Proceedings of the 4th International Workshop on Active Inference (IWAI 2023), 13-15 September 2023, Ghent, Belgium

Contextual Qualitative Deterministic Models for Self-Learning Embodied Agents

Jan Lemeire[1,2][0000−0002−2106−448X], Nick Wouters[2], Marco Van Cleemput[1], and Aron Heirman[2]

[1] Dept. of Industrial Sciences (INDI), Vrije Universiteit Brussel (VUB), Pleinlaan 2, B-1050 Brussels, Belgium
[2] Dept. of Electronics and Informatics (ETRO), Vrije Universiteit Brussel (VUB), Pleinlaan 2, B-1050 Brussels, Belgium
jan.lemeire@vub.be

Abstract. This work presents an approach for embodied agents that have to learn models from the least amount of prior knowledge, solely based on knowing which actions can be performed and observing the state. Instead of relying on (often black-box) quantitative models, a qualitative forward model is learned that finds the relations among the variables, the contextual relations, and the qualitative influence. We assume qualitative determinism and monotonicity, assumptions motivated by human learning. A learning and exploitation algorithm is designed and demonstrated on a robot with a gripper. The robot can grab an object and move it to another location, without predefined knowledge of how to move, grab or displace objects.

Keywords: Autonomous robots · Developmental learning · Open-ended learning · Qualitative Models.

1 Introduction

We believe that self-learning capabilities are crucial for fully autonomous agents. With the right learning architecture, agents will be able to adapt to new, unseen, and uncontrolled, open environments. They can discover knowledge autonomously, with no need for external supervision, while also having the capability to redefine their own behavior in case of unexpected perturbations.

We developed a qualitative approach, that allows for a high level of abstraction. It is therefore effective, data-efficient, relates to symbolic reasoning, and provides explainability. The idea of self-learning agents in general, is to start with an empty brain that doesn't contain any prior knowledge, apart from a generic learning architecture. With this developmental learning philosophy in mind, our agent aims to achieve the following goals. Firstly it wants to formalize the effect of its actions on the world in a forward model. It does so by interacting with the world, and relating its observations to its actions. Secondly, it will exploit the learned model to make effective action plans to achieve desired goal states. In other words, the agent will learn to manipulate its environment through its directly controllable actuators. In order to do this, our agent's learning architecture needs algorithms for exploration, learning, and exploitation.

As opposed to monolithic black box models, such as neural networks, our approach is based on explicitly modeling the structure and qualitative properties of the system. Some examples of qualitative relations that our system will learn are the following: "positive motor input gives an increase of x- and y-coordinate if the robot's orientation is North", "if I touch a wall, I cannot move further, but can move in the other direction", or "If I touch an object that is located North of me, and if I move North, the object's position will change". Those examples illustrate the plausibility of the assumption of qualitative determinism on which our approach relies: in our world, things happen roughly deterministically. Certainly, if we only look at the direction of changes in variables and not the quantity of the changes.

The contributions of this work are the explicit modeling of context, the identification of the context and the graph describing the relations among the variables, and the exploitation algorithm based on the graph and context.

First, the related work is discussed. The three assumptions are given in the subsequent section. Then we present the experimental setup before defining the model class. In the last 2 sections, the learning and exploitation algorithms are given and the experimental results are shown.

2 Related work

There is extensive work on qualitative models [3], [6], in which the advantages over pure quantitative models are extensively discussed and proven. Bratko et al [2, 13] proposed qualitative models for robotic control. They are based on a small set of variables, while we try to learn networks over a large set of variables. Also, with our representation of context, we enable high-level reasoning. The work of Mugan et al. [10] tackles the same problem as in this paper. They also employ qualitative dynamic Bayesian networks. But they use a probabilistic approach and turn the networks into MDPs to use the model for solving tasks. In Section 7, we show how that model can directly be used to determine effective actions. Mugan's quantitative space allows more qualitative values than just the sign. The space is partitioned by so-called landmarks, which resemble our contextual partitioning.

There is quite extensive work on algorithms for causal structure learning, such as the PC algorithm and its variants [11]. However, these algorithms heavily rely on the probabilistic nature of the relations since they rely on some type of faithfulness, which is violated in the presence of deterministic relations [9].

The advantage of explicitly adding context to the models has been studied by [12]. Applied to Bayesian networks this results in context-specific independencies [1], which come from our contextual edges. Our representation is based on the work of [5].

The field of *Active Inference*[7] is also concerned with the self-learning of embodied agents. The theoretical foundations are based on probabilistic models, while we challenge their necessity for the robotic settings on which we focus. Many approaches for active inference are based on (deep) neural networks, e.g., Çatal et al. [4]. These are monolithic black-box models, while the presented approach is based on *explicit modeling of the qualitative properties*, which can then be exploited for reasoning about plans and linking them to symbolic approaches.

3 Assumptions

We assume that the system under study can be described by a *limited set of piece-wise deterministic monotonic functions* defined over variables that are observed or derived:

- *piece-wise monotonic functions*: the relations among variables are primarily monotonous. At certain points/boundaries/constraints (called landmarks by [10]), the monotonicity might be 'broken' and a new 'tone' starts. We say that the state space is divided into **subspaces**.
- *limited*: almost every continuous mathematical function can be split into pieces of monotonicity, but we assume that for the functions of our models, the number of pieces is limited.
- we assume that the set of observed variables is sufficient to characterize the state of the system. This will also become possible by relying on *derived variables*. Derived variables are defined over observed variables or other derived variables.
- we assume *qualitative determinism*. Although this will be relaxed later. We plan to add a *don't know*-value for variables and function outputs. This value can also be used in regions where the variable's sign changes and there is some uncertainty.

4 Experimental setup

We will perform experiments on a simulated robot. The robot has 4 motors: one for the right wheel (m_R), one for the left wheel (m_L), one to close the gripper ($close$), and one to lift the gripper ($lift$). The wheel motors can only turn in one direction (to drive forward). The robot knows its position (x and y) and its orientation (or). It also has a camera for the position of an object (obx, oby and obz) and a sensor for detecting whether an object is held ($hold$).

The goal is that the robot explores the effect of its actions and learns a model such that it can grab an object and move it to another location. This setup is similar to the setup used by Mugan and Kuipers [10].

5 Contextual Qualitative Deterministic Causal Models

Here we define the proposed model class.

5.1 Problem definition

Similar to the Markov Decision Processes (MDPs), the problem is defined by a tuple $\langle S, A, T \rangle$, where S is a set of states that are observed, A a set of actions, and T is a transition model, which, in our approach, is a deterministic function $S' = T(S, A)$. As opposed to MDPs, we do not have a reward signal R; the agent will learn a model for T by intrinsic motivation.

5.2 Model definition

A model is defined over the action variables from A, the previous state $\_S$ (the underscore is used for previous state variables) and the new state S. State variables are directly observed or calculated from other variables. The latter we call **derived variables**. For each state variable, we add the derived variable $ds = s - \_s$, which measures the change of variable s and can be regarded as an approximation of the time derivative. Note that we use capitalized names for vectors and small letters for single variables.

The model consists of two parts: the description of the relations among the variables and the nature of these relations. Similar to a dynamic Bayesian network, the relations among the variables are modeled by a Directed Acyclic Graph (DAG), which we will interpret causally: the orientation of the edges represents the causal influence. Variables A and $\_S$ are input or root variables. Only the variables of S have incoming edges. The parents of variable s are denoted with $Pa(s)$.

By assuming determinism, all (dependent) state variables could be expressed as a function of the (free) input variables and the previous state. However, since we want to have simple relations, we want to find an order in which all dependent variables can be calculated from input or other dependent variables such that the relations are 'simple': each variable has a minimum of parents and the relations are basic qualitative functions based on the monotonicity assumption.

Once the DAG is established, the dependence of each state variable on its parents has to be established. As we are only interested in the qualitative relation, the function returns the sign of the variable.

We denote the sign of variable v by $\mathcal{Q}(v)$, which has three possible outcomes: PLUS, MINUS and ZERO, also denoted with +, - and 0. When applied to a vector, $\mathcal{Q}(V)$ returns the vector containing the signs of the vector elements. To each state variable s, a deterministic qualitative function $\mathcal{Q}(s) = QF(Pa(s))$ is determined. For the moment, we allow two types of qualitative functions. In the first case, the qualitative value of s can be determined by the qualitative values of the parent variables only, while in other cases, the quantitative values are needed. The first type is a function that only depends on the qualitative value of the independent variable and can thus be written as $\mathcal{Q}(s) = QF(\mathcal{Q}(Pa(s)))$. The function can be described by a ternary truth table. The second type is a function $\mathcal{Q}(s) = QF(Pa(s))$ that can be described by a monotone *decision function* DF such that $\mathcal{Q}(s) = Sign(DF(Pa(s)))$. The function separates the positives from the negatives, as shown in Figure 1. When the left motor is actuated more than the right motor, the robot turns to the left (the change of orientation is positive). Otherwise to the right, except if both actuations are equal, then the robot drives straight.

5.3 Representing context

So far, our model is able to model deterministic monotonic functions qualitatively. These functions are only valid in parts of the state space, which are defined by the context. For some of the state variables, different qualitative functions might apply according to the **context**, which depends on some action or state variables. Here we limit the context of a specific function to a specific range of one variable (which we call

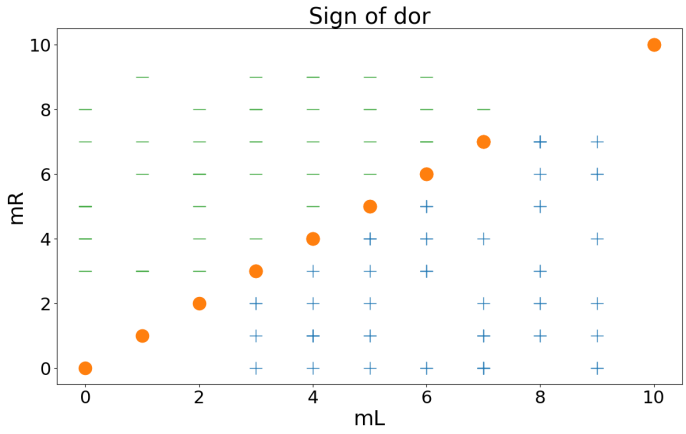

Fig. 1. How the left and right motor determine the sign of the change of the robot's orientation. This is a type 2 function.

the **context variable**): the total range of that variable is partitioned into two or more contexts.

Definition 1. *Variable c is a context variable of state variable s if its range can be subdivided into regions in which the qualitative function can be written as a truth table (Type 1) or with a decision function (Type 2).*

In each context of c, s might depend on another set of parents, or it is just the qualitative function that starts another 'tone'. An example of the latter is shown in Figure 2. The orientation (expressed in degrees) determines the sign of the change of the x-coordinate. Between 90 and -90 degrees, x changes positively, otherwise negatively. Driving forward is determined by the minimum of the left and right motor actuation ($minLR$) since the difference between both actuations results in a turn of the robot.

The edge between context variable c and state variable s is called a **context edge**. We augment the DAG with this contextual information and call it the *meta-DAG*. If some edges towards s depend on the context defined by c, they are called **contextual edges**. When drawing the DAG, we point the context edge towards these edges. The current state makes the contextual edges active or inactive. If a context edge only determines the qualitative function of s, we point it towards s. An example is given in the next section.

Figure 3 shows the meta-DAG for the example robot defined in Section 4. A few additional derived variables are added. $distx$ and $disty$ represent the distance of the robot and the object's x and y coordinate respectively. $mdist$ is the maximal value of $distx$ and $disty$.

6 The learning

In this section we give the algorithm for learning the model defined in Section 5.

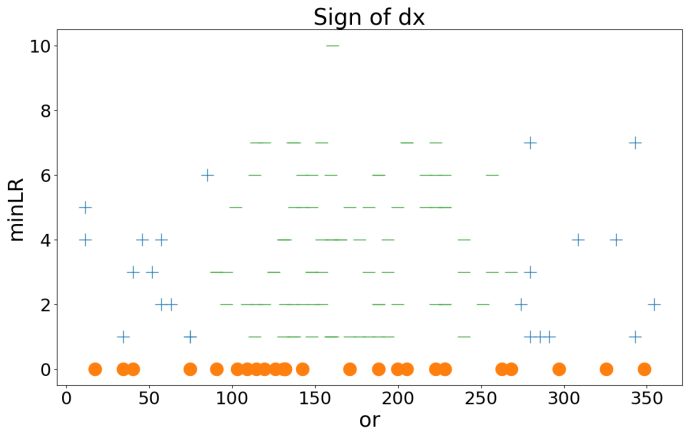

Fig. 2. The change of the x-coordinate depends on the robot's orientation or. The orientation is a context variable. It also depends on $minLR$ which is the minimum of the left and right motor actuation.

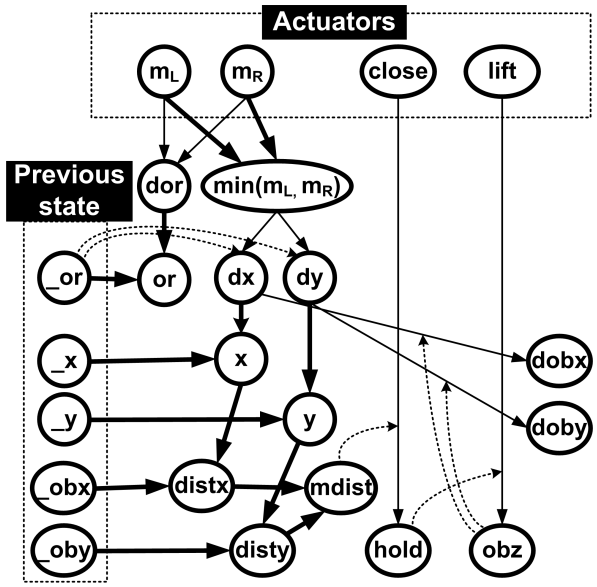

Fig. 3. The contextual qualitative model of the example robot. Contextual edges are shown with dashed lines. The thick edges are known relations coming from derived variables.

6.1 The tests

The algorithm is based on analyzing the relations among the variables by applying the following tests the observed data.

- Function **depStr** measures the dependency strength between two variables. Pearson's correlation coefficient is used for this.
- Function **condDepStr** measures the conditional dependency strength between two variables conditional on some others with Pearson's partial correlation coefficient.
- Function **isQDet** tests whether a variable can be written as a deterministic function given a set of other variables. The test checks in each context the two types of qualitative functions that are allowed:

Type 1 : the data is arranged according to the truth table (all possible sign combinations of the independent variables). A conflict in a cell happens when there are samples with different signs for the dependent variable.

Type 2 : a monotone decision function is fit on the data to separate the PLUS for the MINUS values in the space defined by the independent variables. Then it is checked whether the decision function can effectively separate the PLUS from the MINUS values. A support vector machine is trained with a linear kernel. To test the separation, we ignore the ZERO values and take a margin of 5 percent.

- Function **isContext** for context identification: data is filtered according to the range of the proposed context variable, and the test for determinism is applied. For Type 1 functions, the truth table is gradually filled by gradually enlarging the context's range. As soon as a conflict occurs, a new context starts. For Type 2 functions, the classification is retrained with a conflict to check whether this can annul the conflicts.

6.2 The learning algorithm

The goal is to construct the model: identification of the relations that form the DAG and parameterization of $QF(Pa(s))$ of each variable s. However, this is not necessary for derived variables since their functions are known by their definition. An exception is the variables indicating the change of state variables denoted with the prefix 'd'. These variables are added to the unknown variables, called target variables, while the corresponding state variables are considered to be known by their relation $s = ds + \_s$. The known edges are shown with thick lines in Figure 3.

The algorithm has to find for each target variable a set of parents that qualitatively determine the target variable. By choosing a set of potential parents for a target variable, the tests of Section 6.1 are used to determine whether it results in a possibly-contextual function of type I or type II. The potential parents are chosen in order of the correlation and partial correlation coefficients until a deterministic function is found. We start with the target variable having the highest correlation with one of the action variables. Then, additional action variables or variables from the previous state are added according to their partial correlation. If no deterministic function is found, the target variable will be reconsidered at a later stage (when other target variables have been resolved). Once a state variable is resolved, it is added to the list of action variables to select the next target variable. As such, it can serve as a parent of the other target variables.

6.3 Exploration and learning

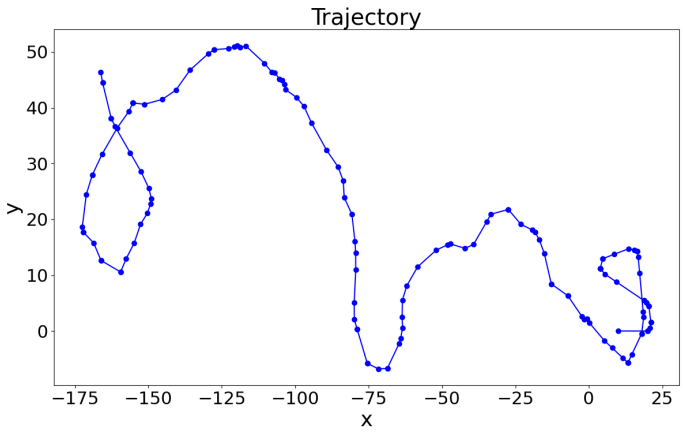

Fig. 4. The exploratory trajectory of the robot with random driving.

Du ring exploration, the robot gathers data that will be used for learning the model. At first, random inputs are given for the motors (a so-called motor 'babbling') by which the driving will be learned. This is the upper part of the model, controlling the robot's position. The exploratory trajectory, which contains 150 data points, is shown in Figure 4.

For learning how to grab and move an object, there must be data acquired in which the object is accidently grabbed and displaced. Herefore, the robot is, during its random exploration, regularly oriented towards the object and the gripper is regularly closed to make the chance of grabbing possible. The second exploratory trajectory, by which the lower part of the model involving the object is learned, is shown in Figure 5. This trajectory contains 350 points during which the object was successfully grabbed and moved 4 times.

With the collected data of 500 points, the learning algorithm correctly learns the model of Figure 3 with the correct qualitative functions.

7 Exploitation

A task is defined by a goal state in which some state variables should attain certain values. The robot has to take actions in a control loop such that the goal state is reached effectively. Algorithm 1 describes how the qualitative model is used to choose the actions to achieve the goals. Applied to our case, the robot has to travel through 5 **subspaces**: turn -> drive to object -> grab -> lift -> move. This chain is calculated backward: to move an object, the context indicates that the object should be lifted, then to lift an object, it should be grabbed, etcetera.

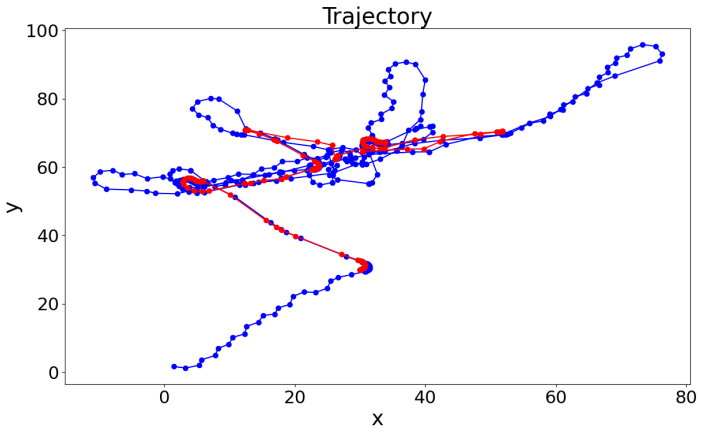

Fig. 5. The exploratory trajectory of the robot (in blue) when trying to grab the object. The trajectory of the object is shown in red.

A hierarchical plan is created: at the higher level, a path across subspaces is sought; at the lower level, a path within a subspace is calculated through a simple control loop. This corresponds to most top-down approaches for robot control [8]. Here it follows naturally from our bottom-up approach.

With the model learned in Section 6, the robot, starting from position $(0, 0)$, is assigned to grab an object at position $(30, 30)$, bring it to $(10, 40)$ and return home. In Figure 6, the paths of the robot and the grabbed object are shown.

The path of the robot might not be a straight line to the object. This is because of our qualitative approach. It makes calculations and reasoning simpler at the expense of accuracy. By accuracy, we mean that we do not calculate the exact command the robot has to drive to reach the desired goal. The qualitative information the robot is using is shown in Figure 2. It tells him in which range the orientation should be to travel in the direction of the object, but it does not tell him exactly how much the orientation should be to travel straight to the object. Once the robot is in the desired orientation, it drives straight. By doing this, the robot gets closer to the object. There comes a point in time when the robot stops getting closer to the object because of this non-exact path. The robot creates a new subgoal at this point. This subgoal is changing the orientation to another quadrant so that the robot once again is able to get closer to the object by driving straight. It keeps doing this until the desired goal is reached.

8 Conclusions

This work is part of the quest for the 'first principles' that allows self-learning. With the human example in mind, we put some thought-provoking ideas on the table. In everyday situations, probabilistic models are not needed except for the notion that there are things we don't know (yet). Modeling qualitative properties explicitly enables rea-

Algorithm 1 ReachGoal(Goal G, State S)

while $G \neq S$ **do**
  $dGS \leftarrow G - S$
  for all dgs in dGS **do**
    if dgs is non-zero **then**
      construct a list of tuples of contexts and action signs such that $\mathcal{Q}(dgs) = \mathcal{Q}(ds)$ is attained in the model
      rank all tuples on amount of context values that have to be changed with respect to the current state (lower is better)
    end if
  end for
  search for the simplest (according to the sum of ranks) combination of tuples of the lists (one tuple per dgs) so that the contexts and action signs are the same for all tuples.
  if context $\neq$ current state S **then**
    recursively execute function ReachGoal with Goal set to the wanted context
  end if
  estimate values with the right sign for the action variables (controller)
  execute actions
  update S with the new state
end while

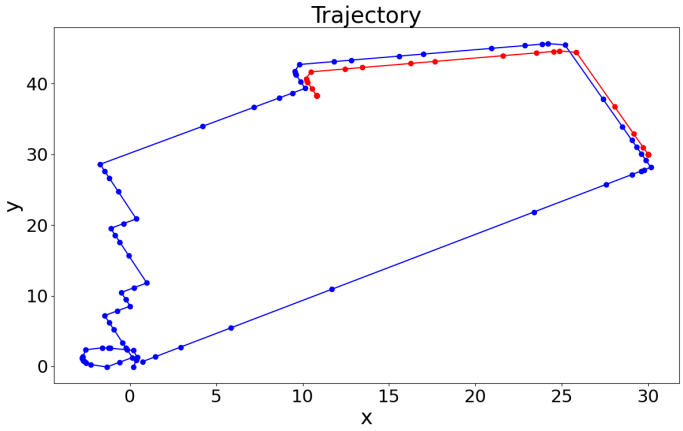

Fig. 6. The trajectory of the robot (in blue) for bringing the object from position (30, 30) to position (10, 40) and returning home. The trajectory of the object is shown in red.

soning, makes the link with top-down symbolic approaches, and is easier to learn than quantitative approaches that often require 1000s of samples. The algorithms presented in this paper showed that it is possible and resulted in promising results. Important remaining challenges are the autonomous identification of useful derived variables, effective curiosity-driven exploration and incremental learning.

References

[1] Craig Boutilier et al. "Context-Specific Independence in Bayesian Networks". In: *Uncertainty in Artificial Intelligence*. 1996, pp. 115–123.

[2] Ivan Bratko. "An assessment of machine learning methods for robotic discovery". In: *Journal of Computing and Information Technology* 16 (Jan. 2008), pp. 247–254.

[3] Ivan Bratko and Dorian Suc. "Learning Qualitative Models." In: *AI Magazine* 24 (Jan. 2004), pp. 107–119.

[4] Ozan Çatal et al. "Learning Generative State Space Models for Active Inference". In: *Frontiers in Computational Neuroscience* 14 (2020).

[5] Jukka Corander et al. "A logical approach to context-specific independence". In: *Annals of Pure and Applied Logic* 170.9 (Sept. 2019), pp. 975–992.

[6] Kenneth D. Forbus. "Chapter 9 Qualitative Modeling". In: *Foundations of Artificial Intelligence*. Vol. 3. Handbook of Knowledge Representation. Elsevier, Jan. 2008, pp. 361–393.

[7] Karl Friston, James Kilner, and Lee Harrison. "A free energy principle for the brain". In: *Journal of Physiology* 100.1-3 (July 2006), pp. 70–87.

[8] George Konidaris, Leslie Kaelbling, and Tomas Lozano-Perez. "Constructing Symbolic Representations for High-Level Planning". en. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 28.1 (June 2014). Number: 1.

[9] Jan Lemeire et al. "Conservative independence-based causal structure learning in absence of adjacency faithfulness". In: *Int. J. Approx. Reasoning* 53.9 (2012), pp. 1305–1325.

[10] Jonathan Mugan and Benjamin Kuipers. "Autonomous Learning of High-Level States and Actions in Continuous Environments". In: *IEEE Transactions on Autonomous Mental Development* 4.1 (Mar. 2012), pp. 70–86.

[11] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. 2nd. Springer Verlag, 1993.

[12] Santtu Tikka, Antti Hyttinen, and Juha Karvanen. "Identifying Causal Effects via Context-specific Independence Relations". In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019.

[13] Jure Zabkar, Ivan Bratko, and Ashok C Mohan. "Learning qualitative models by an autonomous robot". In: *22nd International Workshop on Qualitative Reasoning*. 2008, pp. 150–157.