

Ontology-Assisted Generalisation of Robot Action Execution Knowledge

Alex Mitrevski^{†§}, Paul G. Plöger[†], and Gerhard Lakemeyer[‡]

Abstract—When an autonomous robot learns how to execute actions, it is of interest to know if and when the execution policy can be generalised to variations of the learning scenarios. This can inform the robot about the necessity of additional learning, as using incomplete or unsuitable policies can lead to execution failures. Generalisation is particularly relevant when a robot has to deal with a large variety of objects and in different contexts. In this paper, we propose and analyse a strategy for generalising parameterised execution models of manipulation actions over different objects based on an object ontology. In particular, a robot transfers a known execution model to objects of related classes according to the ontology, but only if there is no other evidence that the model may be unsuitable. This allows using ontological knowledge as prior information that is then refined by the robot’s own experiences. We verify our algorithm for two actions - grasping and stowing everyday objects - such that we show that the robot can deduce cases in which an existing policy can generalise to other objects and when additional execution knowledge has to be acquired.

I. INTRODUCTION

When acting in everyday environments, autonomous robots need to be able to minimise the possibility of failures during execution. Failures are, however, rather likely to occur when a robot generalises its behaviour to scenarios unseen during learning, often because the robot is not even aware that its current knowledge may be insufficient or even inappropriate for performing a particular action. To deal with this problem, it is necessary to have a strategy based on which experiences of attempted generalisations are used for determining when the existing knowledge can be reused and when additional knowledge has to be acquired. This, in turn, requires a structured knowledge base that a robot can adapt as it incorporates more knowledge about its own actions.

One well-known strategy for knowledge representation and generalisation is using an ontology [1], [2], [3], [4], which allows encoding knowledge about objects and more generally about environments. Ontologies include relationships between objects, which can be informative about when an action that is useful in one context can also be useful in another context.¹ Ontological models are, however, static in

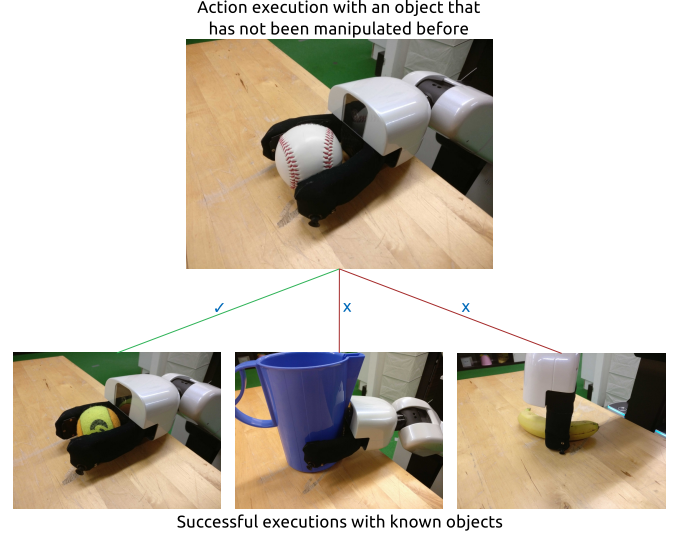


Fig. 1: When generalising execution knowledge between objects, a robot should be able to consider (i) any known object relations and (ii) previous generalisation experiences that provide information about the (un)suitability of the existing execution knowledge in a given context

general; this makes them suitable for representing encyclopedic knowledge², but potentially less so for dynamically changing properties of the world. In addition, ontology-based generalisation depends on the richness of the ontology model: the more complete the ontology is, the more appropriate the behaviour of a robot using the ontology would be. For a more intelligent generalisation, the knowledge stored in the ontology needs to be augmented with the experiences of a robot itself, as this would reduce the limitations introduced by the incompleteness of the ontology model.

Data-driven models, particularly those based on neural networks, such as [6], [7], have been shown to generalise well to different, even unseen, objects; however, on its own, such generalisation is usually only based on sensory features rather than inherent knowledge about objects and their underlying contexts. While sensory features are sufficient in some applications, the applicability of learning-based methods in real-world, human-centered scenarios is generally limited, as generalisation failures are difficult to analyse without assigning a semantic meaning to the robot’s decisions.

In this paper, we address the problem of generalising action execution policies over different objects, such that we

^{*}This work was supported by the b-it foundation

[†]Alex Mitrevski and Paul G. Plöger are with the Department of Computer Science, Hochschule Bonn-Rhein-Sieg, Sankt Augustin, Germany
<aleksandar.mitrevski, paul.ploeger>@h-brs.de

[‡]Gerhard Lakemeyer is with the Department of Computer Science, RWTH Aachen University, Aachen, Germany
gerhard@informatik.rwth-aachen.de

[§]Corresponding author

¹For example, a robot that has learned how to push cups should be able to deduce that cups and glasses, which are related objects, should be pushed away in a similar way, while cups and screwdrivers, which are unrelated objects, need to be pushed away differently. An ontology can be used to encode and/or infer such knowledge.

²An illustrative example of the extent to which encyclopedic knowledge can be specified is given in [5].

combine a plain object ontology with a probabilistic graph that indicates whether generalisation over objects is possible and likely to lead to successful execution of a particular action, such as grasping. The learned graph is derived from the object ontology, but includes edge weights that indicate the generalisation strength between object classes. We show that such a representation can be useful to determine when already learned models can be reused and when additional learning is needed. We analyse the method with learned execution models of parameterised skills as in [8] using a Toyota HSR [9] manipulating objects in two common domestic scenarios - grasping everyday objects and stowing them in a drawer. The results show that a robot can often reuse knowledge about objects it has manipulated before when dealing with new object classes, but also that feedback about failures can allow it to adapt its generalisation accordingly.³

II. RELATED WORK

Most methods that allow object generalisation are designed for a specific action, such as grasping or pushing, and are built without an ontological layer, such that generalisation is done based on extracted sensory features. Stüber et al. [10] model a pushing action for rigid objects; the model is generalised to previously unseen objects by first using a learned density to sample contact points that most closely resemble the contacts seen in the training data and then predicting the motion of the object using a learned motion model. Liu et al. [11] develop a network-based grasping model which, given a grasp candidate and a context description, calculates the probability that the grasp candidate is suitable for the given context; the proposed architecture combines object features with contextual object and task information, which allows the method to generalise reasonably well to new objects. A similar grasping method is presented by Song et al. [12], where the action is modelled as a hybrid Bayesian network that encodes relationships between the task for which grasping is performed, object features, action parameters, and grasp constraints; by appropriate conditioning, the model can be used for object generalisation, but only to known objects. In principle, our method could be used in conjunction with any of these action-specific models, although we aim to reduce the data requirements of such models by leveraging prior information about objects and their relations.

From a perceptual point of view, knowledge-assisted generalisation over objects and lifelong object learning have been addressed in different contexts. Denninger and Triebel [13] present a lifelong learning-based object classification method which modifies the trees in a random forest as new object classes appear; a random forest is used so that new classes can be incorporated without retraining from scratch. In [14], Young et al. propose a method for inferring categories of unknown objects encountered in everyday scenes; the likely category of an unknown object is inferred using information about the object's context - represented through its surrounding objects - and querying a concept

ontology, which finds the most related concept among a set of candidate concepts. Schoeler and Wörgötter [15] introduce a method for recognising objects from point clouds based on affordances, which are modelled through object parts and part relations; an ontology of tools based on their functions is also defined, which allows recognising the function of unseen tools. As our proposed method depends on a meaningful object grounding, the above methods are complementary to ours: in particular, [13] and [14] are required for incorporating new objects, while the inclusion of affordances as in [15] would enrich the execution models of particular actions, for instance as demonstrated in [16] in the context of tool use.

Conceptually, our work is most closely related to approaches for object and action model learning. Bauer et al. [17] propose an approach that allows generalising actions between different objects, where the objective is to find a posterior distribution of the probability of an action effect (with a particular grounding) given a set of previous experiences of similar actions; here, actions are represented by Probabilistic Action Templates [18], where different effects are associated with probabilities, such that two instantiations are considered similar if (i) the actions use the same template and (ii) if all parameters (symbols) of the actions have a shared parent. Sushkov and Sammut [19] present a Bayesian active learning framework based on which a robot can identify the properties of an object or another system by testing informative actions, observing their outcomes, and updating the belief about its hypotheses of the properties; by repeating the process multiple times, the belief would converge to the correct model hypothesis. Sanan et al. [20] describe a demonstration-based method for learning the model of an object that a robot may need to translate and/or rotate around a given axis; once such a model is learned, it can be used in a parameterised skill for manipulating the object in a particular way. Ivaldi et al. [21] analyse a strategy using which a robot can acquire object models by combining perceptual features with exploratory actions, performed either by the robot itself or by human teachers. As in [17], we essentially generalise actions to objects that share common parents, but we aim for adaptive generalisation, since, as shown in our experiments, having a shared parent is not a sufficient condition for successful generalisation. Similar to [19], our method explores different generalisation hypotheses, such that we aim to either converge to a valid hypothesis or conclude that more knowledge needs to be acquired. As in [20], we learn and then generalise object-specific models, but, unlike [20], we do not constrain our method to regular polyhedra. Finally, our model learning method is somewhat similar to the learning procedure in [21], but our main objective is to additionally learn when the learned models can be transferred between objects.

III. BACKGROUND

A. Action Execution Models

This work builds upon [8], where a representation of action execution models was proposed based on which a robot jointly learns a relational model of success preconditions, or

³Accompanying video: <https://youtu.be/gBiYwjUTWQ8>

a collection of such models for multiple qualitative modes, as well as a continuous model that maps action parameters and execution constraints to predicted execution success. An execution model is learned from labelled execution data, such that the relations used for representing the relational model are defined per action, and the continuous model is represented by a Gaussian process so that prediction uncertainty can be encoded. For execution, action parameters that satisfy the relational model under a given qualitative mode are sampled from the learned success model.

During model acquisition, a robot will typically encounter a few object classes⁴, but may also need to work with other object classes when using a learned model during execution; in this case, the learned model needs to be generalised to classes other than the ones seen during training. In this paper, whenever we refer to generalisation, we mean applying a model M_o , which is known to be applicable for an object class o , to another object class o' .

B. Object Ontology

We use an ontology $\mathcal{O} = (\mathcal{T}, \mathcal{A})$ for generalisation over objects, where \mathcal{T} is the TBox, which encodes object class definitions, relations between those classes, as well as object property definitions, and \mathcal{A} is the ABox, which consists of class and property assertions, namely ground objects that belong to specific object classes and properties of those, respectively. For concreteness, we assume an ontology in the Web Ontology Language (OWL)⁵, although our method is not limited to OWL ontologies since we only leverage the encoded relations between object classes under the assumption that the class hierarchy represents a tree. We only consider generalisation over the TBox, although the presented algorithm can be extended to the ABox as well, but at an expected higher computational cost.

IV. EXECUTION MODEL GENERALISATION

For generalising execution models between objects intelligently, a robot that needs to perform an action with a given object, such as pulling the object to a target region, needs to be aware of whether it has manipulated that particular object or similar objects before. If an explicit execution policy for an object class is not known, the robot has to decide how it could generalise its existing knowledge to the new situation.

In this paper, we propose a method for ontology-assisted execution model generalisation that involves two concepts: (i) based on the ontology, a *subset of related object classes* is identified, namely classes whose execution models (if already known) are useful to consider when executing an action with an object for which there is no known execution model, and (ii) a probabilistic model is created for the related classes, which indicates whether an execution model learned for a

given action and a particular object class is *suitable* when used for executing the same action for another object class.

The objective of the subset of related classes is twofold: first of all, it constrains the search for execution models that may be reused to a computationally manageable level, but more importantly, it prevents generalisation between object classes that are unrelated according to the ontology model, thereby reducing the possibility of execution failures due to inappropriate generalisation.⁶ The probabilistic model also has a dual purpose: it weighs the relations encoded by the ontology in order to reinforce meaningful generalisations and allows experience-based acquisition and improvement based on the resulting outcomes of attempted generalisations.

A. Model Selection for Generalisation

Our proposed method for ontology-based generalisation is based on the idea of what we call a *suitability graph*, which encodes the relatedness and suitability concepts discussed above. A suitability graph is derived from an object cluster, which is defined separately for each object class o .

Definition 1: Given an ontology \mathcal{O} and an object class o , an *object cluster* C_o is the set of ancestor, sibling, and children classes of o in \mathcal{O} , such that there is an execution model $M_{\tilde{o}}$ for every $\tilde{o} \in C_o$.

A cluster C_o represents a set of direct and indirect relations between classes in an ontology, namely an object class o is directly related to its parents and children, and indirectly to its siblings and ancestors. The reasoning behind this definition is as follows: (i) considering ancestor classes allows knowledge transfer from a general case to a specific case, (ii) children classes are exactly the opposite, as knowledge could be transferred from a specific case to a more general case, and (iii) since sibling classes have shared parents, they may also transfer knowledge between each other. Fig. 2 illustrates the notion of an object cluster for kitchen objects.

Given C_o , we know which object classes may be considered for generalisation during execution, but not how appropriate each individual generalisation would be. We represent the probability that the model of an object class \tilde{o} is selected for executing an action a under a qualitative mode q by a weighted probability distribution of the form

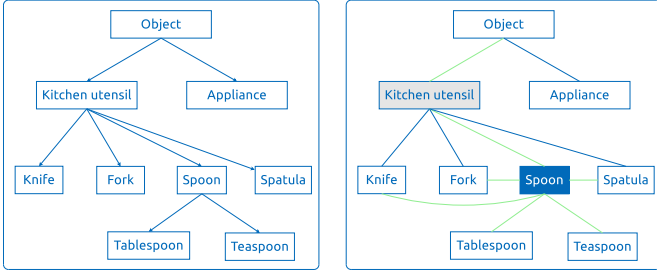
$$P_{t+1}(\tilde{o}|o, S) = \eta s(o, \tilde{o}) P(S|\tilde{o}, o) P_t(\tilde{o}|o, S) \quad (1)$$

where $s(o, \tilde{o})$ represents the similarity between o and \tilde{o} in the ontology, $P(S|\tilde{o}, o)$ is the probability that selecting the model of \tilde{o} leads to successful execution (S is thus a Bernoulli variable), η is a normalising constant, and $P_t(\tilde{o}|o, S)$ is a recursively updated value, such that $P_0(\tilde{o}|o, S)$ is uniform. It should be noted that all probabilities above are additionally conditioned on an action a and a qualitative mode q , but we have omitted these for simplifying the notation. This

⁴For instance, for the experiments in [8], we learned an object pulling model with only one type of object - a yogurt cup. Even such a simple model can be generalised to other object classes since there are other objects that have similar physical properties as a yogurt cup, but a robot does not necessarily know this.

⁵<https://www.w3.org/OWL/>

⁶Our assumption here is that related object classes have similar physical characteristics, which is useful prior knowledge for a robot to have. Our method is still applicable if this does not hold for a given ontology, but a robot may need to perform more generalisation trials in that case.



(a) Example class hierarchy in an ontology (b) Induced object cluster for a given object

Fig. 2: Illustration of an object cluster in an ontology. The blue node is the object class o whose cluster C_o is being extracted. The green edges indicate object classes that could be considered for generalisation if an execution model for o is not known. Gray nodes are classes for which an execution model is not known, and thus do not belong to C_o .

distribution defines a probabilistic model that weighs the relations between the classes in C_o ; we call this weight suitability, and the model itself a suitability graph.

Definition 2: For an object class o and an action a , which is potentially executed under a qualitative mode q , a *suitability graph* $\mathcal{G}_{o,a}$ models a distribution $P(\tilde{o}|o, S, a, q)$ for every $\tilde{o} \in C_o$, which represents the probability that using the execution model of \tilde{o} to execute a with o will result in successful execution.

Suitabilities are composed of two major components - the success probability distribution $P(S|\tilde{o}, o)$ and the similarity coefficient $s(o, \tilde{o})$ - which are described below.

a) Success probability distribution: We model the problem of estimating $P(S|\tilde{o}, o)$ as a sequence of Bernoulli trials with an unknown success probability. Similar to [17], the success probability is found by estimating the parameters of a beta distribution $\text{Beta}(\alpha_{o\tilde{o}}, \beta_{o\tilde{o}})$, whose density is given as

$$p(\theta) = \gamma \theta^{\alpha_{o\tilde{o}}-1} (1-\theta)^{\beta_{o\tilde{o}}-1} \quad (2)$$

where γ is a normalising constant. Given $\alpha_{o\tilde{o}}$ and $\beta_{o\tilde{o}}$, we use $P(S = 1|\tilde{o}, o) \sim \text{Beta}(\alpha_{o\tilde{o}}, \beta_{o\tilde{o}})$ to represent the success probability when applying \tilde{o} 's model to o . To indicate the robot's initial ignorance about the suitability of \tilde{o} 's model, $P(S = 1|\tilde{o}, o) \sim \text{Beta}(\alpha_0, \beta_0)$ before any attempted generalisations, where $\alpha_0 = \beta_0$. The parameters of the beta distribution are subsequently modified according to the results of the action executions. In particular, following [22] and denoting by $N_{o,\tilde{o}}$ the number of times the action has been performed for object class o using the model of \tilde{o} , the posterior parameters of the beta distribution are given as

$$\text{Beta}(\alpha_{o\tilde{o}}, \beta_{o\tilde{o}}) = \text{Beta}(\alpha_0 + N_{o,\tilde{o}}^+ - 1, \beta_0 + N_{o,\tilde{o}}^- - 1) \quad (3)$$

where $N_{o,\tilde{o}}^+ = \sum_i X_{\tilde{o},i}^o$ with $X_{\tilde{o},i}^o = 1$ if the i -th execution was successful and 0 otherwise, and $N_{o,\tilde{o}}^- = N_{o,\tilde{o}} - N_{o,\tilde{o}}^+$. In practice, to estimate $P(S = 1|\tilde{o}, o)$, we draw a collection of

Algorithm 1 Execution Model Selection and Action Execution With the Selected Model

```

1: function GENERALISEEXECUTIONMODEL( $o, X^o, t$ )
2:    $M_o \leftarrow \text{getModel}(o)$ 
3:   if  $M_o \neq \emptyset$  then
4:      $\text{executeAction}(M_o, o)$ 
5:   return
6:    $C_o \leftarrow \text{getObjectCluster}(o)$ 
7:   for  $\tilde{o}$  in  $C_o$  do
8:      $P_{t+1}(\tilde{o}|o, S) = \eta s(o, \tilde{o}) P(S|\tilde{o}, o) P_t(\tilde{o}|o, S)$ 
9:    $o^* \leftarrow \arg \max_{\tilde{o} \in C_o} P_{t+1}(\tilde{o}|o, S = 1)$ 
10:   $M_{o^*} \leftarrow \text{getModel}(o^*)$ 
11:   $\text{outcome} \leftarrow \text{executeAction}(M_{o^*}, o)$ 
12:   $X_{o^*}^o \leftarrow X_{o^*}^o \cup \text{outcome}$ 

```

samples \mathcal{B} from $\text{Beta}(\alpha_{o\tilde{o}}, \beta_{o\tilde{o}})$ rather than a single sample and then set $P(S = 1|\tilde{o}, o) = \frac{\sum_i \mathcal{B}_i}{|\mathcal{B}|}$.⁷

b) Object similarity: To find the similarity between two classes in an ontology, we use the Wu-Palmer (WUP) similarity introduced in [23]:

$$s(o, \tilde{o}) = 2 \frac{\text{depth}(\text{LCS}(o, \tilde{o}))}{\text{depth}(o) + \text{depth}(\tilde{o})} \quad (4)$$

where $\text{depth} : \mathcal{O} \rightarrow \mathbb{N}$ is the depth of an object class in the hierarchy induced by the ontology and $\text{LCS} : \mathcal{O} \times \mathcal{O} \rightarrow \mathcal{O}$ is the least common subsumer of o and \tilde{o} , namely the most specific class that is a common ancestor of o and \tilde{o} .⁸ The WUP similarity corresponds well to \mathcal{G} as it considers (i) siblings to be more similar to each other than nodes among different levels of the hierarchy, which puts an intuitive prior preference on generalisation between sibling classes and (ii) classes with higher depths in the hierarchy to have higher similarity to each other than classes near the top of the hierarchy, which encodes the principle that class similarity increases with the specificity of the ontology model.

c) Model selection: Given the posterior values $P_{t+1}(\tilde{o}|o, S)$ for each $\tilde{o} \in C_o$, an execution model M_{o^*} is selected for the object class o^* that maximises the posterior:

$$o^* = \arg \max_{\tilde{o} \in C_o} P_{t+1}(\tilde{o}|o, S = 1) \quad (5)$$

If there are multiple models that maximise the posterior, all of them are considered to be equally applicable, so one of the models is chosen at random.⁹ The model selection process is summarised in Alg. 1.

Let us now briefly consider how the distribution in Eq. 1 will evolve over time. Given an object class o and before incorporating any evidence about attempted generalisations, classes closer to o in \mathcal{O} will clearly be preferred for generalisation, as the expression is dominated by $s(o, \tilde{o})$; however, as the robot attempts to generalise its knowledge throughout its

⁷We use a mean estimator instead of the expected value $\frac{\alpha}{\alpha+\beta}$ to encourage small random exploration over close posterior values in Eq. 1.

⁸ $s(o, \tilde{o})$ ranges between 0 and 1, with $s(o, \tilde{o}) = 1$ when $o = \tilde{o}$.

⁹If $|C_o| = 1$, the posterior of the only related object will always be 1.

lifetime, its own experiences will start having a more prominent role in the distribution. This will counteract potentially misleading information encoded in the ontology, thereby allowing a robot to exhibit lifelong learning capabilities. Our experiments demonstrate this for some of the objects for which multiple candidate models were available.¹⁰

B. Model Generality and New Model Acquisition

As models are generalised among objects throughout a robot’s lifetime, the robot will acquire knowledge about their applicability for different object classes. This may lead to different outcomes regarding the extent to which a model is applicable: (i) a model can be reliably generalised to all sibling classes, and hence to a parent class, or (ii) existing models are not appropriate for a specific object class, so a robot needs to perform additional learning experiments to acquire a new model for that class. We propose two heuristics to control the generalisation and specification of models.

Generalisation heuristic: A model M_o is general enough to be transferred to a parent class if, for every sibling class \tilde{o} of o , $P(S = 1|\tilde{o}, o) \geq \tau$, where τ is a predefined certainty threshold.

Specification heuristic: A new model M_o for an object class o has to be learned if either $|C_o| = 0$ or, $\forall \tilde{o} \in C_o$, $P(S = 0|\tilde{o}, o) \geq \tau$.

Both of these heuristics have a single hyperparameter: the certainty threshold τ . The definition in Eq. 3 guarantees that the belief of $S = 1$ and $S = 0$ will increase smoothly with the number of execution successes and failures respectively, such that a reasonably high τ will prevent premature decisions about the (un)suitability of a model. For instance, using $\alpha_0 = \beta_0 = 1$ and $\tau = 0.8$ means that a model M_o can be generalised to a parent class if at least three successful and no failed executions have been observed for all sibling classes, but at least eight successful executions will be required if at least one failed execution is observed. This behaviour is rather intuitive and desirable: the more inconclusive the generalisation results are, the more investigation a robot needs to do before drawing conclusions about the need for generalising or specifying a model.

V. EXPERIMENTS

We evaluate our method for ontology-assisted generalisation by considering various domestic objects in the context of two actions performed by a Toyota HSR: grasping an object for subsequent transportation and stowing a grasped object in a drawer. The experimental setup is illustrated in Fig. 3.

For the experiments, we use a subset of object classes from the YCB object set [24], shown in Fig. 4, in particular fruits

¹⁰One thing to note is that $s(o, \tilde{o})$ could, in principle, be incorporated in Eq. 1 through the initial parameters of the beta distribution, but keeping it as an outside multiplicative factor makes it easier to incorporate changes in the ontology hierarchy, as those would only be reflected in the value of $s(o, \tilde{o})$ without disrupting previously collected generalisation experiences.



(a) Grasping an object

(b) Object stowing in a drawer

Fig. 3: Illustration of the setup for the experimental use cases



Fig. 4: Objects used in the experiments

(banana, apple, orange, strawberry), food and drink containers (chips can, tomato can, cracker box, sugar box, mustard container, mug, wine glass, pitcher), and balls (tennis ball, baseball, racquetball).¹¹ We use an OWL ontology similar to KnowRob [1] for organising the object classes, such that most YCB classes and other common domestic objects are included in the ontology¹²; we do not directly use KnowRob because some of the YCB classes are not included there.

For both actions, we learn dedicated execution models (using guided learning as in [8]) for each of the following objects in order to have a reasonable variety of models: apple, chips can, sugar box, mug, and tennis ball. Each model is learned using 25 executions of the action; the remaining objects are used for testing the generalisation. For testing, we perform 10 trials for each action and test object, such that $P(S|\tilde{o}, o)$ is updated after every execution based on the outcome and $P_t(\tilde{o}|o, S)$ is updated according to Eq. 1.^{13,14}

A. Object Grasping

In the grasping experiment, the robot is placed in front of a table and, for every trial, a single object is placed on the table, which the robot needs to find and grasp. A Faster

¹¹As we do not have access to the actual YCB objects, we use local objects that closely resemble the YCB objects.

¹²The ontology used in the experiments can be found at https://github.com/b-it-bots/mas_knowledge_base/blob/devel/common/ontology/apartment.owl

¹³The data from our experiments are available at <https://zenodo.org/record/4551725>

¹⁴Implementation of Alg. 1 available at <https://github.com/alex-mitrevski/explainable-robot-execution-models>

R-CNN model [25] trained on the YCB subset is used for object detection and recognition.^{15,16} The grasping action is parameterised by the grasping pose, which is represented by (i) the relative gripper position with respect to the center of the object’s bounding box and (ii) the relative wrist orientation with respect to an estimated object orientation (in the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$). The robot executes a grasp at the selected pose and the outcome is evaluated by a teacher.¹⁷ To allow generalisation to differently sized objects, the position parameters are normalised to the range $[-1, 1]$ for all axes in the learned model (motivated by [26]).

For simplicity of the experimental setup, the robot attempts a sideways grasping strategy for all objects; this also illustrates the limited generalisability between objects more clearly. To estimate the object’s position and size before grasping, we find the largest point cloud cluster within the 2D bounding box of the detected object; the center of the cluster is then taken to be the object’s position, while the span of the points represents the object’s size. To find the planar orientation of the object with respect to the robot, we use a RANSAC-like procedure [27] that finds the direction of 2D lines fitted to randomly selected points from the object’s point cloud; the average of these line orientations is taken to represent the object’s orientation.¹⁸

The relational model of the grasping action is extracted from the following relations (similar to [8]):

$$\begin{aligned} \text{in_front_of}_{x,y}(p, B) &:= p_{x,y} < \min(B_{x,y}) \\ \text{behind}_{x,y}(p, B) &:= p_{x,y} > \max(B_{x,y}) \\ \text{above}(p, B) &:= p_z > \max(B_z) \\ \text{below}(p, B) &:= p_z < \min(B_z) \\ \text{centered}_{x,y,z}(p, B) &:= |p_{x,y,z} - \overline{B_{x,y,z}}| \leq \epsilon \\ \text{perpendicular_to}(\theta_o, \theta_p) &:= |\theta_o - \theta_p| \approx \frac{\pi}{2} \\ \text{parallel_to}(\theta_o, \theta_p) &:= |\theta_o - \theta_p| < \theta \end{aligned}$$

Here, $\text{rel}_{a_1, \dots, a_n}$ means that the relation rel is defined for each of the axes a_1, \dots, a_n , p is the grasping pose, B is the object’s bounding box, \min/\max are the minimum and maximum coordinates along a given axis, θ_o and θ_p are the object’s planar orientation and the wrist orientation, respectively, ϵ is a distance threshold set to 5cm , and θ is an orientation threshold set to 25° , which is a conservative

¹⁵Available at https://github.com/b-it-bots/mas_models/tree/master/perception_models/detectors/ycb

¹⁶We used real fruits in the experiments, except for a plastic strawberry; this was due to the fact that real strawberries seemed to be too small for the detector. The model is also unable to recognise the wine glass and the racquetball as such, but recognises them as a chips can and apple respectively; in the generalisation trials with these two objects, we treated them as if they were recognised correctly by the model since this does not affect the results of the experiments.

¹⁷A grasp is considered successful if the object remains in the gripper when the robot retracts the arm back after grasping.

¹⁸We estimate an object’s orientation using subsets of the object’s point cloud instead of the full point cloud for computational reasons: particularly for larger objects, processing the full point cloud at once takes several seconds, which is not very practical.

value to allow for noisy angle estimates.

B. Object Stowing

For the stowing experiment, the robot is positioned in front of an open drawer and an object is placed in the robot’s gripper; the robot needs to throw the object in the drawer in an orderly fashion, namely without damaging the object and so that the drawer can be closed afterwards. As in the grasping case, throws are evaluated by a teacher.¹⁹ For repeatability, the object is held by the robot in a similar orientation for all trials; this corresponds to the orientation in which the object is held in the successful grasps of the grasping experiment. Before throwing, the drawer is identified by handle detection (using the same detection model as in [8]); the drawer itself is represented as a box with a known size. The action is parameterised by the throwing pose, which is represented by (i) the relative gripper position with respect to the bounding box of the drawer²⁰ and (ii) the absolute wrist orientation when throwing the object. The position relations used for the grasping model are also used for the relational model of the stowing action; the orientation predicates are used as well, but they evaluate the absolute gripper orientation at which the object is thrown.

C. Results

The number of successful learning trials for each action and training object is shown in Table I.

TABLE I: Successful executions in the data used for learning object-specific execution models (out of 25)

Object	Grasps	Stows
Apple	9	11
Chips can	11	8
Mug	9	4
Sugar box	15	7
Tennis ball	17	15

As could be expected, several of the learning trials failed due to the fact that execution parameters were selected randomly (position parameters within the object and drawer bounding boxes respectively, throwing height up to around 30cm from the top of the drawer, and arbitrary grasping and throwing orientations). Most grasp failures were caused by objects slipping out of the gripper or by the robot attempting to grasp a bit too far from the object. Failed throws were caused either by the object being thrown near the edges of the drawer or by an unreasonably large throwing height.

The results of the generalisation experiments for the two actions are shown in Table II. We report (i) the size of the object cluster C_o (as we only learned models for five objects,

¹⁹Throws from large heights are evaluated as unsuccessful for some of the objects, such as the mug, even if the object ends up in the drawer. Similarly, throws that prevent the drawer from closing afterwards or in which the object falls into the drawer after a lucky bounce off the edges are evaluated as unsuccessful.

²⁰To account for the fact that some part of an object may be extended below the gripper, the average extended length in the successful grasp trials is added to the relative height.

$|C_o|$ varies between 1 and 2 for the test objects²¹), (ii) the number of attempted models for generalisation (in case of multiple available models), (iii) the object class o^* whose model generalises best to each object class o if $\exists o^* \in C_o$ for which $P(S = 1|o^*, o) \geq \tau$, and (iv) the total number of successful executions N^+ over the test trials. During the experiments, we used $\alpha_0 = \beta_0 = 3$ as prior values for the success distribution; we use $\tau = 0.6$ to decide if any of the existing models can be generalised to a test object.²²

a) *Grasping*: As the results of the grasping experiment show, the learned models can be reliably generalised to some of the test objects, but the acquired knowledge is not sufficient in general. As expected, the model of the tennis ball can be generalised to both other balls, but the racquetball was pushed away in a few of the attempts as it is considerably lighter. Similarly, the apple grasping model can be generalised to the orange, although the orange slipped in a few attempts, and surprisingly to the strawberry. As expected, the model cannot be generalised to the banana, for which a top-down grasping strategy would be more suitable (as in Fig. 1); some of the banana grasping attempts succeeded as well, but the banana was deformed in all cases due to an inappropriate grasp approach direction.

The results are more interesting for the containers. Neither of the sugar box and chips can models can be generalised to the cracker box, pitcher, and mustard container. In the case of the cracker box, the sugar box model was successful in half of the attempts, but does not lead to reliable generalisation as its mostly symmetric shape does not provide enough coverage for constraining the grasping orientation, which is important for the elongated cracker box. The mustard container and the pitcher could also not be grasped using the known models due to being quite slippery and also elongated. The tomato can is grasped reliably, but, surprisingly, using the model of the sugar box and not that of the chips can. This is because both the sugar box and the chips can are equally similar to the tomato can based on the ontology and the sugar box model was initially chosen by chance, such that there was no need to attempt the chips can model due to the mostly successful executions using the sugar box model.

b) *Stowing*: According to the results of the stowing experiment, stowing is considerably simpler than grasping, as the learned models can be generalised to most of the test objects. The model of the tennis ball is reliably generalised to the other balls, except for a single failure with the racquetball, which bounced out of the drawer. The model of the apple can generally be reused for the banana and orange; the model can also be reused for the strawberry, but not as successfully, as the strawberry is sometimes thrown from a large height that would damage the fruit.

As in the grasping case, the container results are more

interesting to consider. The sugar box model is reused for both the cracker box and the pitcher, but the execution failed a few times due to the objects being thrown upright, which means that the drawer could not be closed afterwards, or too close to the drawer edges and bouncing out as a result. The sugar box model is also reused for the tomato can; as in the grasping case, the sugar box was chosen by chance initially, such that the box model is quite reliable for the tomato can since the objects have similar masses. The model of the chips can was only reused for the mustard container, initially by chance and then successfully, with only a single failure due to the object being thrown from a large height. Finally, as could be expected, the wine glass requires delicate treatment, so the model of the mug cannot be generalised to it.²³

VI. DISCUSSION AND CONCLUSIONS

Our method, represented by the notion of what we refer to as a suitability graph, allows a robot to use relations between objects encoded in an ontology as well as its own experiences in order to generalise its action execution knowledge to object classes that have not been manipulated before. In particular, the ontology serves as a prior that guides the generalisation between classes when the robot does not have sufficient experiential information, but the robot's behaviour will be dominated by its own experiences as it interacts with objects throughout its lifetime. As the experiences are treated as annotations over the ontology, the result is a generalisation strategy suitable for explainability and failure analysis. Additionally, due to the probabilistic nature of the model, a robot can determine that its existing knowledge is insufficient for execution in a given context, which enables a form of lifelong acquisition of execution knowledge.

Our method is based on the assumption that new object classes which have to be manipulated are already present in the robot's ontology, which means that generalisation to objects that are not encoded in the ontology cannot be done directly; for that, it would be necessary to grow the ontology as completely new objects enter the domain, using methods such as [14], [28]. Related to this and an aspect that we did not address in this paper is that of how to adjust the learned suitabilities if the object cluster is expanded with a new model for a related object class, as this means that there is no generalisation information about that class; future work should investigate strategies to include such models without disrupting the previously learned suitabilities. Another important assumption, which is reflected in the definition of an object cluster, is that only classes that belong to the object's family are included in the cluster, although it might sometimes also be desirable to allow a robot to discover object relations that are not directly encoded in the ontology; one possible strategy for this would be to expand the object cluster by utilising information about object affordances [2], [15], [29] or object materials [11]. The generalisation between objects obtained using our method is also defined per action; however, similar objects will generally be treated

²¹If C_o was not used, all training objects could be considered for generalisation; this increases the number of trials required for identifying an appropriate model and does not scale with the number of objects.

²² $\alpha_0 = \beta_0 = 3$ and $\tau = 0.6$ means that at least seven successes in 10 executions are needed to conclude that a model can be generalised to another object class.

²³In fact, a few wine glasses were broken during the experiment.

TABLE II: Generalisation results for the test objects after 10 executions. We include the size of C_o , the number of attempted models during generalisation, the object class o^* whose model is selected for generalisation (or / in case $P(S = 1|\tilde{o}, o) < \tau$ $\forall \tilde{o} \in C_o$), and the total number of successful executions N^+ over the 10 executions.

Action	Object	Banana	Orange	Strawberry	Cracker box	Can	Container	Pitcher	Wine glass	Baseball	Racquetball
	$ C_o $	1	1	1	2	2	2	2	1	1	1
Grasp	#models	1	1	1	1	1	2	2	1	1	1
	o^*	/	apple	apple	/	sugar box	/	/	mug	tennis ball	tennis ball
	N^+	4	7	9	5	8	2	1	8	8	7
Stow	#models	1	1	1	2	1	1	1	1	1	1
	o^*	apple	apple	apple	sugar box	sugar box	chips can	sugar box	/	tennis ball	tennis ball
	N^+	9	10	7	7	9	9	8	1	10	9

similarly across actions, so already learned relations could potentially be transferred between actions as well, which is an aspect that needs to be investigated in future work. Even though our method is not limited to a specific action, it would be desirable to perform a direct comparison with other existing methods, for instance [6] for grasping. Finally, in this paper, we used the representation in [8] to model actions; however, it should be noted that the suitability graph is not limited to a specific action representation, which allows different policy models to be combined if desired.

ACKNOWLEDGMENT

We would like to thank Ahmed Abdelrahman and Santosh Thoduka for their comments on an earlier draft of this paper.

REFERENCES

- [1] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A. K. Bozcuoğlu, and G. Bartels, “Know Rob 2.0 - A 2nd Generation Knowledge Processing Framework for Cognition-Enabled Robotic Agents,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 512–519.
- [2] I. Awaad, G. K. Kraetzschmar, and J. Hertzberg, “Finding Ways to Get the Job Done: An Affordance-Based Approach,” in *Proc. 24th Int. Conf. Planning and Scheduling (ICAPS)*, Robotics Track, 2014.
- [3] A. Olivares-Alarcos *et al.*, “A review and comparison of ontology-based approaches to robot autonomy,” *The Knowledge Engineering Review*, vol. 34, p. e29, 2019.
- [4] D. Paulius and Y. Sun, “A Survey of Knowledge Representation in Service Robotics,” *Robotics and Autonomous Systems*, vol. 118, pp. 13–30, 2019.
- [5] S. Schneider, N. Hochgeschwender, and G. K. Kraetzschmar, “Declarative Specification of Task-based Grasping with Constraint Validation,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2014, pp. 919–926.
- [6] J. Mahler *et al.*, “Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics,” 2017.
- [7] L. Shao, T. Migimatsu, and J. Bohg, “Learning to Scaffold the Development of Robotic Manipulation Skills,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020, pp. 5671–5677.
- [8] A. Mitrevski, P. G. Plöger, and G. Lakemeyer, “Representation and Experience-Based Learning of Explainable Models for Robot Action Execution,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, Oct. 2020, pp. 5641–5647.
- [9] 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*, vol. 6, pp. 1–15, 2019.
- [10] J. Stüber, M. Kopicki, and C. Zito, “Feature-Based Transfer Learning for Robotic Push Manipulation,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 5643–5650.
- [11] W. Liu, A. Daruna, and S. Chernova, “CAGE: Context-Aware Grasping Engine,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020, pp. 2550–2556.
- [12] D. Song, K. Huebner, V. Kyrki, and D. Kragic, “Learning task constraints for robot grasping using graphical models,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2010, pp. 1579–1585.
- [13] M. Denninger and R. Triebel, “Persistent Anytime Learning of Objects from Unseen Classes,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018, pp. 4075–4082.
- [14] J. Young, V. Basile, L. Kunze, E. Cabrio, and N. Hawes, “Towards Lifelong Object Learning by Integrating Situated Robot Perception and Semantic Web Mining,” in *Proc. 22nd European Conf. Artificial Intelligence*, 2016, pp. 1458–1466.
- [15] M. Schoeler and F. Wörgötter, “Bootstrapping the Semantics of Tools: Affordance Analysis of Real World Objects on a Per-part Basis,” *IEEE Trans. Cognitive and Developmental Systems*, vol. 8, no. 2, pp. 84–98, June 2016.
- [16] P. Gajewski *et al.*, “Adapting Everyday Manipulation Skills to Varied Scenarios,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019, pp. 1345–1351.
- [17] A. S. Bauer, P. Schmaus, F. Stulp, and D. Leidner, “Probabilistic Effect Prediction through Semantic Augmentation and Physical Simulation,” in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020, pp. 9278–9284.
- [18] D. Leidner, C. Borst, and G. Hirzinger, “Things are made for what they are: Solving manipulation tasks by using functional object classes,” in *12th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids 2012)*, 2012, pp. 429–435.
- [19] O. O. Sushkov and C. Sammut, “Active robot learning of object properties,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2012, pp. 2621–2628.
- [20] S. Sanan, M. Bretan, and L. Heck, “Learning Object Models For Non-prehensile Manipulation,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2019, pp. 4784–4789.
- [21] S. Ivaldi *et al.*, “Object Learning Through Active Exploration,” *IEEE Trans. Autonomous Mental Development*, vol. 6, no. 1, pp. 56–72, 2014.
- [22] D. Koller and N. Friedman, “Parameter estimation,” in *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009, ch. 17, pp. 733–739.
- [23] Z. Wu and M. Palmer, “Verbs Semantics and Lexical Selection,” in *Proc. 32nd Annual Meeting on Association for Computational Linguistics*, 1994, pp. 133–138.
- [24] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, “The YCB object and Model set: Towards common benchmarks for manipulation research,” in *Int. Conf. Advanced Robotics (ICAR)*, 2015, pp. 510–517.
- [25] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” in *Advances in Neural Information Processing Systems* 28, 2015, pp. 91–99.
- [26] S. Brandl, O. Kroemer, and J. Peters, “Generalizing Pouring Actions Between Objects using Warped Parameters,” in *14th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, Nov. 2014, pp. 616–621.
- [27] M. A. Fischler and R. C. Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, June 1981.
- [28] G. H. Lim *et al.*, “Interactive Teaching and Experience Extraction for Learning about Objects and Robot Activities,” in *23rd IEEE Int. Symp. Robot and Human Interactive Communication*, 2014, pp. 153–160.
- [29] C. Wang, K. V. Hindriks, and R. Babuska, “Effective Transfer Learning of Affordances for Household Robots,” in *4th Int. Conf. Development and Learning and Epigenetic Robotics*, 2014, pp. 469–475.