On the Diagnosability of Actions Performed by Contemporary Robotic Systems*

Alex Mitrevski[1], **Ahmed Faisal Abdelrahman**[1], **Anirudh Narasimamurthy**[1], and **Paul G. Plöger**[1]
[1]Hochschule Bonn-Rhein-Sieg, Sankt Augustin, Germany
e-mail: {aleksandar.mitrevski, paul.ploeger}@h-brs.de,
{anirudh.narasimamurthy, ahmed.abdelrahman}@inf.h-brs.de

Abstract

When a robotic agent experiences a failure while acting in the world, it should be possible to discover why that failure has occurred, namely to diagnose the failure. In this paper, we argue that the diagnosability of robot actions, at least in a classical sense, is a feature that cannot be taken for granted since it strongly depends on the underlying action representation. We specifically define criteria that determine the diagnosability of robot actions. The diagnosability question is then analysed in the context of a handle manipulation action, such that we discuss two different representations of the action - a composite policy with a learned success model for the action parameters, and a neural network-based monolithic policy - both of which exist on different sides of the diagnosability spectrum. Through this comparison, we conclude that composite actions are more suited to explicit diagnosis, but representations with less prior knowledge are more flexible. This suggests that model learning may provide balance between flexibility and diagnosability; however, data-driven diagnosis methods also need to be enhanced in order to deal with the complexity of modern robots.

1 Introduction

When developing robotic agents, system designers are primarily concerned with the capabilities of the robot, namely with what the robot is able to do (what actions[1] it can perform) and how well it does that (measured by some predefined performance criteria). When robots are deployed in everyday environments, they are, however, bound to experience failures of different natures [Fallatah *et al.*, 2019; Gross *et al.*, 2019]. When a failure happens, it should be possible to easily identify the causes: from the perspective of the robot's user, this may increase the trust in the system [Fallatah *et al.*, 2019], while from the perspective of the robot designer, this is important so that the system can be improved. Failure diagnosis is also relevant in terms of complete autonomy and lifelong learning: if a robot has a means to diagnose itself, it could use the information for improving its interaction with the world in order to avoid such failures.

Traditional consistency-based diagnosis approaches [de Kleer and Williams, 1987; Reiter, 1987] require a nominal model of operation of a system, where the model may be discrete, continuous, or hybrid [Bouziat *et al.*, 2018; Provan, 2018]. The existence of a model allows diagnosis hypotheses to be generated as violations of the model; such hypotheses can then be analysed based on their likelihoods, or by applying sequential diagnosis that would eliminate hypotheses that are inconsistent with some observations. In the context of robot actions, the effort required for creating explicit models of execution is usually too time consuming and not very flexible; this is for instance illustrated by the naive physics model in [Kuestenmacher *et al.*, 2014]. For this reason, robot action representations often involve a learning aspect, which may be as simple as mapping some action parameters to predicted success or as complex as end-to-end visuomotor learning.

In this paper, we discuss learning-based action modelling mechanisms for modern robots and analyse their diagnosability in terms of the ability to generate hypotheses for why a robot fails when performing an action. We argue that the diagnosability of action formalisms exists on a continuum and is proportional to the modelling effort that goes into the action design; however, we also argue that modelling effort usually affects the flexibility of the representation. We ground our argument by analysing two learning-based models of a handle manipulation action for a Toyota HSR robot [Yamamoto *et al.*, 2019] - the first one learns a success model of handle grasping parameters based on sparse feedback, while the second learns a visuomotor policy according to a specified performance criterion that the robot should maximise. We then conclude with some remarks on the challenges of diagnosing robot actions and the need for either model-free diagnosis methods or strategies that learn diagnosis models in order to deal with the complexity of modern robotic systems.

We organise this paper as follows. In section 2, we discuss some criteria for robot action diagnosability as well as levels of diagnosability depending on who is able to diagnose action executions and subsequently leverage this infor-

*This work was supported by the B-IT foundation

[1]For the purposes of this paper, we partially adopt the definition of an action in [Zech *et al.*, 2019], namely we see an action as "something an agent does that was intentional under some description, that is caused by both the agent's current internal state and external percepts, is adaptive and deterministic to achieve desired effects..." Essentially, as in classical planning, we consider an action as having some preconditions and achieving some effects, but we make no assumptions about the effects being deterministic.

mation. We then briefly discuss various contemporary action modelling and learning approaches in section 3 and introduce the handle manipulation use case in section 4, such that we analyse our representations of the action in terms of the diagnosability criteria defined in section 2, as well as in terms of modelling effort and flexibility. We finally make some concluding remarks in section 5.

2 Robot Action Diagnosability

In robotics, actions are usually driven by high-dimensional or continuous inputs and may produce continuous outputs, such that the problem of diagnosing robot actions can often be seen as that of analysing the inputs as well as the mapping that produces the action outputs. Before looking at action formalisms in robotics, we define a set of criteria that determine the diagnosability of robot action representations. We also discuss different levels of diagnosability depending on what execution information is exposed and who is able to utilise it.

2.1 Criteria for Diagnosability

In order for robot actions to be diagnosable, we argue that they need to satisfy at least a few criteria: (i) abstractability, (ii) predictability, and (iii) composability. We discuss each of these criteria below.

Action abstractability determines the ease of reasoning about why a robot has chosen to execute an action in a certain manner or, equivalently, why the action was parameterised in a particular way. We define abstractability as the complexity[2] involved in finding a mapping $\mathcal{A}(a)$ for every action a in a set of actions A, which converts a to an abstract form on which reasoning can be performed. We note that $\mathcal{A}$ can be defined as an identity mapping, or it may be created alongside a parametric model of the action, as we have done in [Mitrevski *et al.*, 2020; 2017], where both a continuous and a discrete model of action execution are learned (here, the discrete model of execution represents an action abstraction).

Robot actions should also be *predictable* in the sense that small changes in the action input should result in predictably small changes in the output, namely $a(\mathbf{x}) \approx a(\mathbf{x} + \epsilon)$ for a given state $\mathbf{x}$ and noise ϵ. Predictability can be seen as imposing a smoothness criterion on actions that simplifies the exploration of the action parameter space. This is important in the diagnosis context since diagnosis may require exploring alternative action parameterisations, and predictability allows doing this more systematically.

A further aspect that is necessary for robot action diagnosability is *composability*. Given two actions a_i and a_j, the composition of these is defined as $(a_j \circ a_i)(\mathbf{x})$. A pair of composable actions a_i and a_j is generally preferred to a meta-action $\hat{a}$ that combines the two since composability allows a robot to reuse actions for accomplishing different tasks, but more importantly for diagnosis, action decomposition makes it possible to assign blame more accurately. In other words, using an action $\hat{a}$ rather than a set of composable actions complicates the analysis of failure causes since all underlying components are then equally suspect. Composability at a very low level of abstraction can also be detrimental to diagnosis, however, due to an overly granular structure of the action components.

[2] In this context, complexity is a qualitative attribute that encompasses various notions including representation, computation, and amenability to reasoning.

2.2 Levels of Diagnosability

As mentioned in section 1, diagnosability of actions may serve different purposes, namely it could be used by robot designers to improve the system, help robot users understand and gain trust in the robot, but also allow robots themselves to reason about their execution. These different uses of action diagnosis can also be seen as defining three levels of action diagnosability.

The first level of diagnosability is rather rudimentary, allowing system designers to analyse failed executions in order to identify failure causes and modify the system so that similar failures can be subsequently avoided. This level of diagnosability can be achieved by (i) making the robot's state as explicit as possible and (ii) extensive data logging, as we have done in our earlier work [Mitrevski *et al.*, 2018].

A second diagnosability level allows robot users to understand why a robot has made a certain decision. This level of diagnosability is related to the abstractability criterion defined above, as being able to create an abstraction of the action mapping is a prerequisite for generating human-understandable explanations of the robot's decisions during execution.

The third level of diagnosability is that which allows a robot itself to reason about its execution so that it can improve how it acts. This level of diagnosability requires a robot to be able to discover causal relations between its action parameterisations and the action outcomes. Extracting causal relations is also important when a robot performs multiple actions in succession, as the action that fails is not always the one that has caused the failure.

It should be mentioned that the diagnosability levels do not necessarily build on each other, particularly not when considering the third level, which may be embedded independently of the other two.[3]

3 Robot Action Representations

A traditional model of actions in robotics comes from the planning domain, where an action a is described by a set of preconditions P that need to hold for the action to be performed and effects E that are expected to become true after the execution of the action. A Markov Decision Process (MDP) is another common representation of robot behaviour, which is defined as a tuple $\mathcal{M} = (S, A, \mathcal{T}, \mathcal{R}, s_0)$, where S is a set of states, A is a set of primitive actions, $\mathcal{T} : S \times A \rightarrow S = P(s'|s, a)$ is a transition model, $\mathcal{R} : S \times A \rightarrow \mathbb{R}$ is a reward function, and s_0 is the initial state. When modelling robots with an MDP, robot behaviour is defined by a policy $\pi : S \rightarrow A$, such that the objective is to find an optimal policy π^*, which maximises the agent's expected return $E\left[\sum_{t=0}^{T} \mathcal{R}(s_t, a_t)\right]$. Optimal policies are most often found using reinforcement learning.

In most practical applications, $\mathcal{T}$ is not known since it is too difficult to model explicitly. For that reason, a policy is usually learned using a model-free method, where a robot

[3] In other words, a robot may be able to diagnose its own failures and learn from them, but whether this knowledge is helpful for the designer or the user depends on the level of abstraction at which self-diagnosis is performed.

learns from experiences it collects while acting in its environment. Learning is usually done in simulation, as most learning algorithms require a large number of experiences for finding a good policy, or are unsafe for learning with a real system.

In this section, we briefly discuss various state-of-the-art learning-based action representations, many of which use the MDP formalisation and learn actions in a reinforcement learning setup. We classify the learned action models into two categories depending on the nature of the learned policy.

3.1 Density-Based and Symbolic Models

We first discuss representations in which action parameters are represented in a symbolic form or using a probability density function.

[Stüber *et al.*, 2018] present a learning-based method that allows a robot to choose how to push an object, and also predict the motion of the object based on the chosen action. The push action is represented by robot-object and object-environment contacts, such that learning the action amounts to learning the contact model and an object motion model. [Kazhoyan and Beetz, 2019] introduce a system that parameterises actions in a plan by projecting parameterised versions of the actions in a simulator; this has the purpose of checking for successful execution and optimising some predefined execution criterion. Here, action grounding rules are augmented by a set of failure cases in a given scenario and by a set of predefined functions for setting the action parameters. [Bozcuoğlu *et al.*, 2019] learn an execution success prediction model, which can be used for sampling robot positions that are likely to lead to a successful execution of an action. The model is learned from execution experiences, which are first clustered and then combined into two Gaussian Mixture Models - one for the successful executions and one for the failed executions; the two models are then combined when making success predictions. [Wang and Kroemer, 2019] present a method based on which a robot can improve the execution robustness of demonstrated tasks by leveraging contact information that is collected during repetitions of the task. The objective is to sequence skills, but also infer new skills, in a way that maximises task performance and provides the ability to recover from failures. [Karapinar and Sariel, 2015] describe an application of inductive logic programming (ILP) where the objective is to discover failure rules for action execution that explain a set of training examples. The rules learned by the algorithm include various qualitative object aspects (colour, shape, category) as well as numerical attributes, such as the location of an object in the world frame. [Ames *et al.*, 2018] discuss probabilistic planning with parameterised skills (i.e. actions that also depend on some parameters) and a procedure for learning representations of the preconditions and effects of such skills. Here, action parameters are sampled and then clustered using DBSCAN for the purpose of learning symbols that appear in the preconditions and effects; precondition-effect relations are then created from the cluster transitions.

3.2 Neural Network-Based Models

The second category of actions we discuss use a neural network-based policy, which is learned either in an end-to-end fashion or with some guidance (either through the architecture or through human demonstrations).

[Liu, 2020] present a learning-based grasping method that takes task context and object affordances into account when selecting a grasp. The task context is described by four pieces of information: (i) a description of the task for which the object is grasped (pouring, handing over, etc.), (ii) a state description of the object to be grasped (hot, empty, etc.), (iii) the affordance of the grasped object part, and (iv) the material of the grasped part. [Lee *et al.*, 2019] introduce a multimodal network-based state representation that combines image inputs, force measurements, and robot state information. The joint representation formed from these combined inputs is then used as input to a learned continuous policy for a peg insertion task. [Tremblay *et al.*, 2018] discuss a multi-network system that is used for learning (from observation) and generating plans for pick-and-place tasks. In particular, the system includes four networks, all of which are trained on synthetic data and are used in a sequential manner at runtime: (i) an object detection network, (ii) a network for recognising object relations, (iii) a network that essentially produces a task plan based on the object relations, and (iv) a plan execution network. [Hermann *et al.*, 2020] describe an algorithm for creating a curriculum for an RL agent that learns actions based on a small sample of human demonstrations, which provide initial guidance to the learning agent. A network-based diagonal Gaussian policy is learned here, such that the input to the policy network is composed of two segments - a convolutional part processes input images and a state segment takes in a robot state representation.

4 Action Use Case: Handle Manipulation

In this section, we present two representations of a handle manipulation action for domestic scenarios, which is useful for tasks in which a robot needs to open doors or drawers. The representations are mapped to the two categories in section 3. In our first representation, we generate random action parameters and use human feedback for evaluating those; the samples are then represented by a density model that is subsequently used for sampling execution parameters. In our second representation, we model handle manipulation by an MDP and find an optimal policy using a network-based reinforcement learning algorithm. We then analyse these two representations in terms of the diagnosability criteria defined in section 2.

4.1 Density-Based Model

Our density-based representation of the handle manipulation action, which is illustrated in Fig. 1 in the context of a kitchen drawer, splits the action execution into substeps. The robot first detects the handle[4] and determines its pose using a point cloud. Once the handle pose is detected, we sample execution parameters, such that we parameterise the action by the distance between the robot's end effector and the center of the handle. Once these parameters are sampled, the robot moves towards the handle using a demonstrated arm trajectory[5] and pulls the handle after grasping. For learning, we sample action parameters from a uniform

[4]Detection is performed by a standard network-based object detector.

[5]We represent demonstrated trajectories using motion primitives, and use whole-body motion in case the handle is initially outside the robot's reach.

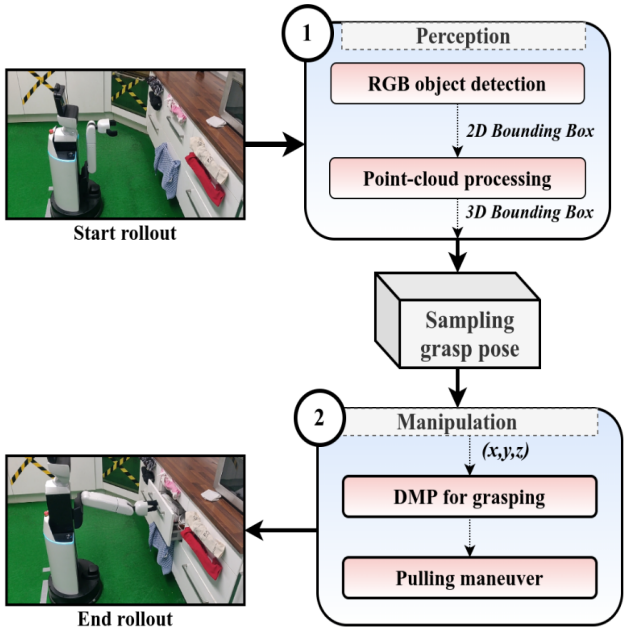

Figure 1: General framework of the density-based execution model

distribution within the bounding box of the detected handle, namely

$$\Delta\mathbf{p} = \begin{pmatrix} \Delta x \sim U(x_{\min}, x_{\max}) \\ \Delta y \sim U(y_{\min}, y_{\max}) \\ \Delta z \sim U(z_{\min}, z_{\max}) \end{pmatrix} \tag{1}$$

The executions are labelled as successful or unsuccessful depending on whether the robot succeeds to grasp the handle and subsequently open the drawer. The parameters of the successful executions S are then used for learning a success distribution, which we represent by a Gaussian distribution $\mathcal{N}(\mu, \Sigma)$. We learn the parameters $\theta = (\mu, \Sigma)$ using maximum likelihood estimation under the assumption that the samples are independent:

$$L(\theta) = P(S|\theta) = \prod_{i=1}^{N} P(\Delta\mathbf{p}_i|\theta) \tag{2}$$

This results in the usual estimates for the sample mean and covariance:

$$\begin{aligned} \mu &= \frac{1}{N}\sum_{i=1}^{N} \Delta\mathbf{p}_i \\ \Sigma &= \frac{1}{N-1}\sum_{i=1}^{N} (\Delta\mathbf{p}_i - \mu)(\Delta\mathbf{p}_i - \mu)^T \end{aligned} \tag{3}$$

During execution, the robot then samples parameters from the learned success distribution, namely

$$\Delta\mathbf{p} \sim \mathcal{N}(\Delta\mathbf{p}|\mu, \Sigma) \tag{4}$$

4.2 Network-Based Policy

Our reinforcement learning-based model of the handle manipulation action uses a network-based policy. Both the state and the action space are continuous: the state space is represented by the measured joint angles of the robot and the distance from the robot's end effector to an estimated pose of the handle, while the action space is given by torque commands for the individual joints. To deal with this continuous action space, the output of the network represents a diagonal Gaussian policy.

To train our robot, we use the MuJoCo-based[6] DoorGym environment [Urakami *et al.*, 2019] and also adopt its reward function:

$$r_t = -a_0 d_t - a_1 \log(d_t + \alpha) - a_2 o_t - a_3 \|\mathbf{u}_t\| + a_4 \varphi_t + a_5 \psi_t \tag{5}$$

which rewards the robot for minimising the distance d_t to the handle and the difference in orientation o_t between the end effector and an optimal orientation for grasping the handle, while also keeping the applied torques u as low as possible. The function additionally rewards opening a door (specified by the door angle φ_t) and pushing down on a handle (for door levers, specified by the handle angle ψ_t). Here, $a_0 - a_5$ are predefined constants. An illustration of our robot in the simulated environment is given in Fig. 2.

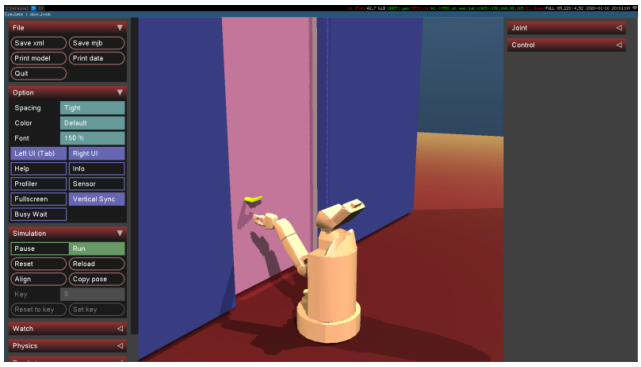

Figure 2: The HSR in the learning simulation

We use the standard Proximal Policy Optimisation (PPO) algorithm [Schulman *et al.*, 2017] for training, which is an on-policy method that controls the amount by which the parameters of the policy network change between two iterations of the algorithm, thereby preventing drastic policy updates.

4.3 Diagnosability of the Handle Manipulation Action

We now analyse our two representations of the handle manipulation actions in terms of the diagnosability criteria and diagnosability levels defined in section 2. We particularly contrast the representations in terms of the criteria in order to identify the aspects that make them more or less compliant with those criteria.

Abstractability Neither of the representations include an explicit abstractability mapping, but the density-based model can be abstracted more directly than the network-based policy. Since we map action parameters to predicted execution success, an abstraction of the success model, or alternatively, failure rules, could also be learned. An abstraction of the network-based policy could be created as well, but this may require going through a process similar to that of learning our density-based model, except that the purpose here would be to extract knowledge about the learned policy rather than learning the policy itself.

[6] http://www.mujoco.org

Predictability In terms of predictability, we consider both representations to be on a similar level since they are both governed by a model that has smoothness imposed by design (a Gaussian success model and a diagonal Gaussian policy respectively). Additionally, both representations have to process visual input in order to detect the handle pose, so they would be equally affected by input noise; however, as the network-based policy is trained purely in simulation, domain shift will have an added effect that may be difficult to predict.

Composability Our density-based model has a high-level compositional structure, since the execution of the action is split into different subactions; this contributes to the diagnosability of the action, since the execution of each individual step can be analysed in isolation (additionally, our action is designed so that if one of the subactions fails, the execution will not continue unless some recovery is possible). The primitive actions of the network-based policy represent joint motions, so we can say that the policy has a compositional structure at a low level; however, since the robot's behaviour is governed by a monolithic policy, blame can only be assigned at the primitive level, which would produce potentially uninformative diagnoses.

Diagnosability Levels In relation to the abstractability criterion, both representations of the handle manipulation action can be seen as having first-level diagnosability since failed executions can be analysed based on the action inputs (handle pose detections) and the decisions made based on those. For the density-based representation, we also aimed to incorporate some aspects of third-level diagnosability; in particular, given its success model, the robot is in principle able to predict the outcomes of different parameterisations and subsequently verify whether the actual outcomes match the predictions.

Modelling Effort and Flexibility It is also important to analyse the two representations in terms of the effort that goes into action design, but also in terms of the flexibility by what is learned. The composability, which we argued is an inherent part of the action, is manually specified rather than discovered by the robot itself. Explicitly constraining the success density to a particular family of distributions also constitutes modelling knowledge. The fact that the robot experiences are human-labelled is another aspect that potentially reduces the action's flexibility.[7]

The network-based policy, on the other hand, includes less prior information and can thus be seen as more flexible. This, however, does not mean that no prior knowledge is encoded there as well; in particular, the input to the policy network combines the vision-based input with the robot's state, which essentially constitutes a prior. The reward function that defines the robot's behaviour is a model in itself, as is the nature of the policy represented by the policy network. In fact, as demonstrated by [Jonschkowski and Brock, 2015], using prior knowledge is key to learning robot policies efficiently.

[7] Although it should be mentioned that the human labelling is done for convenience and could just as well be automated, either in simulation or using some aspect of self-supervision.

4.4 Challenges in Diagnosing Contemporary Robots

The above comparison of action representations in the handle manipulation context points to various challenges for diagnosing contemporary robots. First of all, diagnosis requires a suitable level of representational abstraction, but it is often unclear what constitutes an appropriate level of abstraction. Related to this, robot actions often have a natural hierarchical structure, but how to leverage this structure without sacrificing the flexibility of learning algorithms is an open problem, particularly when dealing with high-dimensional state spaces, such as visual representations. Similarly, action representations can also benefit from prior knowledge about problems, which is also beneficial for diagnosis since priors are directly proportional to predictable behaviour; however, too much prior knowledge can be detrimental to flexibility and generalisation. Finally, the existence of action representations that are significantly different from a conceptual point of view complicates the development of robot diagnosis methods, since a distinct set of methods is generally required for dealing with the specific details of each representation.

5 Summary and Conclusions

In this paper, we took a critical look at the diagnosability of robot actions that are designed using various common action representations, which we categorised into symbolic/density-based and neural network-based. We also defined criteria that are important for diagnosability of robot actions, namely abstractability, predictability, and composability, and introduced three levels of diagnosability depending on whether the robot designer, robot user, or the robot itself benefit from the diagnosis information. We then looked at two representations of a handle manipulation action belonging to the two action representation categories, and finally discussed some conceptual challenges of diagnosing robot actions.

We conclude with some remarks on what we believe to be promising avenues for improving the diagnosability of robot actions. The models utilised by traditional diagnosis methods do not seem to be scalable and flexible enough to deal with the complexity and open-endedness of robot problems, but learning execution abstractions can be useful for improving the diagnosability of robot actions. Furthermore, learning composable structures as in [Ames *et al.*, 2018] or [Tremblay *et al.*, 2018], as well as predictive models as in [Stüber *et al.*, 2018], can improve the self-supervision capabilities of robots, which is useful for predictability and achieving the third level of diagnosability that we discussed in this paper. Finally, sophisticated testing strategies, such as [Zhang *et al.*, 2018] are also relevant in the diagnosis context, particularly for robots that rely on complex visuomotor policies, since they may contribute to identifying unforeseen problems with a policy.

References

[Ames *et al.*, 2018] B. Ames, A. Thackston, and G. Konidaris. Learning Symbolic Representations for Planning with Parameterized Skills. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 526–533, 2018.

[Bouziat *et al.*, 2018] V. Bouziat, X. Pucel, S. Roussel, and L. Travé-Massuyès. Preferential Discrete Model-based Diagnosis for Intermittent and Permanent Faults. In *29th International Workshop on Principles of Diagnosis (DX)*, 2018.

[Bozcuoğlu *et al.*, 2019] A. K. Bozcuoğlu, Y. Furuta, K. Okada, M. Beetz, and M. Inaba. Continuous modeling of affordances in a symbolic knowledge base. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5452–5458, 2019.

[de Kleer and Williams, 1987] J. de Kleer and B. C. Williams. Diagnosing Multiple Faults. *Artificial Intelligence*, 32:97–130, Apr. 1987.

[Fallatah *et al.*, 2019] A. Fallatah, J. Urann, and H. Knight. The Robot Show Must Go On: Effective Responses to Robot Failures. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 325–332, 2019.

[Gross *et al.*, 2019] H. Gross, A. Scheidig, S. Müller, B. Schütz, C. Fricke, and S. Meyer. Living with a Mobile Companion Robot in your Own Apartment - Final Implementation and Results of a 20-Weeks Field Study with 20 Seniors. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2253–2259, 2019.

[Hermann *et al.*, 2020] L. Hermann, M. Argus, A. Eitel, A. Amiranashvili, W. Burgard, and T. Brox. Adaptive Curriculum Generation from Demonstrations for Sim-to-Real Visuomotor Control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[Jonschkowski and Brock, 2015] R. Jonschkowski and O. Brock. Learning State Representations with Robotic Priors. *Autonomous Robots*, 39(3):407–428, Oct. 2015.

[Karapinar and Sariel, 2015] S. Karapinar and S. Sariel. Cognitive Robots Learning Failure Contexts through Real-World Experimentation. *Autonomous Robots*, 39(4):469–485, Dec. 2015.

[Kazhoyan and Beetz, 2019] G. Kazhoyan and M. Beetz. Executing Underspecified Actions in Real World Based on Online Projection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5156–5163, 2019.

[Kuestenmacher *et al.*, 2014] A. Kuestenmacher, N. Akhtar, P. G. Plöger, and G. Lakemeyer. Towards Robust Task Execution for Domestic Service Robots. *Journal of Intelligent & Robotic Systems*, 76(1):5–33, 2014.

[Lee *et al.*, 2019] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg. Making Sense of Vision and Touch: Self-Supervised Learning of Multimodal Representations for Contact-Rich Tasks. In *International Conference on Robotics and Automation (ICRA)*, pages 8943–8950, 2019.

[Liu, 2020] Daruna A. Chernova S. Liu, W. CAGE: Context-Aware Grasping Engine. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[Mitrevski *et al.*, 2017] A. Mitrevski, A. Kuestenmacher, S. Thoduka, and P. G. Plöger. Improving the Reliability of Service Robots in the Presence of External Faults by Learning Action Execution Models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4256–4263, 2017.

[Mitrevski *et al.*, 2018] A. Mitrevski, S. Thoduka, A. Ortega Sáinz, M. Schöbel, P. Nagel, P. G. Plöger, and E. Prassler. Deploying Robots in Everyday Environments: Towards Dependable and Practical Robotic Systems. In *29th Int. Workshop Principles of Diagnosis DX'18*, 2018.

[Mitrevski *et al.*, 2020] A. Mitrevski, P. G. Plöger, and G. Lakemeyer. Representation and Experience-Based Learning of Explainable Models for Robot Action Execution. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. To appear.

[Provan, 2018] G. Provan. Diagnosing Hybrid Systems using Consistency-Based Methods. In *29th Int. Workshop Principles of Diagnosis DX'18*, 2018.

[Reiter, 1987] R Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32(1):57–95, Apr. 1987.

[Schulman *et al.*, 2017] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347, 2017.

[Stüber *et al.*, 2018] J. Stüber, M. Kopicki, and C. Zito. Feature-Based Transfer Learning for Robotic Push Manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5643–5650, 2018.

[Tremblay *et al.*, 2018] J. Tremblay, T. To, A. Molchanov, S. Tyree, J. Kautz, and S. Birchfield. Synthetically Trained Neural Networks for Learning Human-Readable Plans from Real-World Demonstrations. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5659–5666, 2018.

[Urakami *et al.*, 2019] Y. Urakami, A. Hodgkinson, C. Carlin, R. Leu, and P. Rigazio, L. Abbeel. DoorGym: A Scalable Door Opening Environment And Baseline Agent. *CoRR*, abs/1908.01887, 2019.

[Wang and Kroemer, 2019] A. S. Wang and O. Kroemer. Learning Robust Manipulation Strategies with Multimodal State Transition Models and Recovery Heuristics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1309–1315, 2019.

[Yamamoto *et al.*, 2019] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase. Development of Human Support Robot as the research platform of a domestic mobile manipulator. *ROBOMECH Journal*, 6:1–15, 2019.

[Zech *et al.*, 2019] P. Zech, E. Renaudo, S. Haller, X. Zhang, and J. Piater. Action representations in robotics: A taxonomy and systematic classification. *The International Journal of Robotics Research*, 38(5):518–562, 2019.

[Zhang *et al.*, 2018] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid. DeepRoad: GAN-Based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pages 132–142, 2018.