nebel, "Coming up with good excuses: What to do when no plan can be found," in *Twentieth International Conference on Automated Planning and Scheduling*, 2010.

[59] S. Sreedharan, S. Srivastava, D. Smith, and S. Kambhampati, "Why can't you do that hal? explaining unsolvability of planning tasks," in *International Joint Conference on Artificial Intelligence*, 2019.

[60] J. Peters, D. Janzing, and B. Schlkopf, *Elements of Causal Inference: Foundations and Learning Algorithms.* The MIT Press, 2017, ISBN: 0262037319.

[61] M. Scutari, "Learning bayesian networks with the bnlearn R package," *Journal of Statistical Software*, vol. 35, no. 3, pp. 1–22, 2010.

[62] M. J. Vowels, N. C. Camgoz, and R. Bowden, "D'ya like dags? a survey on structure learning and causal discovery," *ACM Computing Surveys*, 2022.

[63] D. Margaritis, "Learning bayesian network model structure from data," Ph.D. dissertation, Carnegie Mellon University, 2003.

[64] X. Zheng, B. Aragam, P. K. Ravikumar, and E. P. Xing, "Dags with no tears: Continuous optimization for structure learning," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[65] T. S. Verma and J. Pearl, "Equivalence and synthesis of causal models," in *Probabilistic and Causal Inference: The Works of Judea Pearl*, 2022, pp. 221–236.

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

[67] K. C. Stocking, A. Gopnik, and C. Tomlin, "From robot learning to robot understanding: Leveraging causal graphical models for robotics," *Proceedings of the 5th Conference on Robot Learning*, vol. 164, pp. 1776–1781, 2022.

[68] 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,” *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4776–4782, 2021.

[69] A. A. Bhat, V. Mohan, G. Sandini, and P. G. Morasso, “Humanoid infers archimedes’ principle: Understanding physical relations and object affordances through cumulative learning experiences,” *Journal of the Royal Society Interface*, vol. 13, no. 120, 2016.

[70] D. Song, K. Huebner, V. Kyrki, and D. Kragic, “Learning task constraints for robot grasping using graphical models,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1579–1585, 2010.

[71] 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,” *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8081–8086, 2020.

[72] M. Diehl, C. Paxton, and K. Ramirez-Amaro, “Automated generation of robotic planning domains from observations,” *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6732–6738, 2021.

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

[74] B. Seegebarth, F. Müller, B. Schattenberg, and S. Biundo, “Making hybrid plans more clear to human users — a formal approach for generating sound explanations,” *International Conference on Automated Planning and Scheduling*, ICAPS’12, pp. 225–233, 2012.

[75] V. Raman and H. Kress-Gazit, “Explaining impossible high-level robot behaviors,” *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 94–104, 2013.

[76] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi, “Neural motifs: Scene graph parsing with global context,” *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5831–5840, 2018.

[77] D. Altan and S. Sariel, "Probabilistic failure isolation for cognitive robots," in *FLAIRS Conference*, 2014.

[78] 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, 2022.

[79] H. Wang, I. Rish, and S. Ma, "Using sensitivity analysis for selective parameter update in bayesian network learning," *Association for the Advancement of Artificial Intelligence (AAAI)*, 2002.

[80] M. Diehl and K. Ramirez-Amaro, "A causal-based approach to explain, predict and prevent failures in robotic tasks," *Conditionally accepted with minor revisions to Robotics and Autonomous Systems (RAS)*, 2022.

[81] D. Colombo and M. H. Maathuis, "Order-independent constraint-based causal structure learning," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3741–3782, Jan. 2014, ISSN: 1532-4435.