Analogical Generalization of Actions from Single Exemplars in a Robotic Architecture

Jason R. Wilson
wilson@cs.tufts.edu

Evan Krause
evan.krause@tufts.edu

Matthias Scheutz
matthias.scheutz@tufts.edu

Morgan Rivers
daniel.rivers@tufts.edu
Human-Robot Interaction Lab
Tufts University
200 Boston Ave
Medford, MA

ABSTRACT

Humans are often able to generalize knowledge learned from a single exemplar. In this paper, we present a novel integration of mental simulation and analogical generalization algorithms into a cognitive robotic architecture that enables a similarly rudimentary generalization capability in robots. Specifically, we show how a robot can generate variations of a given scenario and then use the results of those new scenarios run in a physics simulator to generate generalized action scripts using analogical mappings. The generalized action scripts then allow the robot to perform the originally learned activity in a wider range of scenarios with different types of objects without the need for additional exploration or practice. In a proof-of-concept demonstration we show how the robot can generalize from a previously learned pick-and-place action performed with a single arm on an object with a handle to a pick-and-place action of a cylindrical object with no handle with two arms.

Keywords

analogical generalization, action learning, robotic architecture, mental simulation

1. INTRODUCTION

Humans are known to employ many different methods for learning new information, from data-driven statistical learning (e.g., when infants discover and entrain the motor behaviors) to knowledge-based conceptual learning (e.g., based on "insight" or "understanding"). One striking ability of adult human learners is that they can bring their learned knowledge at all levels of abstraction to bear when faced with new learning problems and can thus generalize new information very quickly from only a few exemplars. For example, human adult learners, after observing another human demonstrating a novel action on a novel object, are typically able to describe the new action and potentially perform it right away; and most importantly, they often can immediately determine whether it might apply to similar objects and how they would have to modify it to perform it on those objects. In a sense, *humans can learn generalized knowledge from single exemplars* and the main goal of this paper is to demonstrate for the first time a similar, albeit much more restricted capability (compared to the human case) for generalization from a single demonstration in a cognitive robotic architecture. This functionality is achieved by integrating algorithms for analogical reasoning and generalization as well as algorithms for generating counter-factual scenarios from a given scenario together with a physics simulation into a cognitive robotic architecture.

The paper proceeds as follows. We start with a motivating scenario which we will use throughout the paper to describe the functionality of our system. Next, we introduce both existing and new algorithms for mental simulation and generalization, and then introduce the relevant parts of the robotic architecture and a proof-of-concept demonstration of the integrated systems on a physical PR2 robot. We then evaluate the generalization process, review related work, and conclude with a discussion of our work and future directions enabled by the integrated architecture and the generalization capabilities.

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

2. MOTIVATION

Consider a scenario where a robot is tasked with picking up a medical kit from a table. Initially, the robot has no concept of the steps necessary to perform the task, which is then demonstrated by a human who picks up the medical kit by its handle and moves it aside. Suppose the robot is then able to construct a step-wise representation that enables it to imitate the human and pick up the same medical kit by the handle and move it to the side. Figure 1 (left) shows the robot demonstrating its ability to use the action representation to mimic the demonstrated action.

This newly learned action, however, is very specific to the human demonstration and does not automatically generalize to other scenarios (e.g., different object orientation/location on the table, different types of objects with and without handles, etc.). A human learner in this case would, for example, notice that many details about the object (e.g., the color, the texture, the size within some ranges, the orienta-

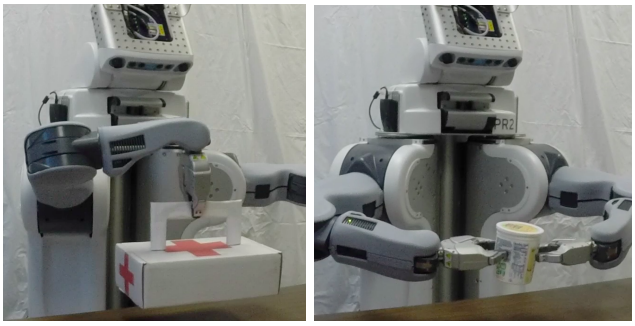

Figure 1: The PR2 performing the action learned from a single exemplar (left) and its generalization to a new object and action (right).

tion) do not matter for the particular pick-and-place action as long as it has a handle; and even if it did not have a handle, there might be different ways to pick it up (e.g., using two hands). Alternatively, humans can adapt the sequence of actions (e.g. add, remove, or repeat some) in a manner that still accomplishes the goal. But how do humans achieve this feat? One suggestion is that humans perform quick mental simulations, possibly physics-based, that allow them to imagine different scenarios and try out the learned activity "in their mind" [1]. Based on these "mental experiments", they are then able to generalize to the successful cases and exclude the unsuccessful ones. The successful or unsuccessful outcome of a mental simulation is, of course, *no guarantee* that an action will or will not work. Clearly, the accuracy of the simulation together with the appropriateness of the assumptions about objects and their properties will be a major determinant of how well mental simulation can predict actual outcomes. However, even when the accuracy of the simulation is not very high, the simulations can at the very least be used as a guideline for whether the simulated action should be considered as a valid alternative to be incorporated into a generalization of the action. When a new scenario is presented and the robot has not learned an appropriate action for it, the generalized action obtained through a series of mental simulations is a promising choice that can later be refined through practice.

3. FROM LEARNED ACTIONS TO GENERALIZATIONS AND BACK

Given a robotic control architecture that comprises the functionality (implemented in various components, say) to learn and execute an action script (such as the one for picking up a medical kit), the question we address in this paper then is how such an architecture can be augmented to allow for action generalization, i.e., what architectural extensions and algorithm integrations are needed for the robot to be able to generalize and immediately execute the previously learned actions in different scenarios. Note that there are several important gaps between the given and the desired architectural capabilities that such an extension has to bridge. These gaps are intrinsically connected to the tension between the concrete and the abstract as well as the tension between the physical world and the mental simulation.

Specific vs general:.

On the one hand, we have very specific knowledge about a single (learned) exemplar, yet we want to generate more abstract knowledge about a class of exemplars. Statistical approaches bridging this gap by learning from numerous specific exemplars are not applicable, hence we attempt to bridge this gap with a novel combination of simulation and analogical reasoning.

Continuous vs discrete:.

Some of the involved sensory data is typically continuous (the robot's vision system might create a point cloud representation of the perceived object, where each point is along three dimensions in continuous space). Similarly, the robot's arm motion is described by 3-D vectors along a continuous trajectory, while classes of trajectories in an action representation might be denoted by single action modifier (e.g., "up" as in "lift up"). One way to bridge this gap is to add explicit links between these continuous and discrete representations (e.g., in the form of a predicate that relates "up" to the vector for moving up).

Exemplars vs generalization:.

Every exemplar has associated with it far more information than is encoded in the generalization (this concerns all aspects of the exemplar including the involved objects and actions as well as other details of the scene). The process of creating the generalization naturally eliminates irrelevant information, but what information is irrelevant? The generalization must maintain enough information such that it is sufficient for performing the generalized action for an interesting class of exemplars. The mental simulation in conjunction with systematic variations of the scenario can partly address this problem by allowing the robot to discover relevant aspects (e.g., if a particular variation led to action failure that that variation was relevant for performing the action).

Simulated vs physical world:.

There are three primary gaps between these worlds: *physics*, *objects*, and *perception*. The obvious gap in the physics is that the simulation is just an approximation of the actual physics of the real world. This is best exemplified in collisions. In the simulation world, collisions can optionally be ignored and in some cases need to be ignored (e.g., at the moment when the gripper "becomes one" with the object). A significant difference in the representation of objects is that in the simulation world we can assume complete knowledge of the object. This means we know its weight, color, size, shape, location, etc. In the physical world, we cannot easily know some of this information (such as the weight) and other information is based on inferences by the vision system. An example is that in the simulation world we have a full 3-D model of the object, but in the physical world we can only get a 2.5-D model of the object. Lastly, the assumption of complete knowledge in the simulation world allows us to avoid problems with inaccurate or wrong perceptions, which have to be addressed in the real world. However, note that action selection, execution, and learning must operate in simulation in nearly the exact same way as in the real world.

To bridge these various gaps, we propose the process depicted in Figure 2: given an initial action representation, (e.g., via learning from demonstration or some other method), the robot can use the representation to either execute the

action or construct a generalization of the action. The generalization process performs mental simulations in counter-factual scenarios that differ from the given one in some respects. The robot must store enough information about the context in which it learned the action to be able to generate counter-factual scenarios. The resulting generalized action representation is applicable to both scenarios it considered during mental simulation and others it did not consider. Note that the robot can either generalize the learned action right away (after having learned it), or it can perform the generalization at a later point.

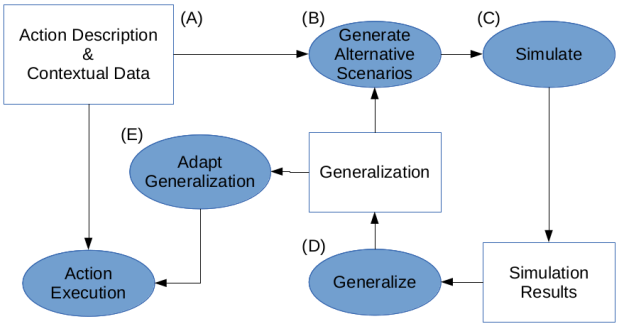

Figure 2: The action generalization process. The letters correspond to the subsections below and depict the typical data flow.

3.1 Action and Scene Representation (A)

The input to this process is a representation of an action and the context in which the action can be applied. This information may have come from a human demonstrating the action, or it may have been provided to the robot in some other form, such as instruction. A description of the action is not sufficient – the context of the action, including all relevant objects and agents involved and relations between these entities, is essential and must be included.

The action representation gives the robot a basic understanding of how to perform the desired action by providing a set of step by step action sequences. In the case of the pick-and-place action representation for picking up a medkit, three simplified action scripts for picking up an object, moving it aside, and setting it back down can be seen in Figure 3. The top-level action is composed of two sub-actions – a pick action and a place action (both action scripts themselves), and the pick sub-action is composed of the three stages: moving to the object, grasping the object, and lifting it vertically relative to its original position. The place sub-action is broken down into a lateral and downward movement, and finally the object is released. Here, the lowest level actions (i.e., move-to-object, grasp-object, move-to-relative, release-object) are action primitives that can be executed by a component in the architecture.

Each action also defines a set of parameters and a set of pre- and post-conditions. The post-conditions serve two important roles. Given an arbitrary goal, the action that has the goal as a post-condition may be executed to accomplish the goal. This is commonly seen in planning systems. Additionally, the post-conditions define the means by which we

```
pick-and-place ?robot ?object ?arm ?dest
  pick ?robot ?object ?arm
  place ?robot ?object ?arm ?dest

pick ?robot ?object ?arm
  move-to-object ?robot ?arm ?object !grasp
  grasp-object ?robot ?arm ?closepos ?object
  move-to-rel ?robot ?arm !up-pt ?orient

place ?robot ?object ?arm ?dest
  move-to-rel ?robot ?arm !over-pt ?orient
  move-to-rel ?robot ?arm !down-pt ?orient
  release-object ?robot ?arm ?object
```

Figure 3: Simplified action scripts for picking up an object, moving it aside, and setting it back down.

measure the success of the goal. For example, the pick-and-place action has the post-condition `at(?object,?dest)`, meaning that the object is at the destination location. The action is intended to move the object to this location, but if the action is not successful (e.g., because it is not the right action or used with the wrong parameters) then the object will not have been displaced.

In addition to the action representation, the scene has additional contextual information that must be represented in order to instantiate meaningful counter-factual simulation scenarios. Here it must be determined what information is relevant, what should be stored (e.g., to allow for simulations to be constructed in the future), and what should be varied during simulation. Scene representations include basic information about objects properties and the relation between the objects. Properties of the object include basic physical properties (e.g. mass, color, etc.), the shape and structure (e.g. cylindrical with a vertical handle on the side), and the location of the object. An object may be related to other objects spatially, compositionally, or otherwise (e.g. the cup is on the table, or the mass of the box is greater than that of the cup). The scene must also have some representation of the agent performing the action and any other relevant agents. For the example of picking up a medical kit by its handle, it is important that the agent have a mechanism by which it can do this. Thus the agent description must include a physical description of the agent and its capabilities (e.g. the robot has a gripper, gripper can grasp object). The scene representation is not limited to a physical description of the scene. An action may be relevant to other contextual information, such as social roles (e.g. the agent giving the instruction is a manager) or cultural norms (e.g. shaking hands when greeting).

3.2 Scenario Generation (B)

The robot knows (because it was been told, it has been demonstrated, or it has been instructed) that a given action is applicable to a specific scenario. This is just one scenario in which the action may be applicable, but it must consider alternative scenarios that may be equally applicable. The robot conducts a scenario generation process to produce counter-factual scenarios that will be subject to simulation and incorporated into the generalization.

Each variant of the exemplar produced during scenario generation is intended to give the robot multiple virtual ex-

periences so that it can form an understanding of the effects of variations on the provided goal. Each counter-factual scenario is constructed by varying one or more variables in the exemplar. Variables can describe either the robot's actions (i.e., action variables as defined above) or scene variables that describe the world with which the robot interacts.

Each scenario variable is carefully defined to allow automated generation of variants and contains five important properties: (1) whether it is an action variable or scene variable, (2) type (continuous or discrete values), (3) its relation to other variables (i.e., valid states), (4) the degree of effect on the success of the simulation, and (5) state history from previous scenarios, to reduce duplication of generated variants on the demonstration. Some of the variables that may be varied and the set of possible values are shown in Table 1.

Table 1: Some variables that may be varied during scenario generation pertain to the action, but many describe the object. Variables have either a set of discrete values or a range of values. Color is unique in that it has 3 range values.

Name	Action?	Values
base shape	N	sphere, cylinder, box
height	N	[0.01, .4]
radius	N	[0.035, .15]
width	N	[0.07, .4, 1]
length	N	[0.07, .4, 1]
has handle	N	handled, handleless
handle type	N	loop, bar, knob
handle location	N	top, left, right, front
handle	N	vertical, horizontal
mass	N	[0.01, .1]
color	N	[0, 1],[0,1],[0,1]
y offset	N	[-.2, .2]
grasp type	Y	close, apart, push, two arm
arm	Y	right, left, both
gripper % open	Y	[.01, .1]
gripper % closed	Y	[.01, .1]

When generating variables that describe the scene, we take advantage of the abilities that generalization by analogical reasoning provides. One such ability is to be able to analyze a scenario with changes in multiple variables at a time and still perform effective analogical analysis between the scenarios. Thus, each counter-factual scenario generated may contain multiple variables that have been altered. The scenario generation algorithm performs (1) a systematic variation of the action variable that affects the greatest number of actions and (2) two random variations on less important variables defining the object to be interacted with. This three-variable variation for each scene allows a significant reduction in the number of scenes required for meaningful learning to occur while still maintaining the accuracy of produced analogies.

The way we generate the values of variables defining our scenario variations involves important assumptions the robot has about the world it must interact with. For example, ubiquitous physical properties such as friction are given default values unless otherwise provided by the demonstration. We also assume that the objects the robot performs actions on can be adequately represented by a combination of a simple base shape (sphere, cylinder, or box) and a handle (knob, loop, or bar) centered on one of its visible faces.

Another relevant assumption about the generated objects is that object interaction is limited to a set of allowable points on the object which the robot can grasp. The effective production of these points has been demonstrated in other research (e.g., [20, 4]), and we did not attempt to re-implement them here. As a result, the object is given a set of predefined grasps based on the object's geometry and handle number.

3.3 Mental Simulation (C)

The mental simulation stage begins by instantiating the scene in the simulation world. The robot is initialized to a specified state, while the rest of the scene is initialized from the scene variables defined in the scenario generation stage. These robot and scene specifications are used to produce a realistic representation of each scenario in the physics simulator. For our scenario, each scene consists of a robot, a table, and single object located on the table-top. Once the simulation scene has been initialized, the robot attempts to execute the action script using the action variables defined in the scenario generation stage.

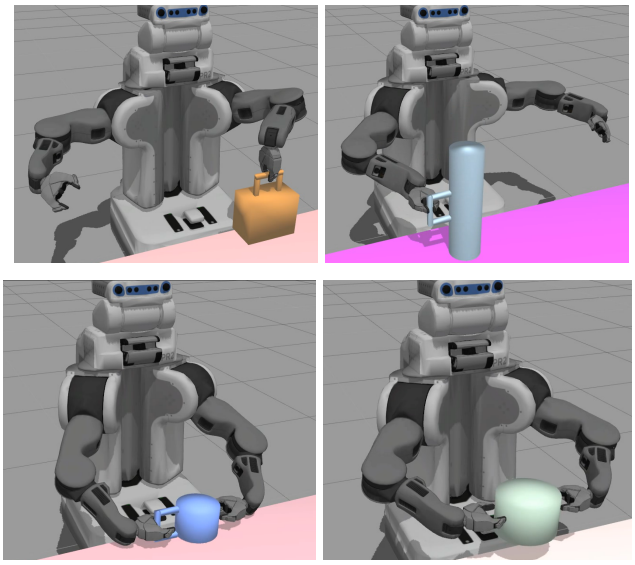

Figure 4: Four examples of the robot simulating the pick-and-place action on different objects.

While the robot performs these actions in the simulated environment (see Figure 4 for examples), the success of the actions it attempts is monitored, and a simulation success rating is recorded for each simulation run. During a simulation run, the inability to perform a required primitive action is considered a failure, ending the simulation of that scenario, resulting in a low score. Otherwise, the robot will execute the entire action script, reevaluating the overall success after each action is completed. The degree to which the action is successful is based on the post- conditions of the action. If the action is to pick up the object but the object is still on the table, then the action was not successful and the simulation success rating would be low. Some actions may not fully succeed or fully fail. For these there needs to be a measure of success. For example, if the goal of the action is to move the object one meter to the right, but the

action only moved it half way, then the action is given a score representing the partial success.

3.4 Generalization by Analogy (D)

The Generalization by Analogy produces a set of generalizations of the exemplar action where each generalization is a structurally different abstraction of the action that still accomplishes the same goal. The generalizations are constructed by analogically comparing each the of counter- factual scenarios to the original exemplar. If the scenario is sufficiently similar, it is assimilated into the generalization. If it is not, then a new generalization is constructed with the scenario as the base exemplar. The algorithm described below has the following enhancements from the one proposed in [21]: (1) incorporating of simulation results and (2) constructing multiple generalizations.

Evidence from psychology (e.g. [8], [9], [14]) suggests that humans use analogical comparison in learning generalizations. Moreover, prior computational approaches have demonstrated that generalizations [12] or schemas [10] can be learned through a series of analogical comparisons. Analogical generalization is thus an alternative from of learning general concepts compared to statistical approaches which typically rely on extracting abstractions from data alone using a large amount of data. In contrast to statistical approaches, analogical generalization can learn general concepts with only a small number of exemplars, which also more closely resembles human rates of learning [15]. Another advantage of analogical generalization is that it can abstract away insignificant elements even if they are frequent enough. For example, the color of an object does not matter for the pick-and-place task and analogical generalization will be able to abstract away this insignificant property as it appears in the majority of the exemplars.

The analogical comparisons done during the generalization process we describe here are done using the Structure Mapping Engine (SME) [5], which is a computational model of analogy based on Structure mapping theory [7]. Given two descriptions – in our case, one for the original exemplar and a counter-factual – SME aligns their common structure to find a mapping between the scenarios. This mapping consists of a set of correspondences between entities, attributes, and relations in the two scenarios. Structure mapping theory defines the principles of *systematicity* and *structural consistency* [7], which then provide the basis for the scoring of the analogical similarity. This score, called the *structural evaluation score*, combined with the simulation success score gives the overall score that is used to determine whether a scenario should be integrated into a generalization.

Algorithm 1 describes the overall process of creating a set of generalizations of an action and its context (β) given a set of counter-factual scenarios (T). Each scenario (τ) is compared with each generalization (γ). If the analogical similarity combined with the simulation success produces a score above a pre-specified threshold (θ), then the scenario is assimilated into the generalization. If the scenario is not sufficiently similar to any of the generalizations, then a new generalization is created with that scenario as the base of it.

3.5 Action Selection and Execution (E)

Given a new scenario in which the robot is to accomplish a goal, the robot must select the appropriate action. Selecting a generalized action requires comparing the overall context of the new scenario (including the goal, the objects involved, and the environment) to the generalized action. If the comparison concludes that there is a sufficient similarity between the generalization and the current scenario, then the generalized action is likely to be applicable. It has been shown that the specific details of the action that are necessary for executing the action may be inferred as part of an analogical comparison of the current scenario with the generalized scenario [21].

Algorithm 1 Generalization of β given T

```
1: function GENERALIZE(β, T) : set of generalizations
2:     γ ←newGen(β)
3:     Γ ← {γ}
4:     for all τ ∈ T do
5:         flag ←F
6:         for all γ ∈ Γ do
7:             μ ←compare(γ, τ)
8:             μ ← μ× simulationSuccess(τ)
9:             if μ > θ then
10:                assimilate (τ, γ)
11:                flag ←T
12:            end if
13:        end for
14:        if flag then
15:            γ ←newGen(τ)
16:            Γ ← Γ + γ
17:        end if
18:    end for
19:    return Γ
20: end function
```

4. PROOF-OF-CONCEPT IMPLEMENTATION ON THE ROBOT

We describe here the integration of this approach into a cognitive architecture and a demonstration of learning to adapt an action to a new object.

4.1 System Architecture

To demonstrate the utility of our approach we have implemented a proof-of-concept demonstration for a simple scenario which has been implemented using the DIARC (Distributed Integrated Affect, Reflection, and Cognition) architecture [18]. A detailed view of the DIARC implementation for this demonstration can be seen in Figure 5. Here, we have highlighted three major computational groups: (1) the *Physical World*, (2) *Simulation Management and Learning*, and (3) the *Simulated World*.

The architectural components in the *Physical World* make up the core set of capabilities for the physical robot. The *Simulation Management and Learning* group is responsible for setting up each simulation environment, running the simulations, collecting simulation results, and using the collected results to learn new actions. The *Simulation World* configuration used by the *Simulation Management and Learning* group is nearly identical to that of the *Physical World*. This is critical, and ensures that the actions explored in simulation are available on the physical robot, and provides a reasonable certainty of behavioral equivalence between the physical and simulated worlds. One notable difference is the simulated vision component. Because the simulation

setup component is responsible for initializing each simulation environment (based on the scene representation described above), perfect ground truth of each object and object location is known and no perceptual processing of video or point cloud data is necessary.

In addition to the DIARC cognitive architecture, we also use the Robot Operating System (ROS) to leverage existing low-level capabilities of this framework (e.g., robot model, motion planning). In particular, we make heavy use of ROS MoveIt! [19], a software package for mobile manipulation. The tight integration between DIARC and ROS is achieved through an off-line tool used to auto-generate rosjava nodes that can then be instantiated in DIARC components to create a seamless bridge between the two frameworks and allows easy use of the Willow Garage PR2 robot and Gazebo simulator.

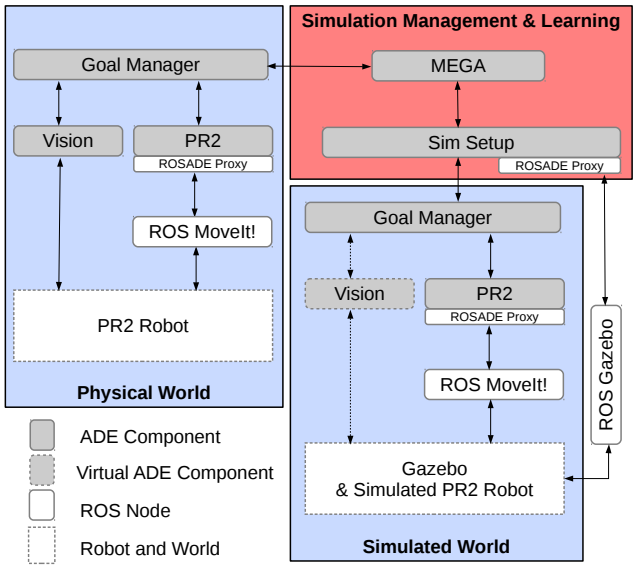

Figure 5: High-level view of the integrated architecture. The blue computational groups represent partial architecture configurations for the physical robot (left) and simulated robot (right). The salmon colored group contains novel algorithms for generalization through analogical reasoning.

4.2 Demonstration on the Robot

Our demonstration starts with an existing action representation for pick-and-place on a medical kit with one arm by the handle. While situated in an environment that contains a table and medical kit (located on the table), the physical robot begins by executing the action script to pick up the medical kit and move it aside.

This action representation is executed by the components in the *Physical World* group and requires the coordination of a goal manager component to execute the action script, and vision and robot components that provide perception and manipulation capabilities, respectively. The vision component is responsible for the detection of the table plane and medical kit (assumed to be on the table). Here, we use a Microsoft Kinect sensor to provide colored 3D point cloud (RGB-D) data, and perform plane detection to detect the table top. Once a table is detected, objects located on the table are segmented from the scene and classified against a predefined database of objects using global feature descriptors (Viewpoint Feature Histograms [17]). Once classified, an object is pose-aligned with the matching database object to transfer a list of known grasp poses to the new object. Given the information about the detected object along with information defined in the action script, the robot component is then able to plan and execute the motion commands for the pick-and-place action.

Next, the *Simulation Management and Learning* group begins the process of generalizing the given action script. The MEGA component determines the scenario (scene and action) to simulate, and then passes that information to a simulation setup component to instantiate the simulation scene. The modified action script is then passed to the goal manager component in the *Simulated World* group to execute the action in much the same way as described in the *Physical World* above. The results from the simulation run are collected by the MEGA component, and the process is repeated for each simulation. Once all the simulations have completed, the MEGA component carries out the generalization process and produces a generalized action script.

In general, the details of the scenarios instantiated and explored by the simulation process should be guided by the robot's current scenario, outcomes from previous simulation runs, and also the kinds of scenarios the robot expects to encounter. This is an open research question and we instead choose five predefined scenarios to simulate based on the scenario that is expected at the end of the generalization (i.e., picking up a cylindrical tub). For the scenario presented here, the simulations are able to achieve speeds of up to twice real time (using the Gazebo simulator on a machine with two Quad-Core i7 Xeon Processors and 24 GB RAM), and all five take less than five minutes in total. While this might seem like a considerable amount of time, the runtime would be considerably longer if these simulations were instead carried out on the physical robot. It should also be noted that each simulation was performed in serial, and the generalization process is amenable to parallel processing, where each simulation could run in parallel in future implementations.

Finally, to validate the usefulness of the generalized action script, the physical robot is presented with a new object (a small plastic tub), and the robot demonstrates that it can perform the pick-and-place action on the new object. The same process as described above for the medical kit pick-and-place is applied here, where instead the new generalized action script is executed which perform a two-handed grasp, successfully picking up the tub. If instead, the original medical kit action script was executed in an attempt to pickup the tub, a one-handed grasp would have been attempted resulting in a failed pick-and-place. A video of this proof-of-concept can be seen here: `https://youtu.be/hcm-nxnYd5k`.

5. GENERALIZATION EVALUATION

The proof-of-concept demonstrates that the robot can learn how to adapt an action to accommodate changes in the object. Our approach is capable of learning far more, including fundamental differences in the action representing *structural changes*. The purpose of this evaluation is to verify that the approach is able to recognize these structural differences and group structurally similar scenarios into the same general-

ization. An example of a structural difference is a step in an action being skipped, repeated, or added. If our approach were limited to learning the appropriate set of parameters to use to accomplish an action, then we would not be able to consider changes to the overall action that are not parameters. Allowing for changes in the structure of the action enables us to explore a broader range of actions that are still applicable to the scenario. The purpose of the evaluation is to confirm that the analogical generalization process will identify structural differences and construct an alternative generalization.

To evaluate this, we defined 22 scenarios to be generalized. These scenarios could be generated by the scenario generation process described above, could be an example of a scenario that has been attempted or was demonstrated, or could have been described to the robot. It does not matter what the source of the scenario is in the evaluation of the generalization. Most of the scenarios (19) resemble the original exemplar in that they had every action and every step of every action. Three of the scenarios represent pushing the object instead of picking it up and placing it. These scenarios skip the steps of lifting the arm and lowering the arm but the goal of moving the object from its original location to the new location is still achieved. All other action steps persist (reaching for the object, grasping it, moving the arm aside). Since we are strictly looking at the generalization process, we assume that each scenario would have successfully simulated. As such, the scenarios we constructed are ones in which the simulation would be successful.

When running the generalization process on the set of scenarios, two generalizations were constructed. One generalization closely resembles the initial action in that it has all of the same actions and steps. The second generalization is structurally different in that the two actions are absent. In both generalizations, the object is moved from the start location to the end location, thereby accomplishing the goal of the action. Table 2 summarizes the results of the evaluation. The number of predicates in the generalization is less than the average number of predicates in each exemplar because many of the terms are abstracted away. These predicates include insignificant descriptions of the object (e.g. color, mass, dimensions, etc.). We conclude from this evaluation that the generalizations have successfully abstracted away irrelevant information and that the generalization process is able to create generalizations for groups of scenarios that are more structurally similar.

Table 2: Summary of two generalizations constructed in evaluation. The first generalization is most similar to the original exemplar. The second generalization resembles a pushing action that still moves the object from the original location to the destination.

	Generalization 1	Generalization 2
No. exemplars	19	3
No. actions	9	7
Avg. no. predicates in exemplars	108.15	102
No. predicates in generalization	98	97

6. RELATED WORK

While there are other integrated architectures that use mental simulations (e.g., [2, 11]), none has been demonstrated to accomplish the kind of generalization learning from a single exemplar and immediate execution in a different scenario. The most closely related work is that of Kunze and his colleagues [13], where they use some of the same technologies (i.e. ROS, Gazebo, and the PR2) to conduct mental simulations in order to learn about actions.

The difference is that our work goes beyond learning parameters of actions, and attempts to construct a generalized representation of the action. This is in contrast to the approach taken by Kunze, who uses decision trees to classify data from simulations and crowdsourced data, to learn the appropriate parameters (e.g., angle to pour) for a desired outcome (e.g., size of pancake). Our approach is able to learn parameters (e.g., close position of gripper) but is also able to learn to adapt the structure of the action (e.g., adding or removing particular sub-actions). A push action learned from a pick-and-place action will, for example, have the lift-up and set-down sub-actions removed from its action representation, as they are not required to accomplish the goal.

It is this kind of structural adaptation that is necessary for two hands instead of one, or to push the object instead of pick it up. The end goal of our approach is not just to learn particular parameters (though this is a necessary step) but to learn the more general concept of an action for which a variety of parameters and sub-actions may work to accomplish the overall goal of the action.

An agent can learn a goal-directed sequence of actions using Reinforcement Learning (RL). For example, [16] demonstrates a robot learning a pick-and place behavior, but policies learned from RL are not necessarily general and applicable to a wide variety of novel scenarios. We demonstrate here that the robot can apply a generalized action in a scenario that it has not seen before.

We highlight the significance of the analogical generalization and contrast it with previous approaches to analogical generalization. Hummel and Holyoak [10] have a process of schema induction using analogical generalization, but their approach to analogy does not necessarily follow the principles defined in structure mapping theory [7]. These principles, *structural consistency* and *systematicity* enable the analogical generalization to recognize scenarios in which the actions are more structurally similar and group them into the same generalization.

Others have used the Structure Mapping Engine (SME) [5] to implement the principles of structure mapping theory in analogical generalization [12]. We are also using SME to do the analogies, and there are some similarities and differences in our generalization algorithm. In both algorithms, the similarity score generated by SME influences how or whether the exemplar is integrated into the generalization. Contrasting with previous approaches, we modify the similarity score with the simulation success rating before determining if an exemplar is to be incorporated into a generalization. There are also key differences in how an exemplar is integrated into a generalization. Previous work used a probability for each entity in a relation based on the frequency with which it occurred. The frequency of an element still affects how we incorporate elements in our algorithm, but the frequency is weighted by a significance score. Our work

associates a significance score with each entity, where this score is calculated using the overall similarity score, the score associated with each mapping within a comparison, and the simulation success rating.

We also note that the generalization process we have demonstrated requires a relatively minimal amount of knowledge. Some knowledge about objects, their properties, and possible values must be known. Additionally, knowledge of the available actions to the robot are required. We contrast this with other computational uses of analogical generalization that use a large knowledge base like OpenCyc [12, 6, 3]. In comparison, we require a small amount of knowledge but are still able to construct generalizations that the robot is able to use.

7. DISCUSSION

The integrated system together with the proof-of-concept evaluation demonstrates that it is possible for a robot to generalize actions from single exemplars and successfully execute these generalized actions on novel objects. Critical for this accomplishment was the integration of various components and component algorithms that had to be connected in the right way, bridging various representational and processing gaps. While these components and their interactions thus point to an integrated architecture that could enable unprecedented rapid learning for robots in the future, the current system is not there yet as several simplifications had to be made in the process of developing it. For one, the entire architecture was known a priori which allowed all software components to be manually instantiated. A more general approach would allow the architecture to introspect on the current *Physical World* configuration and dynamically replicate this configuration in a *Simulation World* when starting the learning process. Another shortcut was the prior knowledge of grasp points on the actual objects (in the real world) and the newly generated objects in the simulation. Ideally, these points would have to be determined based on perceptions only (e.g., using algorithms like [20]). In addition, background knowledge of what dimensions are available to be altered for simulation and which are important for generalization were given to the robot ahead of time, instead of letting the robot infer them from the task or to learn them. Finally, even though our system is able to generate mupliple generalizations, the proof-of-concept only uses one generalization. Previously we have demonstrated how to use analogy to select the appropriate generalization [21].

Paramount in our approach is the analogical generalization. Strictly learning parameters to an action does not allow for the robot to learn more novel (and perhaps creative) means of accomplishing the same goal. Our evaluation confirmed that the analogical generalization process is able to identify the structural differences and construct alternative generalized actions. While our evaluation was limited to removing action steps, other forms of structural changes would also be possible. This includes changing the order of the actions, adding constraints, or using some different actions. Additional constraints may include the spatial relation between the robot and the object or that the color of the target location must match the color of the object. Using a different action brings along with it a different set of arguments, preconditions, and postconditions. This could radically change the structure of the action representation, especially if there is little overlap in the preconditions and postconditions with other actions. Future work will explore the limits of the structural variations in the actions that can be adequately generalized.

Exploring a wide range of variations can be a time consuming process. We point out that the generation of scenarios, simulating them, and generalizing does not necessarily need to occur immediately after the initial action has been demonstrated. There are three time scales on which we envision this process to occur. The first is as demonstrated in our proof-of-concept, where immediately after the action has been demonstrated a small set of scenarios are generated and simulated to that a generalization may be constructed to meet immediate needs. The exploration of possible scenarios in this case can be constrained by the requirements of the current scenario. The second time scale allows the robot to create a generalization of the action that may be needed in the near future but there is no immediate need. While the robot is idle, it can explore the state space of scenarios and produce the appropriate generalizations. At a later time (minutes, hours, or days later), the robot may finally need to perform an action similar to the originally demonstrated action. At this time the robot can examine the set of generalizations that have been produced to see if any of them are applicable. If one is applicable it can be immediately applied, and if none are applicable then it can actively generate and simulate scenarios based on the constraints of the current situation. Lastly, the robot can actively explore a broad state space, requiring significant processing resources. This approach allows the robot to consider scenarios that differ greatly from the original action. In particular, the robot can consider scenarios with more structural differences, where different primitive actions are attempted or some action is repeated multiple times. This proactive exploration of the set of possible scenarios is likely to produce numerous generalizations, where some represent novel and creative ways to accomplish the same goal as the original action.

8. CONCLUSION

In this paper we demonstrate the first integrated robotic architecture that can learn generalized actions from a single exemplar and immediately apply this knowledge to perform modified actions on similar objects in different scenarios. In a next step, we intend to improve the architecture by addressing the shortcomings mentioned in the Discussion section and by parallelizing some the mental simulations to achieve better real-time performance. We also plan to improve the algorithms for devising counter-factual exemplars by integrating them with the goal and context knowledge of the architecture to allow for better determination of relevant scenarios that should be generated. Future work will also test the limits of the analogical generalization by considering larger numbers of scenarios and ones that significantly differ from the original. Finally, we plan to combine the current system with previous systems that could learn a new action from natural language instruction and observation to demonstrate the whole sequence of learning and generalization from a single exemplar.

Acknowledgments

This work was supported by ONR BRC No. N00014-14-l-0144.

REFERENCES

[1] P. Battaglia, J. Hamrick, and J. Tenenbaum. Simulation as an engine of physical scene understanding. *PNAS*, 110:18327–18332, 2013.

[2] N. L. Cassimatis, J. G. Trafton, A. C. Schultz, and M. D. Bugajska. Integrating Cognition , Perception and Action through Mental Simulation in Robots. *Robotics and Autonomous Systems*, 49(1):13–23, 2004.

[3] M. D. Chang and K. D. Forbus. Clustering Hand-Drawn Sketches via Analogical Generalization. In *Proceedings of the Twenty-fifth Annual Conference on Innovative Applications of Artificial Intelligence*, pages 1507–1512, 2013.

[4] S. El-Khoury and A. Sahbani. A new strategy combining empirical and analytical approaches for grasping unknown 3d objects. *Robotics and Autonomous Systems*, 58(5):497–507, 2010.

[5] B. Falkenhainer, K. D. Forbus, and D. Gentner. The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, 41(1):1–63, 1989.

[6] S. Friedman, J. Taylor, and K. Forbus. Learning Naive Physics Models by Analogical Generalization. In *New Frontiers in Analogy Research. Proceedings of the Second International Conference on Analogy*, pages 145–154, 2009.

[7] D. Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7:155–170, 1983.

[8] D. Gentner and J. Medina. Similarity and the development of rules. *Cognition*, 65(2-3):263–97, Jan. 1998.

[9] D. Gentner and L. L. Namy. Analogical Processes in Language Learning. *Current Directions in Psychological Science*, 15(6):297–301, Dec. 2006.

[10] J. Hummel and K. Holyoak. Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review*, 104:427–466, 1997.

[11] W. G. Kennedy, M. D. Bugajska, W. Adams, A. C. Schultz, and J. G. Trafton. Incorporating Mental Simulation for a More Effective Robotic Teammate. In *AAAI*, pages 1300–1305, 2008.

[12] S. E. Kuehne, K. D. Forbus, D. Gentner, and B. Quinn. SEQL: Category learning as progressive abstraction using structure mapping. *Proceedings of the 22nd Annual Meeting of the Cognitive Science Society*, pages 286–291, 2000.

[13] L. Kunze. *Naïve Physics and Commonsense Reasoning for Everyday Robot Manipulation*. PhD thesis, München, Technische Universität München, Diss., 2014, 2014.

[14] C. M. Liao and R. S. Masters. Analogy learning: a means to implicit motor learning. *Journal of sports sciences*, 19(January 2015):307–319, 2001.

[15] A. Lovett, K. Lockwood, M. Dehghani, and K. Forbus. Modeling human-like rates of learning via analogical generalization. *Analogies: Integrating Multiple Cognitive Abilities*, 5:35, 2007.

[16] M. Luciw, Y. Sandamirskaya, S. Kazerounian, J. Schmidhuber, and G. Schoner. Reinforcement and Shaping in Learning Action Sequences with Neural Dynamics. In *Development and Learning and Epigenetic Robotics (ICDL-Epirob), 2014 Joint IEEE International Conferences on*, pages 48–55, Chicago, 2014. IEEE.

[17] R. B. Rusu, G. R. Bradski, R. Thibaux, and J. M. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *IROS*, pages 2155–2162. IEEE, 2010.

[18] M. Scheutz, G. Briggs, R. Cantrell, E. Krause, T. Williams, , and R. Veale. Novel mechanisms for natural human-robot interactions in the diarc architecture. In *Proceedings of AAAI Workshop on Intelligent Robotic Systems*, 2013.

[19] I. A. Sucan and S. Chitta. Moveit! http://moveit.ros.org, 2015. Accessed: 2015-09-01.

[20] A. ten Pas and R. Platt. Using geometry to detect grasps in 3d point clouds. In *International Symposium on Robotics Reserach*, 2015.

[21] J. R. Wilson and M. Scheutz. Analogical generalization of activities from single demonstration. In *Advances in Artificial Intelligence*, 2014.