A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms

Oliver Kroemer [* 1] Scott Niekum [* 2] George Konidaris [3]

Abstract

A key challenge in intelligent robotics is creating robots that are capable of directly interacting with the world around them to achieve their goals. The last decade has seen substantial growth in research on the problem of robot manipulation, which aims to exploit the increasing availability of affordable robot arms and grippers to create robots capable of directly interacting with the world to achieve their goals. Learning will be central to such autonomous systems, as the real world contains too much variation for a robot to expect to have an accurate model of its environment, the objects in it, or the skills required to manipulate them, in advance. We aim to survey a representative subset of that research which uses machine learning for manipulation. We describe a formalization of the robot manipulation learning problem that synthesizes existing research into a single coherent framework and highlight the many remaining research opportunities and challenges.

1. Introduction

Robot manipulation is central to achieving the promise of robotics—the very definition of a robot requires that it has actuators, which it can use to effect change on the world. The potential for autonomous manipulation applications is immense: robots capable of manipulating their environment could be deployed in hospitals, elder- and child-care, factories, outer space, restaurants, service industries, and the home. This wide variety of deployment scenarios, and the pervasive and unsystematic environmental variations in even quite specialized scenarios like food preparation, suggest that an effective manipulation robot must be capable of dealing with environments that neither it nor its designers

have foreseen or encountered before.

Researchers have therefore focused on the question of *how a robot should learn to manipulate the world around it.* That research has ranged from learning individual manipulation skills from human demonstration, to learning abstract descriptions of a manipulation task suitable for high-level planning, to discovering an object's functionality by interacting with it, and many objectives in between. Some examples from our own work are shown in Figure 2.

Our goal in this paper is twofold. First, we describe a formalization of the robot manipulation learning problem that synthesizes existing research into a single coherent framework. Second, we aim to describe a representative subset of the research that has so far been carried out on robot learning for manipulation. In so doing, we highlight the diversity of the manipulation learning problems that these methods have been applied to as well as identify the many research opportunities and challenges that remain.
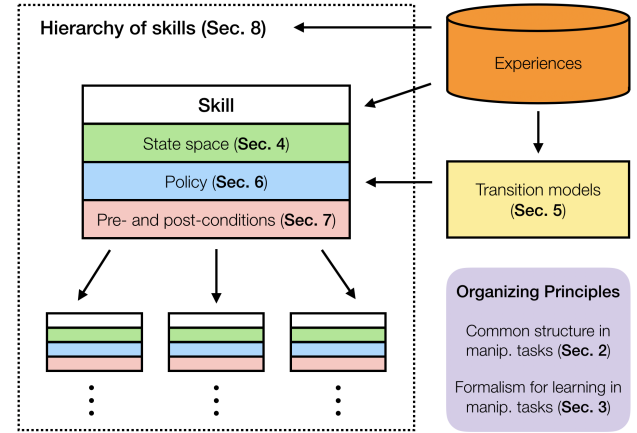

Figure 1. Overview of the structure of the review.

Our review is structured as follows. First, we survey the key concepts that run through manipulation learning, which provide its essential structure (Sec. 2). Section 3 provides a broad formalization of manipulation learning tasks that encompasses most manipulation problems but which contains the structure essential to the problem.

---
*Equal contribution
[1] Carnegie Mellon University
[2] University of Texas at Austin
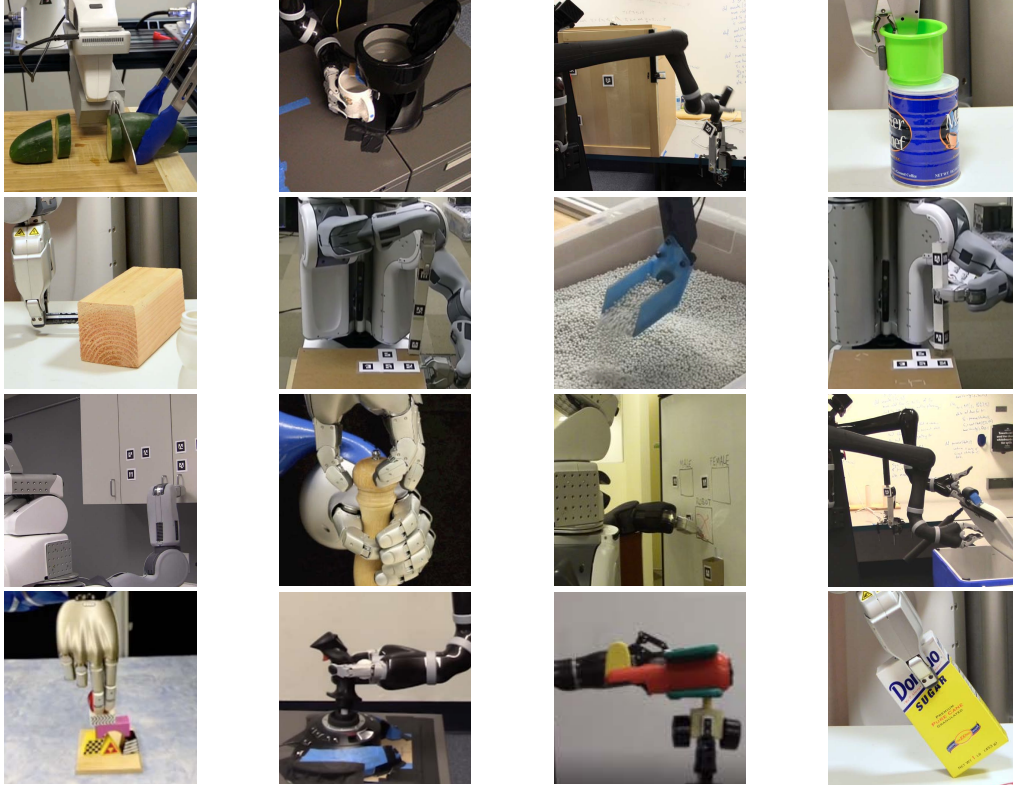[3] Brown University

Figure 2. Example manipulation skills including inserting, stacking, opening, pushing, cutting, screwing, pouring, and writing.

The remainder of the review covers several broad technical challenges. Section 4 considers the question of learning to define the state space, where the robot must discover the relevant state features and degrees of freedom attached to each object in its environment. Section 5 describes approaches to learning an environmental transition model that describes how a robot's actions affect the task state. Section 6 focuses on how a robot can learn a motor control policy that directly achieves some goal, typically via reinforcement learning [371], either as a complete solution to a task or as a component of that solution. Section 7 describes approaches that characterize a motor skill, by learning a description of the circumstances under which it can be successfully executed, and a model of the resulting state change. Finally, Section 8 surveys approaches to learning procedural and state abstractions that enable effective high-level learning, planning, and transfer.

2. Common Concepts in Learning for Manipulation

Manipulation tasks have significant internal structure and exploiting this structure may prove key to efficient and effective manipulation learning. Hence, before formalizing the manipulation learning problem, we will first discuss this internal structure.

2.1. Manipulations as Physical Systems

Every manipulation involves a physical robot interacting with its environment. As a result, all manipulations are subject to the laws of physics and the structure that they impose. This fairly obvious statement has wide-reaching implications for manipulation learning algorithms. Basic physical concepts (e.g., distinct objects cannot occupy the same space, and gravity applies a mass-dependent force to objects) provide strong prior knowledge for manipulation tasks. Concepts from physics, such as irreversible processes and object masses, are so fundamental that we generally take them for granted. However, these concepts provide invaluable prior knowledge and structure that can be exploited by learning algorithms and thus make learning manipulation skills tractable. Most of the concepts discussed in the remainder of this section are to some extent the result of manipulations being physical processes.

2.2. Underactuation, Nonholonomic Constraints, and Modes in Manipulations

Manipulation tasks are almost always characterized as underactuated systems. Each robot has a fixed set of actua-

tors that define its low-level action space, e.g. eight control variables for a fully actuated 7-DoF arm and 1-DoF gripper. Inanimate objects do not increase the action space, but every object in the environment does contribute a set of independent state variables to the state space. Hence, having even one free object in the environment will result in the state having more degrees of freedom than there are actuators, which means that the system is underactuated. This discrepancy between the numbers of actuators and DoFs increases as more inanimate objects are added to the environment.

The states of the different objects can however be altered by the robot through manipulation. The robot must first move into a state from which it can alter the object's state and then apply the desired manipulation, e.g., make contact with an object and then push the object. These requirements can be represented as a set of nonholonomic constraints that define how the robot can move through the full state space based on different interactions with the environment. Manipulation tasks can be modelled as hybrid systems, wherein the system dynamics are continuous within each of a number of discrete dynamical modes. The dynamics are thus piecewise continuous. The system jumps between these distinct modes depending on the current state.

In manipulation tasks, the mode switches often correspond to making or breaking of contacts, with different contacts applying corresponding constraints and allowing the robot to interact with various objects. Thus the robot must reach a suitable mode before it can perform a desired manipulation. Similarly, some mode switches should be avoided as they correspond to undesired effects, e.g., dropping a glass. Manipulation modes provide a modular structure to manipulation tasks with obstacles and subgoals often corresponding to mode transitions. Unfortunately, modes also make manipulation tasks inherently discontinuous. Hence, smoothness assumptions for generalization are often difficult to guarantee, and small changes in the state can have a significant effect on the outcome of a manipulation if the robot is near a mode transition. It is therefore important that robots monitor their skill executions for unexpected and undesired mode transitions, e.g., collisions with obstacles.

2.3. Interactive Perception and Verification

As autonomous embodied agents, robots are capable of selecting different actions and observing the resulting outcomes. The effects of these manipulations depend both on the selected action and the properties of the objects being manipulated. The robot can thus perceive latent object properties by observing the outcomes of different manipulation actions. This process is known as *interactive perception*.

Many properties of objects, such as material or kinematic constraints, can only be determined reliably through interactive perception. If the goal of the manipulation task is to alter one of these latent properties, then the robot will need to use interactive perception to verify that the manipulation was successful, e.g., perturbing a peg to ensure it was properly inserted into a slot. Even if a property's value can be approximated using passive perception, interactive perception often provides a more accurate estimate. In some cases, the estimate from the interactive perception can be used as the ground truth value for learning the passive perception. Interactive perception thus provides the basis for *self-supervised learning*. Interactive perception can be used to estimate object properties and predict the effects of different actions on objects.

As the perception depends on the action, selecting informative actions is a key challenge for using interactive perception in an efficient manner. Interactive perception is therefore often combined with *active learning*, i.e., the process of actively selecting samples to label to maximize learning performance.

2.4. Hierarchical Task Decompositions and Skill Reusability

Manipulation tasks exhibit a highly hierarchical structure. At the top of this hierarchy is a single primary task that defines the autonomous robot's fundamental purpose, e.g., maintain a household. This task can be divided into subtasks, such as clean the dishes, vacuum the floors, and dispose of trash. These subtasks can be further divided into their own subtasks. Cleaning the dishes can be divided into grasp the dishes, place the dishes in the dishwasher, and turn on the dishwasher. Even basic skills, such as grasping, can be further divided into multiple goal-oriented action phases. This hierarchy divides the primary task into smaller, more tractable problems. The robot can thus learn skill policies for performing the lowest level tasks and then in turn use these skills as an action basis for performing the next level of tasks. The complexity of the learning challenges at each level of the hierarchy is thus reduced, enabling faster skill learning. The robot can incrementally learn a hierarchical policy of skills, with the resulting policy hierarchy reflecting the task hierarchy.

In addition to reducing the complexity of learning individual tasks, the hierarchy is also important because it results in a modular structure. Subtasks and skills can often be interchanged to perform tasks in different ways depending on the scenario. Modularity also allows for some components to be predefined and others learned, e.g., an agent may be provided with a basic grasping reflex. More importantly, given a suitable hierarchical decomposition, similar tasks will often appear multiple times within the hierarchy. For

example, a food preparation task may involve many cutting subtasks (e.g., each slice of bread or vegetables), and a furniture assembly task will include multiple insertion subtasks (e.g., each peg and screw). Each cutting subtask will be different, but still similar enough that the robot should generalize over them rather than treating them as distinct. Similarly, the robot should generalize over the different insertion subtasks when assembling furniture.

We refer to sets of similar tasks as *task families*. By exploiting the similarity of these tasks, the robot can learn skills that are applicable to entire task families and can thus be reused multiple times. This approach allows the robot to collect more learning experiences for individual skills and master these skills over time. Without this modularity, the robot would need to continuously learn new skills for every slight novelty in a task.

Incorporating procedural modularity into the robot's controllers, and its model of when they can be used, is a major topic in learning for manipulation research. One key question is how such structure should be discovered by the robot. In this area, research ranges from recovering structure from demonstrations, to discovering it autonomously from exploration, to identifying direct measures of when to separate tasks (e.g., a mode switch when contact constraints change).

2.5. Object-Centric Generalization

One common structural assumption for manipulation tasks is that the world is made of up objects and that the robot's goals will typically be to modify some aspect or attribute of a particular set of objects in the environment. Consequently, *generalization via objects*—both across different objects, and between similar (or identical) objects in different task instances—is a major aspect of learning to manipulate. Object-centric representations of manipulation skills and task models are often sufficient to generalize across task instances, but generalizing across different objects will require both motor skills and object models that adapt to variations in object shape, properties, and appearance. In some cases this can be done implicitly—e.g., with a compliant gripper that automatically adapts its shape to that of an object during grasping.

One powerful approach to generalizing across objects is to find a representation under which we can consider a family of objects to be equivalent or identical, even though they vary substantially at the pixel or feature level. Such *abstract representations* are a promising but under-explored aspect of object-centric generalization.

2.6. Discovering Novel Concepts and Structures

Robots working in unstructured environments will often come across new types of objects. Depending on the actions afforded by these objects, the robot may be able to perform new tasks with these objects by learning new skills. Learning in these open world environments is thus not just a matter of the robot filling in gaps in its knowledge base. Instead, the scope of the knowledge bases will continue to expand, sometimes in unforeseen manners.

The ability to handle novel concepts is an important aspect of robot autonomy, as it allows robots to handle unforeseen situations. To operate efficiently, robots will need to be capable of generalizing and transferring knowledge from prior experiences to structure the learning processes for these new concepts. This transfer learning may require more abstract reasoning depending on the similarity of the new concept to previous ones. Fortunately, as explained in this section, manipulation tasks exhibit a significant amount of structure that an autonomous robot can leverage when learning manipulations.

3. Formalizing Manipulation Learning Tasks

Robot learning problems can typically be formulated as individual Markov Decision Processes (or MDPs),[1] described as a tuple:

$$(S, A, R, T, \gamma),$$

where $S \subseteq \mathbb{R}^n$ is a set of states; $A \subseteq \mathbb{R}^m$ is a set of actions; $R(s, a, s')$ is a *reward function*, expressing the immediate reward for executing action a in state s and transitioning to state s'; $T(s'|s, a)$ is a *transition function*, giving the probability distribution over states s' reached after executing action a in state s; and $\gamma \in [0, 1]$ is a discount factor expressing the agent's preference for immediate over future rewards. Here the goal of learning is to find a control policy, π, that maps states to actions so as to maximize the *return*, or discounted sum of future rewards $\sum_{i=0}^{\infty} \gamma^i r_i$, for that specific problem.

This formulation is commonly used because it captures a wide range of tasks, enabling researchers to develop broadly applicable general-purpose learning algorithms. However, as we have discussed in the previous section, the robot manipulation problem has more structure; a major task for robot manipulation researchers is to identify and exploit that structure to obtain faster learning and better generalization. Moreover, generalization is so central to

---

[1]We should note that the Partially-Observable Markov Decision Process (POMDP) [184] is a more accurate characterization of most robot learning problems. While these have been employed in some cases [163, 405, 406, for example], the difficulty of solving POMDPs, especially in the learning setting, means their usage here is so far uncommon.

manipulation learning that a single-task formulation is necessarily insufficient.

We therefore model a manipulation learning task as a structured collection of MDPs, which we call a *task family*. Rather than being asked to construct a policy to solve a single task (MDP), we instead ideally attempt to learn a policy that generalizes across the entire task family.

A task family is a distribution, $P(M)$, over MDPs, each of which is a *task*. The the action space is determined by the robot and remains the same across tasks, but each task may have its own state space and transition and reward functions. The reward function is typically formulated as a robot-dependent background cost function—shared across the entire family—plus a terminal reward that is specific to that task:

$$R_i = G_i - C,$$

where C is the general background cost function and G_i is the reward function for the ith task.

The state space of the ith task is written as:

$$S_i = S_r \times S_{e_i},$$

where S_r is the state of the robot and S_{e_i} is the state of the ith environment. S_{e_i} can vary from raw pixels and sensor values, to a highly pre-processed collection of relevant task variables. However, as described above, it is treated monolithically, with no internal structure.

The key to successful learning and generalization in manipulation is exploiting the structure present within a task and across the task family. Many task environments will consist of a collection of objects and the variables describing the aspects of those objects that are relevant to the task. It is therefore common to model the environment as a collection of object states, resulting in a more structured state space where S_{e_i} is partly factorized into a collection of relevant object states:

$$S_{e_i} = S_{w_i} \times \Omega_1^i \times ... \times \Omega_{k_i}^i,$$

where S_{w_i} is the state of the general environment, Ω_j^i is the state of the jth relevant object in task i, and task i contains k_i objects. The number of relevant objects may vary across, and occasionally within, individual tasks. Deciding which attributes and objects are relevant is an important and understudied question. The major advantage of modeling tasks in this factorized way is that it facilitates object-centric generalization, because policies and models defined over individual objects (or small collections of objects) can be reused in new environments containing the same objects.

Factorization and object-centric generalization can also be seen in the use of symbolic state representations. More abstract state representations tend to take on proposition-based (e.g., `CupOnTable=True`) or predicate-based (e.g., `On(Cup,Table)=True`) representations. These factorized representations support modularity as the robot often only needs to consider a subset of the symbols for any given skill or task. For manipulation tasks, we often employ predicate-based representations for their explicit generalization over objects.

Manipulation tasks also present modularity in their transition functions, i.e., the robot will only be able to affect a subset of objects and state variables from any given state. To capture the underactuated nature of manipulation tasks, we can model the tasks as hybrid systems with piecewise continuous dynamics. Each of the continuous dynamical subsystems is referred to as a *mode*, and the state will often contain discrete variables to capture the current mode. Mode switches occur when the robot enters certain sets of states known as guard regions, e.g., when the robot makes or breaks contact with an object. The robot can thus limit the state variables that it may alter by restricting itself to certain modes.

In some cases there is also structure in the action space, exploited through the use of higher-level actions, often called *skills*. Such skills are typically modeled using the options framework [373], a hierarchical learning framework that models each motor skill as an *option*, $o = (I_o, \beta_o, \pi_o)$, where:

- $I_o : S \to \{0, 1\}$ is the *initiation set*, an indicator function describing the states from which the option may be executed.

- $\beta_o : S \to [0, 1]$ is the *termination condition*, describing the probability that an option ceases execution upon reaching state s.

- π_o is the option policy, mapping states in the option's initiation set to low-level motor actions, and corresponding to the motor skill controller.

The robot may sometimes be pre-equipped with a set of motor skills that it can reuse across the family of tasks; in other settings, the robot discovers reusable skills as part of its learning task.

There is one additional complication. The requirement that we learn policies, skills, or general domain knowledge that applies across the entire task family leads to a dilemma. When learning to solve a single task, we can learn the relevant policies as functions of the task state. However, we must then transfer these functions across the tasks in the task family, which may not share a state space. Transfer may be aided by adding extra information to the task—for example, information about the color and shape of various objects in the task—but since that information does not

change over the course of a task execution, it does not properly belong in the state space. We model this extra information as a *context vector* τ that accompanies each task MDP, and which the robot can use to inform its behavior. Just like the state space, that vector can be monolithic for each task, or factored into object contexts.

Our model of a family of manipulation tasks therefore consists of a *task family* specified by a distribution of manipulation MDPs, $P(M)$. Each manipulation MDP M_i is defined by a tuple $M_i = (S_i, A, R_i, T_i, \gamma, \tau_i)$, where:

- $S_i = S_r \times S_{e_i}$, is the state space, where S_r describes the robot state, and S_{e_i} the environment state. Often the environment state is primarily factored into a collection of object states: $S_{e_i} = S_{w_i} \times \Omega_1^i \times ... \times \Omega_{k_i}^i$, for a task with k_i objects;

- A is the action space, common across tasks, which may include both low-level primitive actions and a collection of options O;

- $R_i = G_i - C$ is the reward function, comprising a background cost function C (common to the family) and a task-specific goal function G_i;

- T_i is the transition function, which may contain exploitable structure across the sequence of tasks, especially object-centric structure;

- γ is a discount factor, and

- τ_i is a vector of real numbers describing task-specific context information, possibly factored into object context: $\tau_i = \tau_1^i \times ... \times \tau_k^i$, for k objects.

If we are to learn policies, motor skills, and general domain knowledge that applies across the task family, rather than only to a specific task instance, then they cannot be defined directly as functions of the problem state. We must instead either use a specific subset of the state (e.g., a motor skill that can be applied to a specific object and ignores the remaining aspects of the state) or it must explicitly adapt to the information contained in the context vector (e.g., a motor skill that adapts to the shape contexts of different objects).

As an example of a task family, consider the problem of learning to open doors. Each door corresponds to a different task, with variations in the doors' shapes, kinematics, and masses being captured in the context vector. The current angle of the door and its articulated handle would then be part of the state. The robot could treat each door as an unrelated task and attempt to learn a policy for opening each one individually. However, a more efficient approach would be to learn transition models and motor skills that

explicitly take into account the context vector and generalize across all of the doors accordingly. In this manner, the robot would be able to reuse the skill directly for opening new doors as well (once the new contexts have been determined). The robot may also divide the task into two subtasks, i.e., turning the handle and pulling open the door, and learning a skill for each one accordingly.

Another example of a task family would be cutting vegetables. Cutting a single slice may be considered a separate task with its own transition and reward function. The robot should again attempt to generalize explicitly across the objects' different shape and material contexts, even though some cutting skills will work directly for a wide range of contexts. The robot may cut one or more vegetables at a time and, thus, the number of objects may change depending on the task. A suitable state abstraction can however simplify the learning problem and allow the robot to learn a policy that generalizes across different numbers of objects.

The learning problems posed by these and other manipulation tasks can typically be placed into one of five broad categories:

When **learning to define the state space** (Sec. 4), the robot must discover the state features and degrees of freedom attached to each object in its environment. This information is assumed to be given in the traditional reinforcement learning and planning settings. That is not the case in robotics, and in particular in learning for manipulation, which involves interacting with objects that the robot's designers do not have a priori access to. Learned representations of object states can be transferred across the task family as components of each task's state space.

When **learning a transition model of the environment** (Sec. 5), the robot must learn a model of how its actions affect the task state, and the resulting background cost, for use in planning. This is closely connected to learning to define the state space. If the learned transition models and reward functions are object-centric, then they can be ported across the task family, resulting in a natural means of object-centric generalization across tasks.

When **learning motor skills** (Sec. 6), the robot attempts to learn a motor control policy that directly achieves some goal, typically via reinforcement learning [371]. Here, the goal ranges from learning task-specific solution policies, to policies that can produce a solution to any task in the task family given the context vector, to useful motor skills that constitute a component of the solution policy but are not themselves a complete solution.

Given a learned motor skill, we may also **learn to characterize that motor skill** (Sec. 7), where the robot learns a description of the circumstances under which it can be successfully executed (often called preconditions, and cor-

responding to an option's initiation set I_o), and a model of the resulting state change (often called an effect).

Finally, **learning compositional and hierarchical structure** (Sec. 8) aims to learn hierarchical knowledge that enables the robot to become more effective at solving new tasks in the family. Here, the goal is to learn component motor skills—completely specified options—and models of their operation to construct more abstract representations of the learning task.

We now deal with each broad class of learning problem in turn.

4. Learning Object and Environment Representations

Modelling manipulation tasks and generalizing manipulation skills requires representations of the robot's environment and the objects that it is manipulating. These representations serve as the basis for learning transition models, skill policies, and skill pre- and post-conditions, as discussed in later sections.

This section explains how the object-based state and context spaces of manipulation tasks can be defined and learned. We will also explain how the robot can discover objects and estimate their properties using passive and interactive perception. Since many of the extracted object properties and features may be irrelevant to learning certain components of manipulation tasks, we conclude the section by discussing how to select and learn relevant features.

4.1. Object Representations

As discussed in Section 2, a robot's physical environment has considerable structure that it can exploit. In particular, the world can be divided into objects, each of which can be described by a collection of features or properties. Examples include movable objects such as mugs, tables, and doors, and stationary objects such as counters and walls. The robot can create a modular representation by segmenting the environment into objects and then estimating the values of their properties. This representation supports the reuse of skills by allowing the robot to efficiently generalize between similar objects across different tasks.

Object representations capture how objects vary both within tasks and across tasks of the same family. Within-task variations are captured by the *state* space—those features that a manipulation action can change; across-task variations are captured as part of the *context* space—attributes that are fixed in any specific task but aid generalization across pairs of tasks. For example, when stacking assorted blocks, the shapes and sizes of the blocks are fixed for a given task and are thus part of the context. Different stacking tasks may however use different sets of blocks, and thus the context changes across the tasks. Generalizing manipulation skills usually implies adapting, implicitly or explicitly, to variations in both context and state. For example, a versatile pick-and-place skill should generalize over the shapes of different objects (fixed in any specific task) as well as their positions in the environment (modifiable by actions during a task).

4.1.1. TYPES OF OBJECT VARIATIONS

Several types of within- and across-task object variations are common in the literature. **Object pose** [288, 237, 84] variations are the most common, and must be included in the state space when they can be manipulated (e.g., by a pick-and-place skill). However, in some cases, these can be fixed within a task but vary across the task family and thus belong in the context [73, 199, 180, 237] (e.g., the height of a surface in a wiping task [92]. **Object shape** may vary within a task, via articulated [278, 362, 191, 258, 363], deformable [334, 245, 338, 330, 244, 342], or divisible [236, 417, 427] objects. Rigid object shapes may also vary across tasks [49, 39], offering both a challenge to and opportunity for generalization [330]. Similarly, objects may vary in **material properties**, which can have significant effects on manipulation but typically only vary across-tasks (e.g., cutting objects of different materials [236]), though there is some work on altering the material properties of manipulated objects [63, 332, 168].

Finally, objects may vary in their **interactions or relative properties**, including both robot-object [26, 209] and object-object [358, 362, 220, 183] interactions. Objects may interact with each other resulting on constraints between them [170], and manipulation may result in mode switches that add or remove constraints [278, 22, 222, 20]. Variance across, rather than within, tasks is also possible—e.g., the properties of the joint connecting a cabinet and its door will remain constant for a given cabinet, but different cabinets can have different constraints [362, 146, 278, 74].

4.1.2. OBJECT REPRESENTATION HIERARCHIES

Object models can be represented hierarchically, with layers corresponding to point-, part-, and object-level representations, each decreasing in detail and increasing in abstraction, and affording different types of generalization. This representation hierarchy mirrors the geometric structure of objects and their parts. Geometric properties and features capture *where* the points, parts, and objects are, and non-geometric properties tend to capture the corresponding *what* information defining these elements. In addition to representing individual objects, the robot can also represent interactions between objects at the different layers of the hierarchy.

Point-level Representations: are the lowest level of the hierarchy, and include point cloud, pixel, and voxel representations for capturing the partial or complete shapes of objects in detail [334, 34, 195, 114]. Point-level representations provide the robot with the most flexible representations for capturing important details of objects and manipulation tasks.

Each element of these representations may be associated with additional features, such as the color or material properties corresponding to this point. Segmentation methods can be used to assign labels to individual points according to which part or object they belong [330]. Interactions may be modeled at this level of the hierarchy as contact points [192, 75, 209, 408, 364, 403, 298].

Generalization across tasks can be accomplished by establishing correspondences between objects and environments at the point level. The robot may identify individual key points of objects (e.g., the tip of a tool [99]), or use non-rigid registration or geometry warping to determine correspondences of entire sets of points across task instances [338, 156, 361, 315, 12]. These correspondences can then be used to directly map manipulation skills between tasks or to compute higher-level representations [361].

Part-level Representations: correspond to sets of multiple contiguous points from the lower level of the hierarchy [370, 87, 74, 51], typically focusing on parts associated with certain types of manipulations. For example, a mug can be seen as having an opening for pouring, a bowl for containing, a handle for grasping, and a bottom for placing [105]. Each part can then be described by a set of features describing aspects such as shape, pose, color, material, type, and surface properties. Robots can use part-level features to represent interactions or relations between the parts of different objects, or to indicate the types of interactions and constraints imposed between the parts due to the interaction (e.g., a peg must be smaller than the hole for an insertion task).

Defining correspondences at the part level enables generalization across different types of objects [382, 369, 87]. Many objects have similar parts that afford similar interactions, although the objects may be very different as a whole. For example, a coin and a screwdriver are distinct types of objects, but they both have short thin metallic edge that can be used to turn a screw. Similarly, many objects have graspable handles; identifying the handle correspondences thus allows for transfering grasping skills [87, 217]. Part-based representations therefore allow the robot to generalize between different classes of objects without having to reason about individual points of the object [369].

Object-level Representations: are important because robots generally select objects, not individual features, to manipulate [174, 84, 119, 173]. Thus, robots must generalize between the sets of features attached to each object. Useful object-level representations group together object-specific properties such as an object's pose, mass, overall shape, and material properties (for uniform objects). Similar to parts, object-level interaction features often define the relative poses, forces, and constraints between objects in manipulation tasks, e.g., relative poses for stacking blocks [107, 170, 394, 367]. The robot may also define different types of interactions at this level (e.g., object A is *on* or *inside* object B) or relations between objects (e.g., relative sizes and weights) to capture sets of interactions in a more abstract form [329, 330, 225, 332, 109]. Generalization across objects typically requires establishing correspondences between distinct objects [89] that support similar manipulations.

Robots may also need to represent **groups of objects**—rather than modeling individual objects within the group, it is often more efficient and robust to use features that represent groups of objects as a whole entity. For example, clutter, or the background scene with which the robot should avoid colliding, is often represented in a similar manner to a single deformable or piece-wise rigid object. Identifying and reasoning about individual objects may not be necessary, and may even add additional complexity to learning, causing learning to generalize poorly or be less robust. Manipulating a specific object among a group may require the robot to recognize it, or even actively singulate the object [137, 153].

4.2. Passive and Interactive Perception

As embodied agents capable of manipulating their surroundings, robots can use actions to enhance their perception of the environment. Robot perception is therefore broadly divided into passive and interactive perception, with the key difference being whether or not the robot physically interacts with the environment.

Passive perception refers to the process of perceiving the environment without exploiting physical interactions with it, i.e., non-interactive perception [168]—for example, recognizing and localizing objects in a scene based on a camera image [49, 392, 411, 426]. Passive perception allows the robot to quickly acquire a large amount of information from the environment with little effort. Passive perception does not require the environment or the sensor to be stationary; observing a human performing a manipulation task is still passive, as the robot does not itself perform the interaction. Similarly, moving a camera to a better vantage point is still a non-interactive form of perception, as the robot does not apply force to, or otherwise alter, the state of the environment[323, 185].

In **interactive perception** [33], the robot physically in-

teracts with its surroundings to obtain a better estimate of the environment. For example, a robot may push an object to better estimate its constraints or lift an object to estimate its weight [21, 146, 190]. Robots can use a wide range of sensor modalities to observe the effects of its interactions, including haptic, tactile, vision, and audio [235, 63, 159, 128, 388].

The need to perform actions means that interactive perception requires more time and energy than passive perception. The benefit of interactive perception is its ability to disambiguate between scenarios and observe otherwise latent properties [393, 63, 119], enabling the robot to reduce uncertainty. For example, the robot may not know whether two objects are rigidly connected or simply in contact; interactive perception allows it to test each hypothesis.

Different actions will result in different effects and hence robots can learn about their environment faster by selecting more informative actions [21, 22, 95, 282, 226, 323, 193]. For example, shaking a container will usually provide more information regarding its contents than a pushing action would [332, 348, 333]. Active learning approaches usually estimate the uncertainty of one or more variables in the environment and then select actions based on the resulting entropy, information gain, or mutual information [282, 159, 146, 399, 226].

The ability to test hypothesis means that interactive perception can also be used as a supervisory signal for learning to estimate properties using passive perception [128, 301, 399, 276, 419, 213, 292]. As an example of interactive learning, a robot may learn to predict the mass of objects from their appearance by first using interactive perception to determine the masses of a set of training objects. This form of self-supervised learning allows the robot to gather information autonomously and is thus crucial for enabling robots to operate in unfamiliar environments.

4.3. Learning About Objects and Their Properties

Having explained the different types of object variations and types of perception, we now discuss how the robot can learn about the objects around it from data.

4.3.1. DISCOVERING OBJECTS

A common first step in learning is distinguishing the individual objects in the scene, which is a segmentation problem, typically accomplished using passive perception [213, 9, 341, 51, 152]. However, objects may often be close together in the scene, presenting the robot with ambiguous information about object identity. Here the robot can maintain a probabilistic belief over whether or not different parts belong to the same object [399, 193, 147], and use interactive perception or viewpoint selection to disambiguate the scene [132, 153, 399, 147].

4.3.2. DISCOVERING DEGREES OF FREEDOM

Once individual objects have been identified, the robot may need to identify their kinematic degrees of freedom [278, 147, 191, 22, 362, 190]. These constraints and articulated connections are fundamental to establishing the objects' state space for manipulation tasks, as well as for robust pose tracking [85, 335]. The different types of joints are usually represented using distinct articulation models, e.g., revolute or prismatic, with their own sets of parameters. For example, a revolute joint model is specified by the position, direction, and limits of its rotation axis [362, 21, 278, 363]. The robot must estimate these context parameters to accurately model the degrees of freedom. Passive perception can be used to estimate the kinematic chains of articulated objects, especially if the object is being moved by a person [258, 278, 299, 42]. Interactive perception is also well suited for estimating the articulation model parameters and the resulting motion constraints [362, 21, 146]. Given the high-dimensional parameter space of the articulation models, active learning approaches are often used to select informative actions for quickly determining the model parameters [146, 22].

4.3.3. ESTIMATING OBJECT PROPERTIES

Once the robot has identified an object in the environment, the next step is to estimate the object's properties. Since some properties are only applicable to some classes of objects [91, 47, 95, 30, 332, 419, 63] the robot must first recognize the object class. For manipulation tasks, object classes are often derived from the actions and interactions that they afford the robot, e.g., container, graspable, and stackable, such that the interaction-relevant properties can be easily associated with the objects.

Here the robot may use passive and interactive perception to estimate object property values [119, 168, 242, 401]. In addition to object recognition, passive perception is often used to estimate the position, shape, and material properties [244, 30, 419, 392, 120, 335, 48, 169, 49, 422, 426]. However, additional interactive perception can often be used to acquire more accurate estimates of these properties [62, 31, 95, 175, 297, 212]. Material and interaction properties are often difficult to estimate accurately using only passive perception. The estimates can be significantly improved by interacting with the objects and using tactile sensing [347, 346, 348, 368, 119, 235, 164, 401]. Exploratory actions can also be used to estimate the dynamic properties of objects, e.g., the center of mass, or the contents of containers [134, 333, 63].

4.4. Feature Learning and Selection

Even though an environment may contain a lot of objects and sensory stimuli, only a few object properties or sensory signals will usually be relevant for a given task. For example, when opening a bottle, the size of the cap is a relevant feature, but the color of the chair is irrelevant. Using a suitable set of relevant features simplifies the skill and model learning problems. It also increases robustness and generalization to new situations. If the set of object properties is sufficiently rich, then the robot may only need to *select* a suitable set of these as features for learning [89, 223, 268, 350, 359]. In many cases, the robot will however need to *learn* a set of features for the given task.

Unsupervised feature learning methods extract features from unlabeled training data. Dimensionality reduction and clustering techniques are two common types of unsupervised learning [48, 29, 181]. Dimensionality reduction methods may be used to capture correlations in data and discard noisy components of signals [67, 181]. Clustering methods may be used to cluster together distinct types of objects or continuous effects of manipulations [394].

Supervised approaches learn features as part of the overall model or skill learning process. Decision trees and neural networks are often used to learn features in a supervised setting. Deep learning in particular has become ubiquitous for feature learning in robotics. Different network structures and layers, such as auto-encoders, spatial soft max layers, convolutions, and segmentation masks, can be used to incorporate architectural priors for learning useful features [51, 110, 152]. Deep neural network models can be used to represent state spaces for manipulation tasks with high dimensional observation spaces [230, 353]. Neural networks are also highly effective at combining data from multiple sensor modalities or information sources [119, 233, 236, 370, 369, 240, 173].

5. Learning Transition Models

The goal of manipulation tasks is to alter the states of objects in the robot's environment. Learning transition models that capture the changes in state as a result of the robot's actions is therefore an important component of manipulation learning.

5.1. Representing and Learning Transition Models

Learning a transition model requires a suitable representation. The general form of a transition model for a family of MDPs is either a deterministic function $T : S \times A \to S$ or a stochastic distribution $T : S \times A \times S \to \mathcal{R}$ over the next states given the current state and action. The transition models can also depend upon the context vector τ in order to explicitly generalize between contexts. A transition model is often employed iteratively to perform multi-step predictions.

Continuous Models: In manipulation tasks, robots are generally operating in continuous state and action spaces, e.g., the state is given by the continuous poses of objects and the actions are given by desired joint positions or torques. Robots therefore often learn low-level transition models for predicting the next state as a set of continuous values, even when the set of actions is discrete [336]. Regression methods, such as neural networks, Gaussian processes, and (weighted) linear regressions, are commonly used for learning transition models [334, 237, 83, 15].

The simplest models, such as linear regression models, can be learned from small amounts of data [356], but often underfit the inherent nonlinearities of most manipulation tasks. Hence, the generalization performance of these models is limited. Instead, time-dependent linear models are often used to learn local models for specific regions of the state space [227], e.g., around the trajectory of a skill being learned. These models may be sufficient while maintaining low data requirements, and can be combined with prior models that capture more global information about the task. Gaussian mixture models and neural networks have both been used to learn such informative priors [117].

Nonparametric models, including Gaussian processes and locally weighted regression, can flexibly learn detailed transition models [84, 210]. The generalization performance of these types of models depends heavily on their hyperparameters and is often fairly limited. However, local models can be learned from relatively small amounts of data. For example, an accurate model for planar pushing of a block may be learned from less than a hundred samples [23].

More complex hierarchical models, such as neural networks and decision trees, allow the robot to learn task-specific features that provide better generalization [235, 334, 117, 174, 321, 110, 230, 353, 431]. For example, convolutional encoder-decoder networks can be used to learn models for predicting the transitions of individual points in the scene based on the selected actions [51, 110]. However, unless the models are pre-trained, learning intermediate features significantly increases the amount of data required for training these models. Broadly speaking, the best representation for learning a continuous transition model often depends on the amount of training data available and the amount of model flexibility required by the task.

Predictive state representations (PSRs) allow the robot to model the distributions over future observations based on the history of past observations [249, 349, 36, 357]. These models thus capture latent state information, as part of a POMDP formulation, without explicitly modeling or infer-

ring the latent state. The robot can thus work directly with the observed variables.

More advanced parametric models provide additional structure, up to including entire physics engines [418, 337, 419, 243]. These engines provide a significant amount of prior structure and information, which improves generalization and data efficiency. However, even given the engine, it is difficult for a robot to map a real world scenario into a simulation environment. The simulation may also still fail to accurately capture certain interactions due to their lack of flexibility. Physics engines are therefore sometimes used to provide a prior or to initialize a more flexible representation by combining analytical models with flexible data-driven models [5].

Discrete Models: Discrete transition models are used to learn transitions for tasks with discrete state and action spaces, typically capturing high-level tasks. For example, a robot may learn a discrete transition model for flipping a pancake, in which the transition probabilities indicate the probability of the pancake being flipped, dropped, or not transitioning. The model can then be used to predict the outcome of attempting multiple flips. Discrete models can also capture low-level transitions where the state space has been discretized. This results in the continuous states corresponding to a single discrete state being considered as identical. It also removes the smoothness assumption between states, such that state transition information is not shared between discrete states. As a result, a very fine-grained discretization will lead to poor generalization and a coarse discretization may not capture enough detailed transition information to perform the task. A suitable discretization should therefore cluster together states that will result in similar transitions given the same actions [29]. Adaptive or learned discretizations are a form of feature learning [133].

The most basic representations for discrete models are tabular models or finite state machines. Discrete models treat each state as atomic, with a transition distribution independent of all other states. This approach however severely hampers generalization and transfer. More structured models use collections of symbols to represent discrete state spaces. Proposition-based models define a set of propositions [208], e.g., `Obj1Grasped` and `Obj2Clear`, which are either true or false and thus define the state. A set of N propositions thus defines a set of 2^N states. The factorization provided by the propositions results in a more compact representation of the state space, and also allows transition information to be shared between these states. In particular, symbolic models are often used to express compact (and sometimes stochastic) action models known as *operators*. Each such operator has a a precondition that must be true for the action to be executable, e.g., `WaterReservoirFull=True`

and `CoffeeJugFull=False` for a coffee making action. The effects of the action are then given by lists of propositions that become true (e.g., `CoffeeJugFull`) and that become false (e.g., `WaterReservoirFull`), with all other propositions being unchanged by the transition. As the preconditions and effects only refer to a small number of propositions, the operator generalizes across the values of all other propositions that are not mentioned.

These models may be generalized to relational or first-order models [231], which support extremely compact action models and enable generalization across objects. This approach allows predicate-based models (e.g., `Full(CoffeeJug)` and `Full(WaterReservoir)`) to be used for families of tasks with different numbers of objects and states. This extended generalization can unfortunately also have some unintended effects if distinct situations are treated as the same. For example, a dishwasher loading action would treat a paper plate the same as a ceramic plate unless a suitable predicate, e.g. `IsCeramic(Plate)`, were included to avoid this type of over generalization.

Hybrid Models: Hybrid models combine elements of discrete and continuous models, often resulting from the hierarchical structure of manipulation skills and subtasks. The discrete components of the state are often used to capture high-level task information while the continuous components capture low-level state information.

Robots will often learn continuous models for specific subtasks. For example, a robot may learn separate models for opening a door, moving its body through free space, and stacking newspapers. These models can then be used to predict the effects of actions within each skill. In turn, these models can then become a part of the robot's overall world model to predict the effects of sequences of skills. This approach allows the robot to exploit the modularity of the distinct subtasks for the discrete high-level transition model. Many works on learning continuous models for individual skills or subtasks can thus be seen as learning part of a larger hybrid model.

One common use of hybrid models is to represent modes in manipulation tasks [145, 345, 219, 232, 65, 438]. In these cases, the dynamics are piece-wise continuous with jumps between the modes. These jumps are often defined by guard sets wherein the state transitions to a certain mode if the continuous state enters the guard set. In manipulation tasks, the jumps between modes typically result from contacts being made or broken between objects. The mode switches thus capture the changing dynamics and constraints caused by the changing contacts. Planning approaches for manipulation domains often explicitly take this multi-modal structure into account for planning [391, 171]. Learning a hybrid model entails learn-

ing both the dynamics within each mode and the conditions for switching between modes. The dynamics within a mode can be learned using standard continuous model approaches. The guard regions can be modeled as explicit sets of states or using classifiers. More flexible models, such as nonparametric and neural network models, may also be able to implicitly capture hybrid dynamics of manipulation tasks [110]. Exploiting the modularity of they hybrid system is however more difficult when using an implicit model of the mode structure.

5.2. Stochasticity and Uncertainty in Transition Models

State transitions are often represented using probabilistic models that allow the robot to represent multiple potential outcomes. For example, a robot may choose to move to a trash can and drop an object inside because the resulting state distribution has a lower variance than attempting to throw the object into the trash. When working with probabilistic models, it is important to distinguish between two sources of uncertainty: stochasticity and model uncertainty.

Stochasticity: A stochastic process exhibits a randomness in its state transitions. For example, when throwing a block into the air, the side on which the block will land is random. Similarly, due to slight variations in the execution and surface properties, the exact location of an object being placed on a table may follow a Gaussian distribution. Attempting to open a door may only result in the door opening 98% of the time and remaining closed the other 2%.

Stochasticity is common in manipulation tasks, for both discrete and continuous state spaces [210, 231]. Probabilistic models are thus used to capture the process noise and approximate the distributions over next states [17, 219].

Model Uncertainty: In addition to the inherent stochasticity of manipulation, the outcome of an action may be uncertain due to the robot's limited knowledge of the process. For example, consider the task of placing a closed container into a bowl of water. Whether the container will float or sink is deterministic given its mass and volume. However, the robot's prediction over the two outcomes may still be uncertain if the robot does not know the mass of the container or because it lacks sufficient training samples to accurately predict the mass threshold at which the container will sink. The probabilistic prediction is thus the result of model uncertainty rather than stochasticity. Unlike stochasticity, model uncertainty can be reduced given additional information, e.g., using interactive perception to better estimate the mass of the container or acquiring additional training samples for estimating the threshold for the model.

Bayesian approaches explicitly capture the uncertainty in their predictions by representing the probability distributions over different potential models [84, 337]. They also incorporate explicit prior beliefs over these model distributions. By contrast, point estimates, such as maximum likelihood or maximum a posteriori MAP estimates, assume a single model given the current data and use this model to predict the next state. Linear Bayesian models are often too simple to capture the complexity of manipulation tasks. Kernel-based Gaussian processes have therefore become one of the most popular Bayesian approaches for learning transition models [84, 159]. Gaussian processes, as well as linear Bayesian regression, both have explicit parameters modeling the stochasticity of the system being represented. The model uncertainty can also be used to guide exploration for learning an accurate model from fewer samples [412]

5.3. Self-supervision and Exploration for Learning Transitions

Transition models are usually learned in a self-supervised manner. Given the current state, the robot performs an action and observes the resulting effect on the state. The robot can thus acquire state, action, and next state tuples for training the model. To generalize more broadly, the robot will also need to estimate the context parameters for each task and then incorporate these into the model as well.

The robot can adopt different exploration strategies for acquiring samples. Random sampling is often used to learn general models and acquire a diverse set of samples. A grid approach may be used to ensure that the action samples are sufficiently spread out, but these approaches generally assume that the state can be reset between actions [432]. Active sampling approaches can be used to select action samples that are the most informative [412]. If the model is being used to learn and improve a specific skill, the robot may collect samples within the vicinity of the skill using a small amount of random noise (see model-based skill learning in the next section). Another increasingly popular approach to exploring an environment is to use intrinsic motivation [58, 293]. In this case, the robot actively attempts to discover novel scenarios where its model currently performs poorly or that result in salient events.

5.4. Transferring and Reusing Transition Models

Transition models are not inherently linked to a specific task, and can therefore often be transferred and reused between different manipulation tasks and even task families. For example, a model learned for pushing an object may be used as the basis for learning push grasps of objects. In order to directly reuse a model, the learned model and the new task must have the same state, action, and context spaces, or a mapping between the spaces may be necessary [377]. Given compatible spaces, the ability to transfer or

reuse models depends on the overlap between data distributions for training the old and new task models. This issue is known as covariate shift (input varies) and dataset shift (input and output varies). If the previous model was trained on data distinct from the needs of the new task, then the benefit of using the previous model will be limited and may even be detrimental. Assuming a sufficiently rich model, the robot can acquire additional data from the new task and use it to update the model for the new region [84]. In this manner, the robot's learned model may become more applicable to other tasks in the future. Data can also be shared across models when learning skills in parallel.

6. Learning Skill Policies

All of the robot learning problems discussed thus far are ultimately in service of helping the robot learn a policy that accomplishes some objective. Thus, the final learning goal for the robot is to acquire a behavior, or *skill controller*, that will perform a desired manipulation task. A common representation for skill controllers is a *stochastic policy* that maps state-action pairs to probabilities (or probability densities in the continuous case). This section discusses the spectrum of different policy parameterizations that can be chosen, algorithms for learning skill policies from experience and demonstrations, methods for transferring knowledge across skills and tasks, and approaches that can provide safety and performance guarantees during learning.

6.1. The Spectrum of Policy Structure

In robotic manipulation, specific parameterizations are often used that restrict the representational power of the policy; if these restrictions respect the underlying structure of the task, generalization and data efficiency are often improved without significantly impacting asymptotic performance. Thus, the choice of policy representation is a critical design decision for any robot learning algorithm, as it dictates the class of behaviors that can be expressed and encodes strong priors for how generalization ought to occur. This results in a spectrum of policy structures, ranging from highly general (but often sample-inefficient) to highly constrained (but potentially more sample-efficient) representations.

Nonparametric Policies: The most expressive policy representations are nonparametric, growing as needed with task or data complexity. This category includes nearest neighbor-based approaches, Gaussian processes [310], Riemannian Motion Policies [311], and locally-weighted regression [16, 407]. These representations are the most flexible and data-driven, but they also typically require large amounts of data to produce high-quality generalization. This representational paradigm has been successful in manipulation tasks in which very little task

knowledge is given *a priori* to the robot, such as an 88-dimensional octopus arm control problem in which Gaussian Process Temporal Difference learning is used [100], and 50-dimensional control of a SARCOS arm via locally weighted projection regression [407].

Generic Fixed-size Parametric Policies: More commonly, fixed-complexity parametric policy representations are used, making stronger assumptions about the complexity and structure of the policy. Common parametric policy representations include look-up tables [371], linear combinations of basis functions such as tile coding, the Fourier basis [207, 203], neural networks [240], decision tree classifiers [306, 166], and support vector machines [71, 316]. While any given choice of parameters has a fixed representational power, a great deal of engineering flexibility remains in choosing the number and definition of those parameters. Design decisions must also be made about which features will interact and which will contribute independently to the policy (e.g. the fully-coupled vs. independent Fourier basis [207]), or more general decisions regarding inductive bias and representational power (e.g. neural network architectures).

The choice of a fixed parameterization makes particular assumptions about how generalization ought to occur across state-action pairs. For example, tabular representations can represent any (discrete) function, but do not intrinsically support generalization to unseen states and actions. By contrast, policies comprised of linear combinations of basis functions (for both discrete and continuous state spaces) more naturally generalize to novel situations since each parameter (a basis weight) affects the policy globally, assuming basis functions with global support. However, the success of such generalization relies on the correctness of assumptions about the smoothness and "shape" of the policy, as encoded by the choice of basis functions. Some parametric forms make more specific structural assumptions—for example, by construction, convolutional neural networks [215, 240, 4, 256] support a degree of invariance to spatial translation of inputs.

Restricted Parametric Policies: The expressiveness of a policy is only limited by the underlying policy representation; however, sample complexity, overfitting, and generalization often get worse as representational power increases. This has lead to the development of specialized policy representations that have limited representational power, but exploit the structure of robotics problems to learn and generalize from less data. However, the quality of the generalization in such methods is highly dependent upon the accuracy of the underlying assumptions that are made. Such methods lie on a spectrum of how restrictive they are; for example, structured neural network architectures (e.g. value iteration networks [375], schema net-

works [188], etc.) impose somewhat moderate restrictions, whereas other methods impose much stronger restrictions, such as forcing the policy to obey a particular set of parameterized differential equations [326]. However, it is not always clear how to choose the right point along the spectrum from a completely general to highly-specific policy class. The following are some common examples of restricted policy classes.

Linear Quadratic Regulators (LQR) [437] are commonly used to stabilize around (possibly learned) trajectories or points, in which the cost is assumed to be quadratic in state and the dynamics are linear in state. These restrictions are relaxed in Iterative LQR [117] and Differential Dynamic Programming [376, 429], in which nonlinear dynamics and nonquadratic costs can be approximated as locally linear and locally quadratic, respectively. LQR-based controllers have also been chained together as LQR-trees, allowing coverage of a larger, more complex space [379]. LQR-like methods also assume that the state is fully observable; Linear Quadratic Gaussian (LQG) control methods further generalize LQR by using a Kalman filter for state estimation, in conjunction with an LQR controller [239, 303]. LQR/LQG controllers are optimal when the linear-quadratic assumptions are met, but require full knowledge of the dynamics and cost function, as well as a known trajectory around which to stabilize. Thus, generalization in this case simply translates to being able to stabilize around this known trajectory optimally from anywhere in the state space.

Rather than requiring a known trajectory, another common policy formulation, Dynamic Movement Primitives (DMPs), can learn a policy via demonstrations and reinforcement learning [326, 288]. DMPs leverage the fact that many robotic movements can be decomposed into two primary parts—a goal configuration in some reference frame and a "shape" of the motion. For example, screwing in a screw requires a turning motion to get to the desired final configuration with respect to the hole. DMPs use a set of differential equations to implement a spring-mass-damper that provably drives the system to an adjustable goal from any starting location, while also including the influence of a nonlinear function that preserves the desired shape of the movement. DMP controllers can be learned from very little data, but typically only generalize well in cases in which a good solution policy broadly can be described by a single prototypical motion shape primitive.

Whereas DMPs generate deterministic policies, other approaches treat the problem probabilistically, allowing stochastic policies to be learned. ProMPs are a straightforward probabilistic variant of DMPs that can produce distributions of trajectories [286]. Gaussian Mixture Regression (GMR) [54] models the likelihood of states over time

as a mixture of Gaussians, allowing for a multimodal distribution of trajectories that can encode multiple ways of performing a task, as well as the acceptable variance in different parts of the task.

Goal-based Policies: At the far end of the spectrum, the most restrictive policy representations are primarily parameterized by a goal configuration. Given that the goal configuration is the primary parameter, these methods are typically either fixed strategies (such as splining to a goal point [365], or between keyframes [6]) or have a very small number of adjustable parameters, such as a PID controller or motion planner.

6.2. Reinforcement Learning

For any given policy representation, reinforcement learning (RL) [371] can be used to learn policy parameters for skill controllers. In robotics domains, tasks addressed with RL are usually *episodic*, with a fixed number of time steps or a set of terminal states that end the episode (e.g. reaching a particular object configuration), but occasionally may be *continuing* tasks with infinite horizons (e.g. placing a continual stream of objects into bins as fast as possible).

There are many challenges in applying RL to robotics. Due to the time it takes to collect data on physical robots, the tradeoff between exploration and exploitation becomes significantly more important, compared to problems for which fast, accurate simulators exist. Furthermore, few problems in robotics can be strictly characterized as MDPs, but instead exhibit partial observability [304] and nonstationarity [285]. Since many tasks in robotics are multi-objective (e.g. pick up the mug, use as little energy as possible, and don't collide with anything), it can be difficult to define appropriate reward functions that elicit the desired behavior [139]. Due to the episodic nature of most robotics tasks, rewards tend to be sparse and difficult to learn from. Robotics problems also often have high-dimensional continuous state features, as well as multi-dimensional continuous actions, making policy learning challenging.

RL algorithms vary along three primary axes: (1) model-based or model-free, (2) whether or not they compute a value function, and in what manner they utilize it, and (3) on-policy or off-policy.

Model-Based RL: Accurate models of transition dynamics are rarely available *a priori* and it is often difficult to learn transition models from data. Nonetheless, there have been notable model-based successes in robotic manipulation in which an approximate model is learned from data [83, 239, 235, 227, 110, 331]. In these examples, the model is typically used to guide exploration and policy search, leading to greatly improved data efficiency. More generally, the primary benefits of model-based RL in robotics

are that (1) in some domains, an approximate model is simpler to learn than the optimal policy (since supervised learning is generally easier than RL) (2) models allow for re-planning / re-learning on-the-fly if the task changes, and (3) models allow for certain types of data collection that are simply not possible in the real world (such as resetting the world to an exact state and trying different actions to observe their outcome). The primary disadvantage of model-based methods, aside from the difficulty of obtaining or learning models, is that incorrect models typically add bias into learning. However, some methods mitigate this by directly reasoning about uncertainty [83], whereas methods such as doubly robust off-policy evaluation [179] use a model as a control variate for variance reduction, sidestepping the bias issue. More details on model learning can be found in Section 5.

Model-Free RL: Model-free methods learn policies entirely from experiences of the robot in the real world. When the dynamics of the environment are complex, as they often are in contact-rich manipulation tasks, it can be significantly easier to learn a good policy than a model with similar performance. Model-free methods can learn how to implicitly take advantage of complex dynamics without actually modeling them–for example, finding a motion that can pour water successfully without understanding fluid dynamics. The downside of the model-free approaches is that they cannot easily adapt to new goals without additional experience in the world, unlike model-based approaches. Furthermore, they typically require a large (sometimes prohibitive) amount of experience to learn. For this reason, some recent approaches have sought to combine the benefits of model-based and model-free methods [130, 56, 305, 108, 104].

Value Function Methods: These methods aim to learn the value of states (or more commonly, state-action pairs)—the expected return when following a particular policy, starting from that state(-action pair). Value functions methods are known to be low variance, highly sample efficient, and in discrete domains (or continuous domains with linear function approximation), many such methods can be proven to converge to globally optimal solutions. However, value function methods are typically brittle in the face of noise or poor or underpowered state features, and do not scale well to high-dimensional state spaces, all of which are common in robot manipulation tasks. Furthermore, they are not compatible with continuous actions, severely limiting their application to robotics, except where discretization is acceptable. Actions are often discretized at higher levels of abstraction when they represent the execution of entire skills, rather than primitive actions [219]. Finally, it is difficult to build in useful structure and task knowledge when using value function methods, since the policy is represented implicitly. Nonetheless, value function-based methods such as Deep Q-Networks [264] and extensions thereof have been applied successfully to robotics tasks [435, 186].

Policy Search Methods: Policy search methods parameterize a policy directly and search for parameters that perform well, rather than deriving a policy implicitly from a value function. Policy search methods have become popular in robotics, as they are robust to noise and poor features and naturally handle continuous actions, (the direct parameterization of the policy eliminates the need to perform a maximization over values of actions). They also often scale well to high-dimensional state spaces since the difficulty of policy search is more directly related to the complexity of a good (or optimal) policy, rather than the size of the underlying state space.

Pure policy search approaches eschew learning a value function entirely. These are sometimes referred to as "actor-only" methods, since there is a direct parameterization of the policy, rather than a "critic" value function. Actor-only approaches include the gradient-based method REINFORCE [416], for use when the policy is differentiable with respect to its parameters. However, REINFORCE tends to suffer from high variance due to noisy sample-based estimates of the policy gradient (though a variance-reducing baseline can be used to partially mitigate this) and is only locally optimal. Actor-only policy search also includes gradient-free optimization methods [81, 257, 142, 80, 384], which are usable even when the policy is non-differentiable, and in some cases (e.g. genetic algorithms) are globally optimal, given sufficient search time. Unsurprisingly, these advantages often come at the cost of reduced sample efficiency compared to gradient-based methods.

In contrast to actor-only approaches, actor-critic policy search methods [372, 295, 296, 246, 265, 339, 340, 413, 420, 138] use both a value function (the critic) and a directly parameterized policy (the actor), which often share parameters. These methods typically have most of the advantages of actor-only methods (the ability to handle continuous actions, robustness to noise, scaling, etc), but use a bootstrapped value function to reduce the variance of gradient estimates, thereby gaining some of the sample efficiency and low-variance of critic-only methods. The unique advantages of actor-critic methods have made them state of the art in many robotic manipulation tasks, as well as in the larger field of deep reinforcement learning.

On-Policy vs Off-Policy Learning: When using RL to improve a policy, on-policy algorithms are restricted to using data collected from executions of that specific policy (e.g. SARSA [371]), whereas off-policy algorithms are able to use data gathered by any arbitrary policy in the same environment (e.g. Q-learning [414]). This distinction has several consequences which are notable in robotic domains for

which data collection has significant costs. On-policy algorithms are not able to re-use historical data during the learning process—every time the behavior policy is updated, all previously collected data becomes off-policy. It is also often desirable to use collected data to simultaneously learn policies for multiple skills, rather than only a single policy; such learning is impossible in an on-policy setting. However, despite these advantages, off-policy learning has the significant downside of being known to diverge under some conditions when used with function approximation, even when it is linear [371]. By contrast, on-policy methods are known to converge under mild assumptions with linear function approximation [371].

Exploration Strategies: A significant, but often overlooked, part of policy learning is the exploration strategy employed by the agent. This design decision can have enormous impact on the speed of learning, based on the order in which various policies are explored. In physical robotic manipulation tasks, this is a particularly important choice, as data is much more difficult to collect than in simulated domains. Furthermore, in the robotics setting, there are concerns related to safety and possible damage to the environment during exploration, as will be further discussed in Section 6.5.

The most common exploration strategies involve adding some form of noise to action selection, whether it be in the form of direct perturbation of policy parameters [384], Gaussian noise added to continuous actions [246], or epsilon-greedy or softmax action selection in discrete action spaces [264]. Unsurprisingly, random exploration often fails to efficiently explore policy space, since it is typically confined to a local region near the current policy and may evaluate many similar, redundant policies.

To address these shortcomings, exploration strategies based on the psychological concept of *intrinsic motivation* [58, 283] have used metrics such as novelty [167, 27, 144, 98, 50], uncertainty [259], and empowerment [266] to diversify exploration in a more effective way. However, these methods are heuristic-based and may still lead to poor performance. In fact, many essentially aim to use intrinsic motivation to uniformly explore the state space, ignoring structure that may provide clues about the relevance of different parts of the state space to the problem at hand. By contrast, other approaches have taken advantage of structure specific to robotic manipulation by directing exploration to discover reusable object affordances [268], or learning exploration strategies that exploit the distribution of problems the agent might face via metalearning [425]. Finally, as discussed in the next subsection, the difficulty of exploration in RL is sometimes overcome by leveraging demonstrations of good behavior, rather than learning from scratch.

6.3. Imitation Learning

In contrast to reinforcement learning, which learns from a robots experiences in the world (or a model of it), imitation learning [325, 14] aims to learn about tasks from demonstration trajectories. This can be thought of as a form of programming, but one in which the user simply shows the robot what to do instead of writing code to describe the desired behavior. Learning from demonstration data has been extensively studied in several different settings, because it can enable the robot to leverage the existing task expertise of (potentially non-expert) humans to (1) bypass time-consuming exploration that would be required in a reinforcement learning setting, (2) communicate user preferences for how a task ought to be done, and (3) describe concepts, such as a good tennis swing, that may be difficult to specify formally or programmatically. It is worth noting that imitation learning and reinforcement learning are not mutually exclusive; in fact, it is common for imitation learning to be followed by reinforcement learning for policy improvement [198, 378].

Demonstrations in imitation learning are typically represented as trajectories of states or state-action pairs. There are several mechanisms by which a robot may acquire such demonstration trajectories, including teleoperation, shadowing, kinesthetic teaching, motion capture, which are discussed in greater detail in the survey by Argall et. al [14]. More recently, keyframe demonstrations [6], virtual reality demonstrations [436], and video demonstrations in the so-called learning from observation setting [251] have also become more commonly utilized.

Behavioral Cloning: The simplest way to use demonstration data to learn a motor skill is to use it as supervised training data to learn the robot's policy. This is commonly called *behavioral cloning*. Recall that a deterministic policy π is a mapping from states to actions: $\pi : S \rightarrow A$, whereas a stochastic policy maps state-action pairs to probabilities (which sum to 1 at each state when marginalizing over actions): $\pi : S \times A \rightarrow \mathcal{R}$. The demonstration provides a set of state-action pairs (s_i, a_i), that can be used as training data for a supervised learning algorithm to learn policy parameters that should, ideally, be able to reproduce the demonstrated behavior in novel scenarios.

Behavioral cloning is often used as a stand-alone learning method [316, 19, 390, 288, 55, 6, 96, 151, 327], as well as a way to provide a better starting point for reinforcement learning [289], though this additionally requires the specification of a reward function for RL to optimize. Other recent work has focused on expanding the purview of behavioral cloning by unifying imitation learning and planning via probabilistic inference [308], utilizing additional modalities such as haptic input [211], learning to recognize and recover from errors when imitating [290, 289],

and scaling behavioral cloning to complex, multi-step tasks such as IKEA furniture assembly [281].

Reward Inference: Rather than learning a policy directly from demonstration data, an alternative approach is to attempt to infer the underlying reward function that the demonstrator was trying to optimize. This approach aims to extract the intent of the motion, rather than the low-level details of the motion itself. This approach is typically called inverse reinforcement learning (IRL) [275], apprenticeship learning [1, 37], or inverse optimal control [269, 103, 101]. The inferred reward function can then be optimized via reinforcement learning to learn a policy for the task.

The IRL paradigm has several advantages. First, if the reward function is a function of the objects or features in the world and not the agent's kinematics, then it can be naturally ported from human to robot (or between different robots) without encountering the correspondence problem. In addition, reward functions are often sparse, thereby providing a natural means of generalizing from a small amount of training data, even in very large state spaces. In addition, the human's behavior may encode a great deal of background information about the task—for example, that an open can of soda should be kept upright when it is moved—that are easy to encode in the reward function but more complex to encode in a policy, and which can be reused in later contexts. Unfortunately, IRL also presents several difficulties. Most notably, the IRL problem is fundamentally ill-posed—infinitely many reward functions exist that result in the same optimal policy [275]. Thus, the differentiation between many IRL algorithms lies in the metrics that they use to disambiguate or show preference for certain reward functions [1, 2, 37, 3, 103, 101].

Maximum Entropy IRL [439] addresses the problems of demonstrator suboptimality and ill-posedness by leveraging a probabilistic framework and the the principle of maximum entropy to disambiguate possible reward functions. Specifically, they develop an algorithm that assigns equal probability to all trajectories that would receive equal return under a given reward function and then use this distribution to take gradient steps toward reward functions that better match the feature counts of the demonstrations [440], while avoiding having any additional preferences other than those indicated by the data. Rather than generating a point estimate of a reward function, which forces an algorithm to face the ill-posedness of IRL head on, Bayesian IRL [307] instead uses Markov Chain Monte Carlo to sample from the distribution of all possible reward functions, given the demonstrations. Finally, in the more restricted case of linearly-solvable MDPs, the IRL problem is well-posed, avoiding these problems [97].

All of the IRL algorithms mentioned so far rely on reward functions specified as a linear combination of features. While this does not restrict the expressivity of reward functions in practice (more complex features can always be provided), it burdens the designer of the system to ensure that features can be learned from in a linear manner. By contrast, Gaussian Process and nonparametric IRL [238, 103] and various neural network-based methods [158, 111, 421, 118] aim to partly relieve this burden by searching for reward functions that are a nonlinear function of state features. However, such flexibility in representation requires careful regularization to avoid overfitting [111].

Many of the aforementioned methods have an MDP solver in the inner loop of the algorithm. Computational costs aside, this is especially problematic for robotics settings in which a model is not available and experience is expensive to collect. Some recent IRL methods that have been shown to work in real robotic domains sidestep this obstacle by alternating reward optimization and policy optimization steps [111] or framing IRL as a more direct policy search problem that performs feature matching [93, 158]. If available, ranked demonstrations can be used to get rid of the need for inference-time policy optimization or MDP solving entirely, by converting the IRL problem to a purely supervised problem; furthermore, this approach allows the robot to potentially outperform the demonstrator [43]. Alternately, active learning techniques have been used to reduce the computational complexity of IRL [46, 72, 252], as well as strategies that make non-I.I.D. assumptions about the informativeness of the demonstrator [45, 187]. Finally, outside of an imitation learning framework, goals for the robot are sometimes specified via natural language commands that must be interpreted in the context of the scene [380].

Learning from Observation: A relatively new area of inquiry aims to learn from demonstrations, even when no action labels are available and the state is not exactly known. For example, a robot may visually observe a human performing a task, but only have access to raw pixel data and not the true underlying state of the world, nor the actions that the human took. This problem is referred to Learning from Observation (LfO), and several recent approaches have addressed problems including unsupervised human-robot correspondence learning [343], context translation [251], adversarial behavioral cloning [390], and IRL from unsegmented multi-step video demonstrations [126, 430]. In an extreme version of the LfO setting, the agents is expected to infer an objective from single-frame goal-state images, rather than a full trajectory of observations [434, 423].

Corrective Interactions: Rather than learning from full demonstrations in batch, it is often advantageous to solicit (potentially partial) corrective demonstrations or other

forms of feedback over time. For example, a human could intervene in a pouring task and adapt the angle of the cup and the robot's hand mid-pour. This provides a natural mechanism to collect data in situations where it is most needed—for example, situations in which mistakes are being made, or where the robot is highly unsure of what to do. Some approaches actively ask users for additional (partial) demonstrations in areas of the state space in which confidence is low [59] or risk is high [46], while others rely on a human user to identify when a mistake has been made [281]. Higher level information can also be used to make more robust corrections, such as grounded predicate-based annotations of corrections [270] and action suggestions in a high-level finite state machine [160]. The robot can also actively solicit assistance when needed, for example, via natural language [196]. Finally, rather than using corrective demonstrations, the TAMER framework utilizes real-time numeric human feedback about the robot's performance to correct and shape behavior [197].

6.4. Skill Transfer

Given the high sample complexity of learning in complex robotics tasks, skills learned in one task are often transferred to other tasks via a variety of mechanisms, thereby increasing the efficiency of learning.

Direct Skill Re-use: One of the simplest ways to transfer a skill policy is to directly re-use it in a new task related to the one it was learned on. Typically, some amount of adaptation is required to achieve good performance in the new task. One simple way to perform such refinement is to initialize a new skill with an existing skill's policy parameters and adapt them for the new task via reinforcement learning [288]. However, a naive realization of this approach only works when the original task and the new task have identical state representations. When this is not the case, transfer can still occur via a subset of shared state features that retain semantics across tasks. This is sometimes called an *agent space* [205], since these are typically agent-centric, generic features (e.g laser scanner readings), rather than problem-specific features, or a *deictic representation* [302]. Since an agent space only covers some subset of the features in a problem, transfer occurs via initialization of a value function, rather than policy parameters.

More generally, it is often useful to find a state *abstraction*—a minimal subset of state features required to perform some skill. Abstractions facilitate transfer by explicitly ignoring parts of the state that are irrelevant for a particular skill, which could otherwise serve as distractors (e.g. irrelevant objects), as well as allowing transfer to state spaces of different sizes (as also seen in agent spaces, which are a type of abstraction). In some state spaces, such as the visual domain, the state space is not factored in a

manner that makes abstraction easy (e.g. the texture of an object is not a separate state feature, but distributed across many pixels). One popular way of forcing a deep neural network to abstract away complex variables such as texture and color is *domain randomization* [389, 38], which is discussed in greater detail later in this section.

Parameterized Skills: In certain task families, only some aspects of the task context change, while all other task semantics remain the same or are irrelevant. For example, it may be desirable to transfer a policy that can hit one goal location on a dartboard, in order to hit a different goal location; similarly, transfer learning could acquire a policy for completing a pendulum swing-up task with different pendulum lengths and masses. In these restricted cases, specialized *parameterized skills* can be learned that facilitate transfer via mechanisms that modulate the policy based on the aspect of the task context parameter that is changing [53, 102].

Dynamic Movement Primitives [326, 288] use a simple spring-mass-damper system to smoothly adjust to new initial and goal locations. Another approach uses manifold learning to smoothly modulate policy parameters based on a variable task parameter [73]. Contextual policy search uses a hierarchical, two-level policy for low-level control and generalization across contexts, respectively [229]. Universal value function approximators [328, 13] track a value function for all ⟨state, action, goal⟩ triplets, rather than only ⟨state, action⟩ pairs, allowing policy similarities across nearby goals to be leveraged explicitly.

Metalearning: Rather than re-using a skill directly for initialization, metalearning approaches aim to "learn to learn"—in other words, learn something about a distribution of tasks that allows for more efficient learning on any particular task from that distribution in the future. Thus, metalearning facilitates transfer within a task family by performing learning across samples from a task distribution at training time, rather than performing transfer sequentially and on-line as problems are encountered, as in the direct re-use case.

Model Agnostic Metalearning (MAML) [112] searches for a set of policy parameters that can be adapted quickly for particular tasks drawn from some distribution. Reptile [277] simplifies MAML by replacing a complex optimization with an approximate approach that only requires standard stochastic gradient descent to be performed on each sampled task individually. Another related metalearning approach learns an attention-based strategy that allows the robot to imitate novel manipulation tasks, such as block-stacking, from a single demonstration [96].

Other forms of metalearning have focused on reward functions rather than policies. While many problems in robotic

manipulation have simple sparse reward formulations (e.g. +1 when peg is in hole, -1 otherwise), *potential-based* shaping rewards [274] can be added to any reward function to encourage faster learning (across some distribution of problems) without changing the optimal policy of the MDPs. More generally, modified reward functions can help to overcome multiple forms of *agent boundedness* [352] and can be found via gradient descent [351], genetic programming [280], or other evolutionary methods [161].

Some metalearning approaches have been developed to directly modify the representation of the policy itself, or other parts of the learning algorithm, rather than only the settings of policy or reward parameters. This includes methods that evolve the structures of neural networks [355], or that learn skill embeddings [149], structured exploration strategies [136], transfer feature relevances between families [223],reusable network modules [8], or that co-learn a differentiable model and trajectory optimizer [353].

Domain Adaptation: In contrast to parameterized skills, some task families retain all of their high-level semantics across instances, differing only in lower-level details. In these cases, *domain adaptation* techniques are commonly used to bridge the so-called domain gap between two (or more) domains. For example, in the "sim2real" problem, when switching from a simulated task to a physical robot, the low-level statistics of the visual scene and physics may change, while the high-level steps and goals of the task stay fixed. Grounded action transformations address the physics domain gap by iteratively modifying agent-simulator interactions to better match real-world data, even when the simulator is a fixed black box [140]. Domain randomization is a data synthesis technique used to address visual domain gaps by randomizing environmental attributes such as texture, color, and dynamics in simulation, as to force the system to be invariant to statistical changes in the properties [389, 38, 294]. A different type of visual shift can result from changes in viewpoint, often encountered in the setting of learning from unstructured videos. Approaches to this problem have included unsupervised learning of correspondences between first-person and third-person task demonstrations [354], as well as imitation-from-observation approaches that work from multi-viewpoint data [251]. Other approaches attempt to learn disentangled representations that lead naturally to robust polices that are tolerant of domain shift [155].

In some tasks, the state and action space may change in a manner that requires an analogy to be made between the domains, as in the case of transferring a manipulation policy between two robot arms with different kinematic structures by finding a shared latent space between the two policy representations [135]. Time Contrastive Networks take a self-supervised metric-learning approach to a similar problem, learning representations of behaviors that are invariant to various visual factors, enabling robot imitation of human behaviors without an explicit correspondence [343]. Other methods perform a more direct mapping between states and actions [377], but assume the there exists a complete mapping, while the aforementioned latent space approach is able to discover partial mappings.

Sequential Transfer and Curriculum Learning: While the aforementioned transfer paradigms primarily consider instances of transfer individually, it is sometimes advantageous to view multiple instances transfer as a sequential learning problem. For example, Progressive Neural Networks [318] use a neural network to learn an initial task, transferring this knowledge to subsequent tasks by freezing the learned weights, adding a new column of layers for the new task, and making lateral layer-wise connections to the previous task's neurons. Sequential transfer learning is also a useful paradigm for mastering tasks that are too difficult to learn from scratch. Rather than attack the final task directly, *curriculum learning* presents a sequence of tasks of increasing difficulty to the agent, which provides a smoother gradient for learning and can make positive rewards significantly less sparse in an RL setting. Such methods often utilize curricula provided by human designers [28, 322, 300], though several recent methods seek to automatically generate a curriculum [273, 374, 115].

Scheduled Auxiliary Control [313] learns to sequentially choose from a set of pre-defined auxiliary tasks in a manner that encourages efficient exploration and learning of the target task. Guided Policy Search [239] first learns simple time-dependent controllers for individual task instances, which are then used to generate data to train a more complex neural network. Universal Value Function Approximators (UVFAs) [328] can learn and transfer knowledge more efficiently in curriculum-like settings by estimating the value of a state and action *conditioned upon the current goal*, rather than for a single fixed goal. Hindsight Experience Replay (HER) provides additional goal settings for UVFAs by simply executing actions, and in hindsight imagining that every state that the agent ends up in was actually a goal [13].

Reverse curriculum learning [115] begins at the goal and works backwards, trying to learn to get to the goal successfully from more and more distant states. This stands in contrast to standard curriculum learning that typically starts in some distant state, slowly moving the goal of the subtasks closer and closer to the true goal. This idea is also related to that of skill chaining [206, 203], which will be discussed in further detail in Section 8.1.

6.5. Safety and Performance Guarantees

Whether a policy is learned directly for a specific task or transferred from a previous task, an important but understudied question is "How well will the policy perform across the distribution of situations that it will face?". This is an especially important question in robotic manipulation, in which many future applications will require behaviors that are safe and correct with high confidence: robots that operate alongside humans in homes and workplaces must not cause injuries, destroy property, or damage themselves; safety-critical tasks such as surgery [398] and nuclear waste disposal [194] must be completed with a high degree of reliability; robots that work with populations that rely on them, such as the disabled or elderly [124], must be dependable. While the term "safety" has taken on many different meanings in the literature, in this section, we take a broad view of safe learning to include all methods that aim to bound a performance metric with high confidence.

Performance Metrics: Most performance metrics for robotic manipulation tasks can be represented as cumulative reward under some reward function—for example, the preferences of completing a task accurately, finishing within a certain amount of time, or never (or with low probability) visiting certain states can all be described with particular reward functions and acceptable thresholds on cumulative reward, or *return*. Thus, for simplicity, we will discuss the problem of bounding performance as being equivalent to that of lower-bounding some function of return. However, it is worth noting that certain preferences, such as some temporal logic statements, cannot be expressed as a reward function without augmenting the state space; some cannot be expressed by any reward function at all [250].

The principle axis of variation regarding performance metrics is whether the *expected* return of a policy is being bounded, or whether a risk-aware function of return is used—for example, a robust worst-case analysis [122], a PAC-style bound [360], or bounded value-at-risk [66, 44]. Risk-aware metrics are generally more appropriate for safety-critical tasks in which the robot could cause damage or injury in a single trial that goes poorly, whereas expected performance is often used in scenarios in which long-term performance (e.g. percentage of correctly manufactured items) matters more than the outcome of any single trial.

Finally, one important distinction to make is *when* a policy must obey performance bounds. Some robotics tasks, such as learning to manipulate a fragile object, demand performance guarantees during the exploration and learning phase, rather than only after deployment of the final policy. For example, in one recent work, demonstrations were used to constrain exploration in safety-critical manipulation tasks [383]. By contrast, other tasks may lend themselves to safe practice, such that only the final policy needs to be accompanied by guarantees. For example, a legged robot may learn a walking gait while attached to a support rig so that it cannot catastrophically fall while learning.

Classes of Guarantees and Bounding Methods: Given that standard policy learning in robotics is a challenging open problem, in part due to limited real-world data collection abilities, it is not surprising that safe learning is even more difficult. Safe learning methods are typically significantly more data-hungry and/or require more accurate models than standard learning methods, as they need to provide guarantees about the quality of the policy. For this reason, very few safe learning methods (e.g. from the reinforcement learning community) have been used in robotics applications. This is a significant gap in the current literature and an opportunity for future work that can provide strong performance guarantees in low-data and poor-model robotics settings.

Safe learning methods fall roughly into two categories: Exact methods and probabilistic high-confidence bounds. Formal verification-based approaches are exact methods that use a known model, along with a safety specification (e.g. a finite-state automata) to guarantee (or bound the probability, in the stochastic case) that following a policy will not lead to an unsafe outcome [116, 106, 61, 57, 10]. However, some research has focused on control synthesis that obeys linear temporal logic constraints without access to a model [319]. By contrast, probabilistically safe approaches aim to provide lower-bounds on expected return, rather than utilizing logical specifications of safety [387, 386, 179, 141, 385]. While probabilistic methods appear to be promising for robotics applications (since they do not require a model), to the best of our knowledge, they have not been used in real robotics problems, potentially due to their high sample complexity.

Thus far, we have only considered performance guarantees in a standard MDP setting, in which either (1) the reward function is known, or (2) samples from the reward function are available. However, this description of the problem does not cover a common scenario that occurs when learning from demonstrations—having access to states and actions, but not rewards. In the inverse reinforcement learning setting, several approaches have examined how to bound the policy loss between the optimal policy under the inferred reward function and the optimal policy under the ground-truth reward function of the demonstrator, even though it is unknown [1, 44, 46].

7. Characterizing Skills by Preconditions and Effects

Executing a manipulation skills alters the state of the robot's environment; if the robot is to use its skills to achieve a specific goal, it requires a model of those outcomes. This model is termed a *postcondition*, and describes the regions in the state space that the robot will find itself in after skill execution. The robot must also model the circumstances under which the skill can be executed—termed its *preconditions*. Knowledge of these two models for each skill can be used to determine whether, for example, a sequence of skills can be executed from a particular state [208]. Pre- and postconditions are used to sequence skills (or actions) for task planning. The planner searches through sequences of actions with the constraint that the postconditions of each skill must fulfill the preconditions of the next skill.

7.1. Pre- and Postconditions as Propositions and Predicates

The pre- and postconditions of manipulation skills are typically represented abstractly as either *propositions* or *predicates* [224, 225, 208, 24, 344, 394] that are either true or false in any specific state. For example, the robot may represent the outcome of a particular navigation skill using the boolean proposition `AtTableB`. It could also use a predicate representation, `At(TableB)`, which supports more compact and efficient representations, and allows for easier generalization across objects. We therefore use predicate-based symbols for the remainder of this section although most of the explanation also applies to proposition symbols.

The *grounding* or anchoring of a predicate refers to the mapping between the (often continuous) low-level state and context and the predicate's truth value; it defines the meaning of the symbol [208, 70]. To avoid confusion, we will not use the term grounding for assigning objects to predicates as is done in the planning literature [317].

Classifier Representation: The grounding of a predicate can be modelled as a binary classifier [208, 216, 204]. The classifier represents the mapping from the state and context to either true or false. If a predicate is defined for a subset of objects, then the state and context features of those objects are used as the input to the predicate classifier [216]. For example, the predicate `Grasped(KnifeA, RHand)` would consider the state and context of the knife and hand to determine if the knife was being grasped by the the hand. The robot may learn a probabilistic classifier to determine the likelihood that the predicate is true given the current state and context. In this manner, the robot may handle situations where the predicate is only sometimes valid, or when the classifier has been learned and the robot is uncertain of its true value in states it has not encountered before [208].

Distribution Representation: Alternatively, predicates can be modelled as probability distributions over the state space [208, 86]. The distributions are defined in the state and context space. For a non-probabilistic approach, the distribution defines the set of states and contexts where the distribution is true, and the predicate is false otherwise. For a probabilistic approach, the distribution defines a probability density over the states and contexts given that the predicate is true [208, 86]. The distribution may be defined only for the objects assigned to the predicate. This distribution is useful for sampling states and contexts in which the predicate is true.

Modularity and Transfer: One precondition proposition and one postcondition proposition for each skill are sufficient for skill sequencing. These predicates can be monolithic representations that define the sets of states and contexts for that specific skill. However, by decomposing the conditions into modular predicates, the robot can share knowledge between different skills and tasks. These predicates can often be defined for subsets of objects, e.g., `Full(mug, water)` and `Grasped(mug, hand)`. The predicates often define labels for individual objects, e.g., `Container(obj1)`, and the relationships between the objects, e.g., `Above(obj1,obj2)`. Modular predicates will often capture the contact-based manipulation modes between pairs of objects, e.g., `Grasped(Obj1,Hand)` or `On(Obj2,Obj3)`. Previous works have explored methods for learning specific predicates or discovering suitable sets of predicates [225, 268, 208, 157].

Learning the preconditions and postconditions is generally easier when reusing predicates from previous tasks rather than learning from scratch. However, some discrepancies may exist between tasks and thus require additional learning [395]. For example, a robot may require a specific type of grasp for performing a task. The predicate `Grasped(obj3,hand)` is thus not sufficient. The robot should instead learn a general predicate for transferring knowledge between tasks, and a task-specific predicate for incorporating additional constraints [88, 381, 131, 253, 35, 157]. The latter predicate is generally easier to learn given the former predicate. For example, a robot may learn to identify stable grasps for a wide range of objects, and subsequently learn to identify a subset of grasps for specific objects or to grasp handles of cooking utensils without relearning to grasp from scratch [88, 131, 381].

7.2. Learning Pre- and Postcondition Groundings

The robot can train the pre- and postcondition predicate classifiers using samples of states and contexts where the conditions were known to be true or false. The ground truth labels could be provided by a human supervisor, although this approach would limit the autonomy of the robot and may require substantial expertise from the user. In some cases a human can provide data for *desired* post- and preconditions—what the skill should achieve, from which conditions—but the actual conditions will ultimately depend on the robot's capabilities.

Instead of relying on manual labeling, the robot can learn the condition labels from experience. The labels for the preconditions can be learned given a fixed postcondition — all states from which a skill execution leads to a state satisfying the postcondition are positive examples, and all other states are negative examples. The robot should use a probabilistic classifier to capture the stochasticity of the transitions. The robot can thus obtain the precondition labels by executing the skill from different states and observing the resulting outcomes.

The postconditions are less trivial to define. A human may predefine a desired postcondition, but to achieve autonomy the robot must *discover* different postconditions on its own. Distinct postconditions can be learned by clustering the outcomes of the skill from different initial states and contexts [94, 396, 394]. Manipulation skills often have distinct effects, e.g., a box remained upright or toppled over, which can be extracted through clustering or by detecting salient events. Each distinct postcondition cluster will then be associated with its own preconditions. This approach is often employed in developmental learning to discover specific useful skills from more general skill policies, e.g., grasping and pushing from reaching [284, 182].

A more goal-oriented approach to specifying postconditions is to construct skills where the postcondition is either the goal of the task or another skill's precondition [206, 202]. This approach directly learns pre- and postconditions for constructing skills that can be sequentially executed, and may avoid learning conditions that are irrelevant to the robot's task set, but it introduces dependencies between the robot's skills.

Finally, pre- and postcondition predicates can be grounded in sensory data with the assistance of natural language, for example, as part of an interactive dialogue between a human and a robot [388]. Conversely, knowledge of pre- and postconditions of skills can help to ground natural language commands to those skills, especially when the language description is incomplete [263]. For example, the command "Stir the soup" may imply picking up a spoon first, which could be determined via the precondition to a stirring skill.

7.3. Skill Monitoring and Outcome Detection

Most skills will have distinct pre- and postconditions with some of the predicates changing as a result of the skill execution, e.g., executing a grasping skill on a book should result in `Grasped(Book,RHand)=True` once the book has been grasped [75, 121]. The predicates may also change due to errors in the skill execution. For example, when executing a placing skill, the predicate values `On(Book, Table)=False` or `InCollision(Book, Obstacle)=True` would correspond to errors. To perform manipulation tasks robustly, the robot must monitor its skill executions and determine if and when it has achieved the intended outcome or whether an error has occurred.

Learning Goal and Error Classifiers: Goals and errors in skill executions can be modeled as distinct predicates with values that are switched for the postconditions. Detecting goals and errors can thus be modelled as a classification problem [26, 314]. Rather than using only the current state, the robot can incorporate action and sensor information from the entire skill execution [26, 254, 314], although it is often better to stop a skill early when an error occurs [291, 365, 366]. Transient events, such as vibrations from mode transitions or incipient slip can then be used to better detect the predicate switches [287, 403, 365]. The robot can use a variety of sensor modalities, including vision and audio, to detect the predicate switches, e.g., a robot can use tactile sensing to determine if a grasp attempt succeeded `Grasped(obj)=True` [52, 75, 254, 76]. A robot can also learn optimal locations for placing a camera, or other sensor, to reliably verify if a desired postcondition has been fulfilled [323].

Detecting Deviations from Nominal Sensory Values: Stereotypical executions of a manipulation skill will usually result in similar sensations during the execution. Larger deviations from these nominal values often correspond to errors [427]. Hidden Markov models can be used to track the successful progress of a skill's execution based on sensory signals [287, 234, 162, 25]. The robot may also learn the nominal sensory signals as a regression problem [289, 189, 366]. Significant deviations from the expected sensory values would then trigger the stopping of the current skill. An error may also trigger a corresponding recovery action [76, 402, 428]. Unlike the outcome classifiers from the previous section, these models are trained using only data from successful trials.

Verifying Predicates: The pre- and postconditions of skills often change the values of predicates corresponding to modes and constraints between objects. For example, a placing skill can make `On(DishC,DishB)=True` and an unlocking skill can make `Locked(DoorA)=False`. The robot can verify these swiches in the predicate values

using *interactive perception*. In some cases the predicate can be verified by directly performing the next skill in the sequence, e.g., attempting to lift an object after it has been grasped [300, 186]. For contact constraints, the robot can often perform small perturbations to verify the predicate's final value [82, 410]. In both examples, the robot can obtain a better estimate of the predicate by performing the additional skill and observing the effect. However, the additional skills can sometimes also change the predicates, and hence some care needs to be taken when verfiying predicates.

7.4. Predicates and Skill Synthesis

Skills will often have additional high-level arguments that define how to execute the skill [397]. For example, a grasping skill may take in a grasping pose, or a scrubbing skill may take in a desired force [32]. In this manner, a higher-level policy may adapt the skill to its specific needs.

Representing and Synthesizing Skill Parameters: Similar to the predicate representations, these policy parameter arguments can be modeled as additional input features for the precondition classifier or as distributions over valid and invalid argument values [86, 180]. Many of these arguments can even be thought of as virtual or desired object states [180]. For example, one could sample potential hand positions for grasping objects and model each one as a symbol for specifying the grasping action [121, 32]. To obtain valid grasp frames, the robot can learn a classifier to determine if sampled grasp frames will lead to successful grasps [324, 276, 154, 300], or sample from a learned probability density over successful grasps or cached grasps from similar objects [86, 41, 125]. The process of learning the pre- and post- conditions with additional policy arguments is thus similar to learning the preconditions without the arguments.

Once the robot has learned the pre- and postconditions over the arguments, it can use these to select skill parameters for new situations. This process usually involves sampling different parameter values and evaluating them in terms of the pre- and postconditions. The robot could use sampling and optimization methods to select argument values with high likelihoods of successful skill executions. As these parameters are often lower dimensional than full skills, active learning and multi-armed bandit methods can be used to select suitable argument values [267, 255, 221]. When the positive distribution is learned directly, the robot can sample from the distribution and then evaluate if it is consistent with other predicates, e.g., not in collision [68]. The arguments are usually selected in order to achieve certain predicates in the postconditions. One can therefore think of the argument selection process as *predicate synthesis* [11].

Preconditions and Affordances: Affordances are an im-

portant concept in manipulation and a considerable amount of research has explored learning affordances for robots [320, 262, 172]. The affordances of an object are the actions or manipulations that the object affords an agent [123]. An object that can be used to perform an action is thus said to *afford* that action to the agent. For example, a ball affords bouncing, grasping, and rolling. Affordances are thus closely related to the preconditions of skills. As affordances connect objects to skills [268], affordance representations often include skill arguments that define how the skill would be executed[397, 217].

The exact usage of the term *affordances* tends to vary across research papers [172]. In most cases, affordances can be seen as a form of partial preconditions. The affordances often correspond to specific predicates that the robot learns in the same manner as the preconditions. The other components of the precondition are usually constant or at least all valid, such that the success or failure of the skill only depends on the component being learned. As partial preconditions, some affordances are more specific than others, e.g., balls can be rolled versus balls on a plane within reach of the robot can be rolled. Ultimately, the full preconditions need to be fulfilled to perform the skill. However, the partial preconditions of affordances provide modularity and can help the robot to search for suitable objects for performing tasks.

8. Learning Compositional and Hierarchical Task Structures

In the previous sections, we have focused on learning models of individual objects, or on learning to perform or characterize individual motor skills. However, manipulation tasks often have a substantial *modular* structure that can be exploited to improve performance across a family of tasks. Therefore, some research has attempted to decompose the solution to a manipulation task into *component skills*. Decomposing tasks this way has several advantages. Individual component skills can be learned more efficiently because each skill is shorter-horizon, resulting in a substantially easier learning problem and aiding exploration. Each skill can use its own internal skill-specific abstraction [90, 201, 400, 69, 178] that allows it to focus on only relevant objects and state features, decomposing a problem that may be high-dimensional if treated monolithically into one that is a sequence of low-dimensional subtasks. A skill's recurrence in different settings results in more opportunities to obtain relevant data, often offering the opportunity to generalize; conversely, reusing skills in multiple problem settings can avoid the need to relearn elements of the problem from scratch each time, resulting in faster per-task learning. Finally, these component skills create a *hierarchical structure* that offers the opportunity to solve manipula-

tion tasks using higher-level states and actions—resulting in an easier learning problem—than those in which the task was originally defined.

8.1. The Form of a Motor Skill

The core of hierarchical structure in manipulation learning tasks is identifying the component skills from which a solution can likely be assembled.

Recall that skills are often modeled as options, each of which is described by a tuple $o = (I_o, \beta_o, \pi_o)$, where:

- $I_o : S \to \{0,1\}$ is the initiation set, which corresponds to a precondition as discussed in Section 7.

- $\beta_o : S \to [0,1]$, the termination condition, describes the probability that option o ceases execution upon reaching state s. This corresponds to a goal as discussed in Section 7, but is distinct from an effect; the goal is the set of states in which the skill *could* (or perhaps *should*) terminate, whereas the effect describes where it *actually* terminates (typically either a subset of the goal or a distribution over states in the goal).

- π_o is the option policy.

In many cases, option policies are defined indirectly using a reward function R_o, often consisting of a completion reward for reaching β_o plus a background cost function. π_o can then be obtained using any reinforcement learning algorithm, by treating β_o as an absorbing goal set. The core question for finding component skills when solving a manipulation problem is therefore to define the relevant termination goal β_o—i.e, identify the target goal—from which R_o can be constructed.

The robot may construct a *skill library*, consisting of a collection of multiple skills that can be frequently reused across tasks. This requires extracting a collection of skills, either from demonstrations, or from behaviors generated autonomously by the robot itself. The key question here is how to identify the skills, which is a difficult, and somewhat under-specified, challenge. There are two dominant approaches in the literature: segmenting task solution trajectories into individual component skills, or directly including skill specification as part of the overall problem when learning to solve tasks.

8.2. Segmenting Trajectories into Component Skills

One approach to identifying a skill library is to obtain solution trajectories for a collection of tasks, and segment those trajectories into a collection of skills that retroactively decompose the input trajectories. This is commonly done using demonstration trajectories, though it could also be performed on trajectories generated autonomously by the robot, typically after learning [143, 202, 312]. However they are generated, the resulting trajectories must be segmented into component skills. The literature contains a large array of methods for performing the segmentation, which we group into two broad categories.

Segmentation Based on Skill Similarity: The most direct approach is to segment demonstration trajectories into repeated subtasks, each of which may occur in many different contexts [74, 260]. Identifying such repeated tasks both reduces the size of the skill library (which in turn reduces the complexity of planning or learning which skill to use when) and maximizes the data available to learn each skill [247]. This requires a measure of *skill similarity* that expresses a distance metric between two candidate skill segments, or more directly models the probability with which they were generated by the same skill. Therefore, several approaches have used a measure of skill similarity to segment demonstration trajectories, often based on a variant of a Hidden Markov model, where the demonstrated behavior is modeled as the result of the execution of a sequence of latent skills; segmentation in this case amounts to inferring the most likely sequence of skills.

The most direct approach [176, 60, 129, 281, 260, 79] is *policy similarity*, which measures skill similarity by fitting the data to a parameterized policy class and measuring distance in parameter space. Alternative approaches to policy similarity fit models to the underlying value function [203] or the unobserved reward function that the skill behavior is implicitly maximizing [309, 214, 261, 64, 18]. Some approaches segment demonstration trajectories and then merge similar skills in a separate post-processing step [203], while the most principled probabilistic approaches infer shared skills across a collection of trajectories as part of the segmentation process [281, 309]. Rather than using a parametric model, a latent space can be discovered that efficiently encodes skills, which are either learned from experience in a reinforcement learning context, or via a latent-space segmentation of trajectories in an imitation learning setting [148, 149]. Other recent approaches have relied on observations only, learning to segment videos of multi-step tasks into composable primitives [127, 433, 165, 424].

A less direct approach is to measure skill-similarity based on *pre- and post-condition similarity*, where the skill policy or trajectory itself is not used for segmentation. Instead, trajectory segments that achieve the same goal can be clustered together, while those that do so from very different initial conditions could be split [218]. Thus, reaching and pushing motions are different skills due to their distinct pre- and post- conditions, i.e., moving or not moving an object, even if the skill policies are similar or the same. This approach is often used in developmental learn-

ing approaches to discover skills [415]; the robot executes a skill policy in a variety of scenarios and then clusters together the post- and pre- conditions to create distinct skills [423, 279, 397, 394, 395].

Segmentation Based on Specific Events: Several approaches use pre-designated events to indicate skill boundaries. These can range from hand-specified task-specific events to more generally applicable principles.

One common approach is to segment by *salient sensory events* defined by haptic, tactile, audio, or visual cues [182, 365, 113, 7]. For example, insertion skills are easier to monitor if they result in a distinct *click* upon successful completion. Similarly, we can determine if a light switch was properly pushed if the light goes on or the button cannot be pushed any further. Human-made objects are often designed to provide salient event feedback to reduce errors while using the objects. Such skills have the advantage that their termination conditions are easy to detect and monitor.

Another important class of pre-defined segmentation events is *transitioning between modes* [20, 219]. Switching between modes allows the robot to switch between the ability to interact with different objects. Hence, the robot must first transition to a suitable mode to manipulate an object. Skills for transitioning to specific modes allow the robot to decouple accessing a mode and using the mode to perform a manipulation task. Grasping, lifting, placing, and releasing are all examples of skills used to switch between modes in pick-and-place tasks. Mode transitions can also be verified by applying additional actions to determine if the skill execution was successful. However, multiple skills within a mode, e.g., squeezing, shaking, and tilting a held object, cannot be detected using only this type of decomposition.

Other approaches, such as the Roboearth project [409], seek to acquire a library of skills from information available online, such as natural language and video data, which can then be shared with other robots [409, 430].

8.3. Discovering Skills While Solving Tasks

An alternative approach is to discover component motor skills *during* the process of learning to solve one or more manipulation tasks. This has two crucial advantages over solution trajectory segmentation. First, learned skills could aid in solving the tasks to begin with; many complex manipulation tasks cannot be solved directly without decomposing them into simpler tasks, so it will be infeasible to require that a complete solution precede skill discovery. Second, imposing hierarchical structure *during* learning can inject bias that results in much more compact skill libraries. For example, there may be multiple ways to turn a switch; there is no reason to expect that a robot that independently learns to solve several tasks involving switches will find a

similar policy each time. However, if the learned policy for turning a switch is identified and retained in the first task, then it will likely be reused in the second task.

Broadly speaking, there are substantially fewer successful approaches that learn skills while solving tasks than there are approaches that retroactively segment, in part because the problem is fundamentally harder than segmentation. For example, skill similarity approaches are difficult to apply here. However, some researchers have applied methods based on specific salient events [143, 22] to trigger skill creation. These methods could be combined with skill chaining [206, 202]—an approach where skills are constructed to either reach a salient event or to reach another skill's preconditions—to learn the motor skills online. An alternative is to structure the policy class with which the robot learns to include hierarchical components which can be extracted after learning and reused. This approach naturally fits recent research using deep neural networks for policy learning [272, 404, 241], which are sometimes structured to be successively executable in a manner similar to skill chaining [228], but have been used in other more compact representations [77, 78]. These new results are exciting but have only just begun to scratch the surface of how component motor skills can be learned during the robot's task learning process.

8.4. Learning Decision-Making Abstractions

A collection of abstract motor skills provides *procedural* abstraction to the robot; it can abstract over the low-level details of what it must *do* to solve the task. However, these motor skills also provide an opportunity for abstracting over the input the robot uses to decide which skill to execute. Learning such abstract structures have two potential advantages: First, the new abstract input may make learning for new tasks easier (or even unnecessary by enabling generalized abstract policies that work across tasks) and support abstract, task-level planning. Second, the resulting abstract representations may be much easier for a non-expert to edit, update, and interpret.

Broadly speaking, the literature contains two types of learned abstractions for decision-making. The first type learns an abstract policy directly, while the second learns an abstract state space that serves as the basis for either planning or much faster learning in new tasks.

Learning Abstract Policy Representations: Here the goal is to learn a generalized abstract policy that solves a class of problems. Often this policy encodes a mapping from abstract quantities to learned motor skills. For example, the policy for opening a door might include first checking to see if it is locked and if so, executing the unlock skill; then grasping the handle, turning it, and opening the door. Here the motor skills subsume the low-level variations in

task execution (e.g., turning differently shaped doorknobs) while the abstract quantities subsume differences in task logic (e.g., checking to see if the door is locked). Approaches here differ primarily by the representation of the policy itself, ranging from finite-state machines [281] to associative skill memories [291] to hierarchical task networks [150] to context-free grammars [248]. In some cases these result in a natural and intuitive form of incremental policy repair and generalization [281, 150].

Learning Abstract State Spaces: Alternatively, the robot could learn abstract representations of *state*, which when combined with abstract actions result in a new, but hopefully much simpler, and typically discrete, MDP. The robot can then use that simpler MDP to either learn faster, or to learn a task model and then plan. Several approaches have been applied here, for example using the status of the skills available to the robot as a state space [40, 143], finding representations that minimize planning loss [225, 177, 394, 395], using qualitative state abstractions [271] and constructing compact MDPs that are provably sound and complete for task-level planning [200, 208, 11]. These approaches offer a natural means of abstracting away the low-level detail common to learning for manipulation tasks, and exploiting the structure common to task families to find minimal compact descriptions that support maximally efficient learning.

Taken together, hierarchical and compositional approaches have great promise for exploiting the structure in manipulation learning tasks, to reduce sample complexity and achieve generality, and have only begun to be carefully explored. This is a challenging area full of important questions, and where several breakthroughs still remain to be made.

9. Conclusion

This paper has presented an overview of key manipulation challenges and the types of robot learning algorithms that have been developed to address these challenges. We explained how robots can represent objects and learn features hierarchically, and how these features and object properties can be estimated using passive and interactive perception. We have discussed how the effects of manipulations can themselves be captured by learning continuous, discrete, and hybrid transition models. Different data collection strategies and model types determine how quickly the robot can learn the transition models and how well the models generalize to new scenarios.

Skill policies can often be learned quickly from human demonstrations using behavioural cloning or apprenticeship learning, while mastery of manipulation skills often requires additional experience and can be achieved using

model-based or model-free reinforcement learning. Given a skill, the robot can learn its preconditions and effects to capture its utility in different scenarios. Learning to monitor manipulation skills by detecting errors and goals imbues them with an additional level of robustness for working in unstructured environments.

Finally, we have seen how a robot can exploit the modular nature of manipulation tasks, such that each learned skill can be incorporated into a larger hierarchy to perform more advanced tasks, promoting reusability and robustness of skills across different tasks.

Given the multi-faceted nature of manipulation, researchers have found great utility in being able to draw from methods that span the breadth of machine learning. However, despite access to these excellent generic machine learning methods, the challenges of learning robust and versatile manipulation skills are still far from being resolved. Some—but by no means all—of these pressing challenges are:

- Integrating learning into complete control systems

- Using learned components as they are being learned (*in situ* learning)

- Safe learning, and learning with guarantees

- Exploiting and integrating multiple sensory modalities, including human cues

- Better exploration strategies, possibly based on explicit hypotheses or causal reasoning

- Exploiting common-sense physical knowledge

- Better algorithms for transfer across substantially different families of tasks

- Drastically improving the sample complexity of policy learning algorithms, while avoiding having to empirically tune hyper-parameters

As the community tackles these and other challenges, we expect that the core themes that have emerged repeatedly in manipulation learning research—modularity and hierarchy, generalization across objects, and the need for autonomous discovery—will continue playing a key role in designing effective solutions.

Acknowledgment

References

[1] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine learning*, pages 1–8, 2004.

[2] Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *International Journal of Robotics Research*, 29 (13):1608–1639, 2010.

[3] Navid Aghasadeghi and Timothy Bretl. Maximum entropy inverse reinforcement learning in continuous state spaces with path integrals. In *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1561–1566, 2011.

[4] Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems 29*, pages 5074–5082, 2016.

[5] Anurag Ajay, Jiajun Wu, Nima Fazeli, Maria Bauza, Leslie Pack Kaelbling, Joshua B. Tenenbaum, and Alberto Rodriguez. Augmenting physical simulators with stochastic neural networks: Case study of planar pushing and bouncing. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3066–3073, 2018.

[6] Baris Akgun, Maya Cakmak, Karl Jiang, and Andrea L. Thomaz. Keyframe-based learning from demonstration: Method and evaluation. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[7] Eren Erdal Aksoy, Alexey Abramov, Johannes Dörr, Kejun Ning, Babette Dellen, and Florentin Wörgötter. Learning the semantics of object-action relations by observation. *International Journal of Robotics Research*, 30(10):1229–1249, 2011.

[8] Ferran Alet, Tomas Lozano-Perez, and Leslie P. Kaelbling. Modular meta-learning. In *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 856–868, 2018.

[9] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, 2012.

[10] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[11] Barrett Ames, Allison Thackston, and G.D. Konidaris. Learning symbolic representations for planning with parameterized skills. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 526–533, 2018.

[12] H. Ben Amor, O. Kroemer, U. Hillenbrand, G. Neumann, and J. Peters. Generalization of human grasping for multi-fingered robot hands. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2043–2050, 2012.

[13] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems 30*, pages 5048–5058, 2017.

[14] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[15] Christopher G. Atkeson, Andrew W. Moore, and Stefan Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11(1-5):11–73, 1997.

[16] Christopher G Atkeson, Andrew W Moore, and Stefan Schaal. Locally weighted learning for control. *Artificial Intelligence Review*, 11(1-5):75–113, 1997.

[17] Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H. Campbell, and Sergey Levine. Stochastic variational video prediction. In *Proceedings of the 2018 International Conference on Learning Representations*, 2018.

[18] M. Babes, V. Marivate, K. Subramanian, and M. L. Littman. Apprenticeship learning about multiple intentions. In *Proceedings of the 28th International Conference on Machine Learning*, pages 897–904, 2011.

[19] JA Bagnell, Joel Chestnutt, David M Bradley, and Nathan D Ratliff. Boosting structured prediction for imitation learning. In *Advances in Neural Information Processing Systems 19*, pages 1153–1160, 2007.

[20] Andrea Baisero, Yoan Mollard, Manuel Lopes, Marc Toussaint, and Ingo Lutkebohle. Temporal segmentation of pair-wise interaction phases in sequential manipulation demonstrations. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 478–484, 2015.

[21] P.R. Barragán, L. P. Kaelbling, and T. Lozano-Perez. Interactive Bayesian identification of kinematic mechanisms. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 2013–2020, 2014.

[22] Manuel Baum, Matthew Bernstein, Roberto Martín-Martín, Sebastian Höfer, Johannes Kulick, Marc Toussaint, Alex Kacelnik, and Oliver Brock. Opening a lockbox through physical exploration. In *Proceedings of the 2017 IEEE-RAS International Conference on Humanoid Robots*, pages 461–467, 2017.

[23] M. Bauza and A. Rodriguez. A probabilistic data-driven model for planar pushing. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, pages 3008–3015, 2017.

[24] M. Beetz, L. Mösenlechner, and M. Tenorth. CRAM: A cognitive robot abstract machine for everyday manipulation in human environments. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1012–1017, 2010.

[25] Y. Bekiroglu, D. Kragic, and V. Kyrki. Learning grasp stability based on tactile data and hmms. In *Proceedings of the 19th International Symposium in Robot and Human Interactive Communication*, pages 132–137, 2010.

[26] Yasemin Bekiroglu, Dan Song, Lu Wang, and Danica Kragic. A probabilistic framework for task-oriented grasp stability assessment. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation*, pages 3040–3047, 2013.

[27] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems 29*, pages 1471–1479, 2016.

[28] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 41–48, 2009.

[29] N. Bergström, C. H. Ek, D. Kragic, Y. Yamakawa, T. Senoo, and M. Ishikawa. On-line learning of temporal state models for flexible objects. In *Proceedings of the 2012 IEEE-RAS International Conference on Humanoid Robots*, pages 712–718, 2012.

[30] Tapomayukh Bhattacharjee, Joshua Wade, and Charles Kemp. Material recognition from heat transfer given varying initial conditions and short-duration contact. In *Robotics: Science and Systems XI*, 2015.

[31] M. Björkman, Y. Bekiroglu, V. Högman, and D. Kragic. Enhancing visual perception of shape through tactile glances. In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3180–3186, 2013.

[32] J. Bohg, A. Morales, T. Asfour, and D. Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30(2):289–309, 2014.

[33] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

[34] Jeannette Bohg and Danica Kragic. Learning grasping points with shape context. *Robotics and Autonomous Systems*, 58(4):362–377, 2010.

[35] Jeannette Bohg, Kai Welke, Beatriz León, Martin Do, Dan Song, Walter Wohlkinger, Marianna Madry, Aitor Aldóma, Markus Przybylski, Tamim Asfour, Higinio Martí, Danica Kragic, Antonio Morales, and Markus Vincze. Task-based grasp adaptation on a humanoid robot. In *Proceedings of the 10th IFAC Symposium on Robot Control*, pages 779–786.

[36] Byron Boots, Arthur Gretton, and Geoffrey J Gordon. Hilbert space embeddings of predictive state representations. In *Proceedings of the 29th International Conference on Uncertainty in Artificial Intelligence*, pages 92–101, 2013.

[37] Abdeslam Boularias, Oliver Kroemer, and Jan Peters. Structured apprenticeship learning. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 227–242. Springer Berlin Heidelberg, 2012.

[38] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, 2018.

[39] S. Brandl, O. Kroemer, and J. Peters. Generalizing pouring actions between objects using warped parameters. In *Proceedings of the 2014 IEEE-RAS International Conference on Humanoid Robots*, pages 616–621, 2014.

[40] O. Brock, A. Fagg, R. Grupen, R. Platt, M. Rosenstein, and J. Sweeney. A framework for learning and control in intelligent humanoid robots. *International Journal of Humanoid Robotics*, 2(3):301–336, 2005.

[41] P. Brook, M. Ciocarlie, and K. Hsiao. Collaborative grasp planning with multiple object representations. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 2851–2858, 2011.

[42] Jonathan Brookshire and Seth Teller. Articulated pose estimation using tangent space approximations. *International Journal of Robotics Research*, 35(1-3):5–29, 2016.

[43] Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *Proceedings of the 36th International Conference on Machine Learning*, pages 783–792, 2019.

[44] Daniel S Brown and Scott Niekum. Efficient probabilistic performance bounds for inverse reinforcement learning. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[45] Daniel S Brown and Scott Niekum. Machine teaching for inverse reinforcement learning: Algorithms and applications. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019.

[46] Daniel S Brown, Yuchen Cui, and Scott Niekum. Risk-aware active inverse reinforcement learning. In *Proceedings of the 2nd Conference on Robot Learning*, pages 362–372, 2018.

[47] Kalesha Bullard, Sonia Chernova, and Andrea Lockerd Thomaz. Human-driven feature selection for a robotic agent learning classification tasks from demonstration. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 6923–6930, 2018.

[48] Benjamin Burchfiel and George Konidaris. Bayesian eigenobjects: A unified framework for 3D robot perception. In *Robotics: Science and Systems XIII*, 2017.

[49] Benjamin Burchfiel and George Konidaris. Hybrid Bayesian eigenobjects: Combining linear subspace and deep network methods for 3D robot vision. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6843–6850, 2018.

[50] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation.

In *Proceedings of the 2018 International Conference on Learning Representations*, 2018.

[51] Arunkumar Byravan and Dieter Fox. SE3-nets: Learning rigid body motion using deep neural networks. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 173–180, 2016.

[52] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H. Adelson, and Sergey Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018.

[53] Sylvain Calinon. Robot learning with task-parameterized generative models. In Antonio Bicchi and Wolfram Burgard, editors, *Proceedings of the 2018 International Symposium on Robotics Research*, pages 111–126, 2018.

[54] Sylvain Calinon, Florent Guenter, and Aude Billard. On learning, representing, and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(2):286–298, 2007.

[55] Thomas Cederborg, Ming Li, Adrien Baranes, and Pierre-Yves Oudeyer. Incremental local online Gaussian mixture regression for imitation learning of multiple tasks. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 267–274, 2010.

[56] Yevgen Chebotar, Karol Hausman, Marvin Zhang, Gaurav Sukhatme, Stefan Schaal, and Sergey Levine. Combining model-based and model-free updates for trajectory-centric reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 703–711, 2017.

[57] Taolue Chen, Marta Kwiatkowska, Aistis Simaitis, and Clemens Wiltsche. Synthesis for multi-objective stochastic games: An application to autonomous urban driving. In *International Conference on Quantitative Evaluation of Systems*, pages 322–337, 2013.

[58] Nuttapong Chentanez, Andrew G. Barto, and Satinder P. Singh. Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems 17*, pages 1281–1288, 2005.

[59] Sonia Chernova and Manuela Veloso. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research*, 34:1–25, 2009.

[60] S. Chiappa and J. Peters. Movement extraction by detecting dynamics switches and repetitions. In *Advances in Neural Information Processing Systems 23*, pages 388–396, 2010.

[61] Sandeep Chinchali, Scott C Livingston, Ufuk Topcu, Joel W Burdick, and Richard M Murray. Towards formal synthesis of reactive controllers for dexterous robotic manipulation. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pages 5183–5189, 2012.

[62] S. Chitta, M. Piccoli, and J. Sturm. Tactile object class and internal state recognition for mobile manipulation. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pages 2342–2348, 2010.

[63] Sachin Chitta, Jürgen Sturm, Matthew Piccoli, and Wolfram Burgard. Tactile sensing for mobile manipulation. *IEEE Transactions on Robotics*, 27(3):558–568, 2011.

[64] J. Choi and K.-E. Kim. Nonparametric Bayesian inverse reinforcement learning for multiple reward functions. In *Advances in Neural Information Processing Systems 25*, pages 305–313, 2012.

[65] Shushman Choudhury, Yifan Hou, Gilwoo Lee, and Siddhartha S. Srinivasa. Hybrid DDP in clutter (CHDDP): trajectory optimization for hybrid dynamical system in cluttered environments. *CoRR*, abs/1710.05231, 2017. URL http://arxiv.org/abs/1710.05231.

[66] Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. Risk-sensitive and robust decision-making: a cvar optimization approach. In *Advances in Neural Information Processing Systems 28*, pages 1522–1530, 2015.

[67] M. Ciocarlie, C. Goldfeder, and P. Allen. Dimensionality reduction for hand-independent dexterous robotic grasping. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3270–3275, 2007.

[68] Matei Ciocarlie, Kaijen Hsiao, Edward Gil Jones, Sachin Chitta, Radu Bogdan Rusu, and Ioan A. Şucan. Towards reliable grasping and manipulation in household environments. In Oussama Khatib, Vijay Kumar, and Gaurav Sukhatme, editors, *Proceedings of the 2014 International Symposium on Experimental Robotics*, pages 241–252, 2014.

[69] L.C. Cobo, K. Subramanian, C.L. Isbell, A.D. Lanterman, and A.L. Thomaz. Abstraction from demonstration for efficient reinforcement learning in high-dimensional domains. *Artificial Intelligence*, 216:103–128, 2014.

[70] Silvia Coradeschi, Amy Loutfi, and Britta Wrede. A short review of symbol grounding in robotic and intelligent systems. *Künstliche Intelligenz*, 27(2):129–136, 2013.

[71] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[72] Yuchen Cui and Scott Niekum. Active reward learning from critiques. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 6907–6914, 2018.

[73] Bruno Castro Da Silva, Gianluca Baldassarre, George D Konidaris, and Andrew G Barto. Learning parameterized motor skills on a humanoid robot. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 5239–5244, 2014.

[74] H. Dang and P. K. Allen. Robot learning of everyday object manipulations via human demonstration. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1284–1289, 2010.

[75] H. Dang and P. K. Allen. Learning grasp stability. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pages 2392–2397, 2012.

[76] H. Dang and P. K. Allen. Grasp adjustment on novel objects using tactile experience from similar local geometry. In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4007–4012, 2013.

[77] Christian Daniel, Gerhard Neumann, Oliver Kroemer, and Jan Peters. Learning sequential motor tasks. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation*, pages 2626–2632, 2013.

[78] Christian Daniel, Gerhard Neumann, Oliver Kroemer, and Jan Peters. Hierarchical relative entropy policy search. *Journal of Machine Learning Research*, 17(1):3190–3239, 2016.

[79] Christian Daniel, Herke Van Hoof, Jan Peters, and Gerhard Neumann. Probabilistic inference for determining options in reinforcement learning. *Machine Learning*, 104(2-3): 337–357, 2016.

[80] Lawrence Davis. *Handbook of Genetic Algorithms*. Chapman & Hall, 1991.

[81] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, 2005.

[82] Thomas J. Debus, Pierre E. Dupont, and Robert D. Howe. Contact state estimation using multiple model estimation and hidden Markov models. *International Journal of Robotics Research*, 23(4-5):399–413, 2004.

[83] Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on Machine Learning*, pages 465–472, 2011.

[84] Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, 2015.

[85] Karthik Desingh, Shiyang Lu, Anthony Opipari, and Odest Chadwicke Jenkins. Factored pose estimation of articulated objects using efficient nonparametric belief propagation. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, 2018.

[86] R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, and J. Piater. Learning grasp affordance densities. *Paladyn*, 2(1), 2011.

[87] Renaud Detry, Carl Henrik Ek, Marianna Madry, and Danica Kragic. Learning a dictionary of prototypical grasp-predicting parts from grasping experience. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation*, pages 601–608, 2013.

[88] Renaud Detry, Jeremie Papon, and Larry Matthies. Task-oriented grasping with semantic and geometric scene understanding. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3266–3273, 2017.

[89] Coline Devin, Pieter Abbeel, Trevor Darrell, and Sergey Levine. Deep object-centric representations for generalizable robot learning. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 7111–7118, 2018.

[90] C. Diuk, L. Li, and B.R. Leffler. The adaptive k-meteorologists problems and its application to structure learning and feature selection in reinforcement learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 249–256, 2009.

[91] Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 240–247, 2008.

[92] M. Do, J. Schill, J. Ernesti, and T. Asfour. Learn to wipe: A case study of structural bootstrapping from sensorimotor experience. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 1858–1864, 2014.

[93] Andreas Doerr, Nathan D Ratliff, Jeannette Bohg, Marc Toussaint, and Stefan Schaal. Direct loss minimization inverse optimal control. In *Robotics: Science and Systems XI*, 2015.

[94] M. R. Dogar, M. Cakmak, E. Ugur, and E. Sahin. From primitive behaviors to goal-directed behavior using affordances. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 729–734, 2007.

[95] Stanimir Dragiev, Marc Toussaint, and Michael Gienger. Uncertainty aware grasping and tactile exploration. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation*, pages 113–119, 2013.

[96] Yan Duan, Marcin Andrychowicz, Bradly Stadie, OpenAI Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In *Advances in Neural Information Processing Systems 30*, pages 1087–1098, 2017.

[97] Krishnamurthy Dvijotham and Emanuel Todorov. Inverse optimal control with linearly-solvable MDPs. In *Proceedings of the 27th International Conference on Machine Learning*, pages 335–342, 2010.

[98] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

[99] Aaron Edsinger and Charles C Kemp. Toward robot learning of tool manipulation from human demonstration.

[100] Yaakov Engel, Peter Szabo, and Dmitry Volkinshtein. Learning to control an octopus arm with Gaussian process temporal difference methods. In *Advances in Neural Information Processing Systems 19*, pages 347–354, 2006.

[101] P. Englert and M. Toussaint. Learning manipulation skills from a single demonstration. *International Journal of Robotics Research*, 37(1):137–154, 2018.

[102] P. Englert and M. Toussaint. Kinematic morphing networks for manipulation skill transfer. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2517–2523, 2018.

[103] P. Englert, N.A. Vien, and M. Toussaint. Inverse KKT: Learning cost functions of manipulation tasks from demonstrations. *International Journal of Robotics Research*, 36(13-14):1474–1488, 2017.

[104] Peter Englert and Marc Toussaint. Combined optimization and reinforcement learning for manipulation skills. In *Robotics: Science and Systems XII*, 2016.

[105] Andrew H Fagg and Michael A Arbib. Modeling parietal-premotor interactions in primate control of grasping. *Neural Networks*, 11(7):1277 – 1303, 1998.

[106] Georgios E Fainekos, Antoine Girard, Hadas Kress-Gazit, and George J Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

[107] Zhou Fang, Georg Bartels, and Michael Beetz. Learning models for constraint-based motion parameterization from

interactive physics-based simulation. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4005–4012, 2016.

[108] Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value expansion for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*, 2018.

[109] S. Fichtl, A. McManus, W. Mustafa, D. Kraft, N. Krger, and F. Guerin. Learning spatial relationships from 3D vision using histograms. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 501–508, 2014.

[110] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, pages 2786–2793, 2017.

[111] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 49–58, 2016.

[112] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1126–1135, 2017.

[113] Paul Fitzpatrick, Artur Arsenio, and Eduardo R. Torres-Jara. Reinforcing robot perception of multi-modal events through repetition and redundancy and repetition and redundancy. *Interaction Studies*, 7(2):171–196, 2006.

[114] Peter R. Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 373–385, 2018.

[115] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Proceedings of The 1st Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 482–495. PMLR, 2017.

[116] Jie Fu and Ufuk Topcu. Probably approximately correct MDP learning and control with temporal logic constraints. *arXiv preprint arXiv:1404.7073*, 2014.

[117] Justin Fu, Sergey Levine, and Pieter Abbeel. One-shot learning of manipulation skills with online dynamics adaptation and neural network priors. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4019–4026, 2016.

[118] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. In *Proceedings of the 2017 International Conference on Learning Representations*, 2017.

[119] Yang Gao, Lisa Anne Hendricks, Katherine J. Kuchenbecker, and Trevor Darrell. Deep learning for tactile understanding from visual and haptic data. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 536–543, 2016.

[120] Cristina Garcia Cifuentes, Jan Issac, Manuel Wthrich, Stefan Schaal, and Jeannette Bohg. Probabilistic articulated

real-time tracking for robot manipulation. *IEEE Robotics and Automation Letters*, 2(2):577–584, 2017.

[121] Caelan Reed Garrett, Tomas Lozano-Perez, and Leslie Pack Kaelbling. FFRob: Leveraging symbolic planning for efficient task and motion planning. *International Journal of Robotics Research*, 37(1):104–136, 2018.

[122] Mohammad Ghavamzadeh, Marek Petrik, and Yinlam Chow. Safe policy improvement by minimizing robust baseline regret. In *Advances in Neural Information Processing Systems 29*, pages 2298–2306, 2016.

[123] James J Gibson. *The ecological approach to visual perception*. Psychology Press, 2014.

[124] Aditya Goil, Matthew Derry, and Brenna D Argall. Using machine learning to blend human and robot controls for assisted wheelchair navigation. In *Proceedings of the 2013 IEEE International Conference on Rehabilitation Robotics*, pages 1–6, 2013.

[125] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. K. Allen. Data-driven grasping with partial sensor data. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1278–1283, 2009.

[126] Wonjoon Goo and Scott Niekum. Learning multi-step robotic tasks from observation. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, 2019.

[127] Wonjoon Goo and Scott Niekum. One-shot learning of multi-step tasks from observation via activity localization in auxiliary video. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, 2019.

[128] Shane Griffith, Vladimir Sukhoy, Todd Wegter, and Alexander Stoytchev. Object categorization in the sink: Learning behavior-grounded object categories with water. In *ICRA 2012 Workshop on Semantic Perception, Mapping, and Exploration*, 2012.

[129] D.H. Grollman and O.C. Jenkins. Incremental learning of subtasks from unsegmented demonstration. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 261–266, 2010.

[130] Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep Q-learning with model-based acceleration. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2829–2838, 2016.

[131] M. Gualtieri and R. Platt. Learning 6-DoF grasping and pick-place using attention focus. In *Proceedings of the 2nd Conference on Robot Learning*, pages 477–486, 2018.

[132] Marcus Gualtieri and Robert Platt. Viewpoint selection for grasp detection. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 258–264, 2017.

[133] Charlie Guan, William Vega-Brown, and Nicholas Roy. Efficient planning for near-optimal compliant manipulation leveraging environmental contact. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 215–222, 2018.

[134] Puren Guler, Yasemin Bekiroglu, Xavi Gratal, Karl Pauwels, and Danica Kragic. What's in the container? classifying object contents from vision and touch. In *Proceed-

ings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3961–3968, 2014.

[135] Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. In *Proceedings of the 2017 International Conference on Learning Representations*, 2017.

[136] Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems 31*, pages 5302–5311, 2018.

[137] Megha Gupta, Jörg Müller, and Gaurav S. Sukhatme. Using manipulation primitives for object sorting in cluttered environments. *IEEE Transactions on Automation Science and Engineering*, 12(2):608–614, 2015.

[138] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1861–1870, 2018.

[139] Dylan Hadfield-Menell, Stuart J Russell, Pieter Abbeel, and Anca Dragan. Cooperative inverse reinforcement learning. In *Advances in Neural Information Processing Systems 29*, pages 3909–3917, 2016.

[140] Josiah P Hanna and Peter Stone. Grounded action transformation for robot learning in simulation. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*, pages 3834–3840, 2017.

[141] Josiah P Hanna, Peter Stone, and Scott Niekum. Bootstrapping with models: Confidence intervals for off-policy evaluation. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems*, pages 538–546, 2017.

[142] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[143] S. Hart. Intrinsically motivated hierarchical manipulation. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, pages 3814–3819, 2008.

[144] Stephen Hart. An intrinsic reward for affordance exploration. In *Proceedings of the 2009 IEEE International Conference on Development and Learning*, pages 1–6, 2009.

[145] Kris Hauser. *Motion Planning for Legged and Humanoid Robots*. PhD thesis, Stanford University, 2018.

[146] K Hausman, S Niekum, S Osentoski, and G S Sukhatme. Active articulation model estimation through interactive perception. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 3305–3312, 2015.

[147] Karol Hausman, Christian Bersch, Dejan Pangercic, Sarah Osentoski, Zoltan-Csaba Marton, and Michael Beetz. Segmentation of cluttered scenes through interactive perception. In *ICRA 2012 Workshop on Semantic Perception and Mapping for Knowledge-enabled Service Robotics*, 2012.

[148] Karol Hausman, Yevgen Chebotar, Stefan Schaal, Gaurav Sukhatme, and Joseph J Lim. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. In *Advances in Neural Information Processing Systems 30*, pages 1235–1245, 2017.

[149] Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. Learning an embedding space for transferable robot skills. In *Proceedings of the 2018 International Conference on Learning Representations*, 2018.

[150] Bradley Hayes and Brian Scassellati. Autonomously constructing hierarchical task networks for planning and human-robot collaboration. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 5469–5476, 2016.

[151] Gillian M Hayes and John Demiris. *A robot controller using learning by imitation*. University of Edinburgh, Department of Artificial Intelligence, 1994.

[152] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the 2017 IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[153] T. Hermans, J. M. Rehg, and A. Bobick. Guided pushing for object singulation. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4783–4790, 2012.

[154] Alexander Herzog, Peter Pastor, Mrinal Kalakrishnan, Ludovic Righetti, Jeannette Bohg, Tamim Asfour, and Stefan Schaal. Learning of grasp selection based on shape-templates. *Autonomous Robots*, 36(1-2):51–65, 2014.

[155] Irina Higgins, Arka Pal, Andrei A Rusu, Loic Matthey, Christopher P Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. DARLA: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1480–1490, 2017.

[156] U. Hillenbrand and M. A. Roa. Transferring functional grasps through contact warping and local replanning. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2963–2970, 2012.

[157] Martin Hjelm, Renaud Detry, Carl Henrik Ek, and Danica Kragic. Representations for cross-task, cross-object grasp transfer. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 5699–5704, 2014.

[158] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems 29*, pages 4565–4573, 2016.

[159] V. Högman, M. Björkman, A. Maki, and D. Kragic. A sensorimotor learning framework for object categorization. *IEEE Transactions on Cognitive and Developmental Systems*, 8(1):15–25, 2016.

[160] Jarrett Holtz, Arjun Guha, and Joydeep Biswas. Interactive robot transition repair with SMT. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4905–4911, 2018.

[161] Rein Houthooft, Yuhua Chen, Phillip Isola, Bradly Stadie, Filip Wolski, OpenAI Jonathan Ho, and Pieter Abbeel. Evolved policy gradients. In *Advances in Neural Information Processing Systems 31*, pages 5400–5409, 2018.

[162] Geir E. Hovland and Brenan J. McCarragher. Hidden Markov models as a process monitor in robotic assembly. *International Journal of Robotics Research*, 17(2):153–168, 1998.

[163] K. Hsiao, L.P. Kaelbling, and T. Lozano-Perez. Grasping POMDPs. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation*, pages 4685–4692, 2007.

[164] Kaijen Hsiao, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Task-driven tactile exploration. In *Robotics: Science and Systems VI*, 2010.

[165] De-An Huang, Suraj Nair, Danfei Xu, Yuke Zhu, Animesh Garg, Li Fei-Fei, Silvio Savarese, and Juan Carlos Niebles. Neural task graphs: Generalizing to unseen tasks from a single video demonstration. In *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[166] Han-Pang Huang and Chao-Chiun Liang. Strategy-based decision making of a soccer robot system using a real-time self-organizing fuzzy decision tree. *Fuzzy Sets and Systems*, 127(1):49–64, 2002.

[167] Xiao Huang and John Weng. Novelty and reinforcement learning in the value system of developmental robots. pages 47–55, 2002.

[168] Phillip Isola, Joseph J. Lim, and Edward H. Adelson. Discovering states and transformations in image collections. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1383–1391, 2015.

[169] Jan Issac, Manuel Wüthrich, Cristina Garcia Cifuentes, Jeannette Bohg, Sebastian Trimpe, and Stefan Schaal. Depth-based object tracking using a robust gaussian filter. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 608–615, 2016.

[170] Advait Jain and Charles C. Kemp. Improving robot manipulation with data-driven object-centric models of everyday forces. *Autonomous Robots*, 35(2-3):143–159, 2013.

[171] Ajinkya Jain and Scott Niekum. Efficient hierarchical robot motion planning under uncertainty and hybrid dynamics. In *Proceedings of the 2nd Conference on Robot Learning*, pages 757–766, 2018.

[172] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor. Affordances in psychology, neuroscience, and robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems*, 10 (1):4–25, 2018.

[173] Eric Jang, Coline Devin, Vincent Vanhoucke, and Sergey Levine. Grasp2vec: Learning object representations from self-supervised grasping. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 99–112, 2018.

[174] Michael Janner, Sergey Levine, William T. Freeman, Joshua B. Tenenbaum, Chelsea Finn, and Jiajun Wu. Reasoning about physical interactions with object-oriented prediction and planning. In *Proceedings of the 2019 International Conference on Learning Representations*, 2019.

[175] Shervin Javdani, Matthew Klingensmith, J. Andrew (Drew) Bagnell, Nancy Pollard, and Siddhartha Srinivasa. Efficient touch based localization through submodularity. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation*, pages 1828–1835, 2013.

[176] O.C. Jenkins and M. Matarić. Performance-derived behavior vocabularies: data-driven acquisition of skills from motion. *International Journal of Humanoid Robotics*, 1(2): 237–288, 2004.

[177] N. Jetchev, T. Lang, and M. Toussaint. Learning grounded relational symbols from continuous data for abstract reasoning. In *ICRA 2013 Workshop on Autonomous Learning*, 2013.

[178] N. Jiang, A. Kulesza, and S. Singh. Abstraction selection in model-based reinforcement learning. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 179–188, 2015.

[179] Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 652–661, 2016.

[180] Yun Jiang, Marcus Lim, Changxi Zheng, and Ashutosh Saxena. Learning to place new objects in a scene. *International Journal of Robotics Research*, 31(9):1021–1043, 2012.

[181] Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39 (3):407–428, 2015.

[182] Jonathan Juett and Benjamin Kuipers. Learning to grasp by extending the peri-personal space graph. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 8695–8700, 2018.

[183] Philipp Jund, Andreas Eitel, Nichola Abdo, and Wolfram Burgard. Optimization beyond the convolution: Generalizing spatial relations with end-to-end metric learning. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 1–7, 2018.

[184] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[185] G. Kahn, P. Sujan, S. Patil, S. Bopardikar, J. Ryde, K. Goldberg, and P. Abbeel. Active exploration using trajectory optimization for robotic grasping in the presence of occlusions. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 4783–4790, 2015.

[186] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 651–673, 2018.

[187] Parameswaran Kamalaruban, Rati Devidze, Volkan Cevher, and Adish Singla. Interactive teaching algorithms for inverse reinforcement learning. *arXiv preprint arXiv:1905.11867*, 2019.

[188] Ken Kansky, Tom Silver, David A Mély, Mohamed Eldawy, Miguel Lázaro-Gredilla, Xinghua Lou, Nimrod Dorfman, Szymon Sidor, Scott Phoenix, and Dileep George. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1809–1818, 2017.

[189] Daniel Kappler, Peter Pastor, Mrinal Kalakrishnan, Manuel Wüthrich, and Stefan Schaal. Data-driven online decision making for autonomous manipulation. In *Robotics: Science and Systems XI*, 2015.

[190] Dov Katz and Oliver Brock. A factorization approach to manipulation in unstructured environments. In Cédric Pradalier, Roland Siegwart, and Gerhard Hirzinger, editors, *Proceedings of the 2011 International Symposium on Robotics Research*, pages 285–300, 2011.

[191] Dov Katz, Andreas Orthey, and Oliver Brock. Interactive perception of articulated objects. In *Proceedings of the 2010 International Symposium on Experimental Robotics*, pages 1–15, 2010.

[192] Lukas Kaul, Simon Ottenhaus, Pascal Weiner, and Tamim Asfour. The sense of surface orientation - a new sensor modality for humanoid robots. In *Proceedings of the 2016 IEEE-RAS International Conference on Humanoid Robots*, pages 820–825, 2016.

[193] Jacqueline Kenney, Thomas Buckley, and Oliver Brock. Interactive segmentation for manipulation in unstructured environments. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, pages 1343–1348, 2009.

[194] Kiho Kim, Hohee Lee, Jangjin Park, and Myungseung Yang. Robotic contamination cleaning system. In *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1874–1879, 2002.

[195] Matthew Klingensmith, Martin Hermann, and Siddhartha Srinivasa. Object modeling and recognition from sparse, noisy data via voxel depth carving. In *Proceedings of the 2014 International Symposium on Experimental Robotics*, June 2014.

[196] Ross A Knepper, Stefanie Tellex, Adrian Li, Nicholas Roy, and Daniela Rus. Recovering from failure by asking for help. *Autonomous Robots*, 39(3):347–362, 2015.

[197] W Bradley Knox, Peter Stone, and Cynthia Breazeal. Training a robot via human feedback: A case study. In *Proceedings of the 2013 International Conference on Social Robotics*, pages 460–470, 2013.

[198] Jens Kober and Jan R Peters. Policy search for motor primitives in robotics. In *Advances in Neural Information Processing Systems 22*, pages 849–856, 2009.

[199] Jens Kober, Erhan Oztop, and Jan Peters. Reinforcement learning to adjust robot movements to new situations. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 2650–2655, 2011.

[200] G.D. Konidaris. Constructing abstraction hierarchies using a skill-symbol loop. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 1648–1654, 2016.

[201] G.D. Konidaris and A.G. Barto. Efficient skill learning using abstraction selection. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1107–1112, July 2009.

[202] G.D. Konidaris, S.R. Kuindersma, R.A. Grupen, and A.G. Barto. Autonomous skill acquisition on a mobile manipulator. In *Proceedings of the Twenty-Fifth Conference on Artificial Intelligence*, pages 1468–1473, 2011.

[203] G.D. Konidaris, S.R. Kuindersma, R.A. Grupen, and A.G. Barto. Robot learning from demonstration by constructing skill trees. *International Journal of Robotics Research*, 31 (3):360–375, 2012.

[204] G.D. Konidaris, L.P. Kaelbling, and T. Lozano-Perez. Symbol acquisition for probabilistic high-level planning. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 3619–3627, 2015.

[205] George Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 895–900, 2007.

[206] George Konidaris and Andrew G Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems 22*, pages 1015–1023, 2009.

[207] George Konidaris, Sarah Osentoski, and Philip S Thomas. Value function approximation in reinforcement learning using the Fourier basis. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pages 380–385, 2011.

[208] George Konidaris, Leslie Pack Kaelbling, and Tomas Lozano-Perez. From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research*, 61:215–289, 2018.

[209] Marek Kopicki, Renaud Detry, Maxime Adjigble, Rustam Stolkin, Ales Leonardis, and Jeremy L Wyatt. One-shot learning and generation of dexterous grasps for novel objects. *International Journal of Robotics Research*, 35(8): 959–976, 2016.

[210] Marek Kopicki, Sebastian Zurek, Rustam Stolkin, Thomas Moerwald, and Jeremy L. Wyatt. Learning modular and transferable forward models of the motions of push manipulated objects. *Autonomous Robots*, 41(5):1061–1082, 2017.

[211] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. *Advanced Robotics*, 25(5):581–603, 2011.

[212] Michael C. Koval, Matthew Klingensmith, Siddhartha S. Srinivasa, Nancy Pollard, and Michael Kaess. The manifold particle filter for state estimation on high-dimensional implicit manifolds. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, pages 4673–4680, 2017.

[213] Dirk Kraft, Nicolas Pugeault, Emre Baseski, Mila Popovic, Danica Kragic, Sinan Kalkan, Florentin Wörgötter, and Norbert Krüger. Birth of the object: Detection of objectness and extraction of object shape through object-action complexes. *International Journal of Humanoid Robotics*, 5(2):247–265, 2008.

[214] Sanjay Krishnan, Animesh Garg, Richard Liaw, Brijen Thananjeyan, Lauren Miller, Florian T Pokorny, and Ken Goldberg. SWIRL: A sequential windowed inverse reinforcement learning algorithm for robot tasks with delayed rewards. *International Journal of Robotics Research*, 38 (2-3):126–145, 2019.

[215] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, 2012.

[216] O. Kroemer and G. S. Sukhatme. Learning spatial preconditions of manipulation skills using random forests. In *Proceedings of the 2016 IEEE-RAS International Conference on Humanoid Robots*, pages 676–683, 2016.

[217] O. Kroemer, E. Ugur, E. Oztop, and J. Peters. A kernel-based approach to direct action perception. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pages 2605–2610, 2012.

[218] O. Kroemer, H. van Hoof, G. Neumann, and J. Peters. Learning to predict phases of manipulation tasks as hidden states. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 4009–4014, 2014.

[219] O. Kroemer, C. Daniel, G. Neumann, H. van Hoof, and J. Peters. Towards learning hierarchical skills for multi-phase manipulation tasks. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 1503–1510, 2015.

[220] O. Kroemer, S. Leischnig, S. Luettgen, and J. Peters. A kernel-based approach to learning contact distributions for robot manipulation tasks. *Autonomous Robots*, 42(3):581–600, 2018.

[221] O. B. Kroemer, R. Detry, J. Piater, and J. Peters. Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems*, 59(9):1105–1116, 2010. doi: 10.1016/j.robot.2010.06.001.

[222] Oliver Kroemer and Jan Peters. Predicting object interactions from contact distributions. In *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3361–3367, 2014.

[223] Oliver Kroemer and Gaurav Sukhatme. Meta-level priors for learning manipulation skills with sparse features. In Dana Kulić, Yoshihiko Nakamura, Oussama Khatib, and Gentiane Venture, editors, *Proceedings of the 2017 International Symposium on Experimental Robotics*, pages 211–222, 2017.

[224] Norbert Krüger, Christopher Geib, Justus Piater, Ronald Petrick, Mark Steedman, Florentin Wörgötter, Aleš Ude, Tamim Asfour, Dirk Kraft, Damir Omrčen, Alejandro Agostini, and Rdiger Dillmann. Object-action complexes: Grounded abstractions of sensory-motor processes. *Robotics and Autonomous Systems*, 59(10):740 – 757, 2011. ISSN 0921-8890.

[225] J. Kulick, M. Toussaint, T. Lang, and M. Lopes. Active learning for teaching a robot grounded relational symbols. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 1451–1457, 2013.

[226] Johannes Kulick, Stefan Otte, and Marc Toussaint. Active exploration of joint dependency structures. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 2598–2604, 2015.

[227] V. Kumar, E. Todorov, and S. Levine. Optimal control with learned local models: Application to dexterous manipulation. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 378–383, 2016.

[228] VCV Kumar, S. Ha, and C.K. Liu. Expanding motor skills using relay networks. In *Proceedings of the 2nd Conference on Robot Learning*, pages 744–756, 2018.

[229] Andras Gabor Kupcsik, Marc Peter Deisenroth, Jan Peters, Gerhard Neumann, et al. Data-efficient generalization of robot skills with contextual policy search. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pages 1401–1407, 2013.

[230] Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart J Russell, and Pieter Abbeel. Learning plannable representations with causal InfoGAN. In *Advances in Neural Information Processing Systems 31*, pages 8733–8744, 2018.

[231] Tobias Lang, Marc Toussaint, and Kristian Kersting. Exploration in relational domains for model-based reinforcement learning. *Journal of Machine Learning Research*, 13 (1):3725–3768, 2012.

[232] Gilwoo Lee, Zita Marinho, Aaron M Johnson, Geoffrey J Gordon, Siddhartha S Srinivasa, and Matthew T Mason. Unsupervised learning for nonlinear piecewise smooth hybrid systems. *arXiv preprint arXiv:1710.00440*, 2017.

[233] Michelle A Lee, Yuke Zhu, Krishnan Srinivasan, Parth Shah, Silvio Savarese, Animesh Fei-Fei, Li andGarg, and Jeannette Bohg. Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, 2019.

[234] Enrico Di Lello, Markus Klotzbücher, Tinne De Laet, and Herman Bruyninckx. Bayesian time-series models for continuous fault detection and recognition in industrial robotic tasks. In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5827–5833, 2013.

[235] Ian Lenz, Ross A Knepper, and Ashutosh Saxena. Deepmpc: Learning deep latent features for model predictive control. In *Robotics: Science and Systems XI*, 2015.

[236] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[237] Sergey Levine and Vladlen Koltun. Guided policy search. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1–9, 2013.

[238] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with gaussian processes. In *Advances in Neural Information Processing Systems 24*, pages 19–27, 2011.

[239] Sergey Levine, Nolan Wagener, and Pieter Abbeel. Learning contact-rich manipulation skills with guided policy search. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 156–163, 2015.

[240] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[241] A. Levy, G.D. Konidaris, R. Platt, and K. Saenko. Learning multi-level hierarchies with hindsight. In *Proceedings of the 2019 International Conference on Learning Representations*, 2019.

[242] Y. Li, C. Chen, and P. K. Allen. Recognition of deformable object category and pose. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 5558–5564, 2014.

[243] Y. Li, Y. Yue, D. Xu, E. Grinspun, and P. K. Allen. Folding deformable objects using predictive simulation and trajectory optimization. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 6000–6006, 2015.

[244] Y. Li, Y. Wang, Y. Yue, D. Xu, M. Case, S. Chang, E. Grinspun, and P. K. Allen. Model-driven feedforward prediction for manipulation of deformable objects. *IEEE Transactions on Automation Science and Engineering*, 15(4): 1621–1638, 2018.

[245] Yinxiao Li, Xiuhan Hu, Danfei Xu, Yonghao Yue, Eitan Grinspun, and Peter K. Allen. Multi-sensor surface analysis for robotic ironing. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 5670–5676, 2016.

[246] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[247] Rudolf Lioutikov, Gerhard Neumann, Guilherme Maeda, and Jan Peters. Probabilistic segmentation applied to an assembly task. In *Proceedings of the 2015 IEEE-RAS International Conference on Humanoid Robots*, pages 533–540.

[248] Rudolf Lioutikov, Guilherme Maeda, Filipe Veiga, Kristian Kersting, and Jan Peters. Inducing probabilistic context-free grammars for the sequencing of movement primitives. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 1–8, 2018.

[249] Michael L. Littman and Richard S Sutton. Predictive representations of state. In *Advances in Neural Information Processing Systems 14*, pages 1555–1561. 2002.

[250] Michael L Littman, Ufuk Topcu, Jie Fu, Charles Isbell, Min Wen, and James MacGlashan. Environment-independent task specifications via GLTL. *arXiv preprint arXiv:1704.04341*, 2017.

[251] YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 1118–1125. IEEE, 2018.

[252] Manuel Lopes, Francisco Melo, and Luis Montesano. Active learning for reward estimation in inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 31–46, 2009.

[253] Qingkai Lu and Tucker Hermans. Modeling grasp type improves learning-based grasp planning. *IEEE Robotics and Automation Letters*, 4(2):784–791, 2019.

[254] Marianna Madry, Liefeng Bo, Danica Kragic, and Dieter Fox. ST-HMP: Unsupervised spatio-temporal feature learning for tactile data. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 2262–2269, 2014.

[255] Jeffrey Mahler, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James Kuffner, and Ken Goldberg. Dex-Net 1.0: A cloud-based network of 3D objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 1957–1964, 2016.

[256] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Robotics: Science and Systems XIII*, 2017.

[257] Shie Mannor, Reuven Y Rubinstein, and Yohai Gat. The cross entropy method for fast policy search. In *Proceedings of the 20th International Conference on Machine Learning*, pages 512–519, 2003.

[258] Roberto Martín-Martín, Sebastian Höfer, and Oliver Brock. An integrated approach to visual perception of articulated objects. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 5091 – 5097, 2016.

[259] Ruben Martinez-Cantin, Nando de Freitas, Arnaud Doucet, and José A Castellanos. Active policy learning for robot planning and exploration under uncertainty. In *Robotics: Science and Systems III*, pages 321–328, 2007.

[260] F. Meier, E. Theodorou, F. Stulp, and S. Schaal. Movement segmentation using a primitive library. In *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3407–3412, 2011.

[261] B. Michini and J.P. How. Bayesian nonparametric inverse reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163, 2012.

[262] H. Min, C. Yi, R. Luo, J. Zhu, and S. Bi. Affordance research in developmental robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems*, 8(4): 237–255, 2016.

[263] Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me Dave: Context sensitive grounding of natural language to mobile manipulation instructions. In *Robotics: Science and Systems X*, 2014.

[264] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529533, 2015.

[265] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1928–1937, 2016.

[266] Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems 28*, pages 2125–2133, 2015.

[267] Luis Montesano and Manuel Lopes. Active learning of visual descriptors for grasping using non-parametric smoothed beta distributions. *Robotics and Autonomous Systems*, 60(3):452–462, 2012.

[268] Luis Montesano, Manuel Lopes, Alexandre Bernardino, and José Santos-Victor. Learning object affordances: from sensory–motor coordination to imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

[269] Peter J Moylan and Brian DO Anderson. Nonlinear regulator theory and an inverse optimal control problem. In *Joint Automatic Control Conference*, number 9, pages 462–467, 1971.

[270] Carl Mueller, Jeff Venicx, and Bradley Hayes. Robust robot learning from demonstration and skill repair using conceptual constraints. In *Proceedings of the 2018*

IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 6029–6036, 2018.

[271] J. Mugan and B. Kuipers. Autonomous learning of high-level states and actions in continuous environments. *IEEE Transactions on Autonomous Mental Development*, 4(1):70–86, 2012.

[272] O. Nachum, S. Gu, H. Lee, and S. Levine. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems 31*, pages 3303–3313, 2018.

[273] Sanmit Narvekar, Jivko Sinapov, Matteo Leonetti, and Peter Stone. Source task creation for curriculum learning. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems*, pages 566–574, 2016.

[274] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the 16th International Conference on Machine Learning*, pages 278–287, 1999.

[275] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670, 2000.

[276] Hai Nguyen and Charles C. Kemp. Autonomously learning to visually detect where manipulation will succeed. *Autonomous Robots*, 36(1-2):137–152, 2014.

[277] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[278] S Niekum, S Osentoski, C G Atkeson, and A G Barto. Online Bayesian changepoint detection for articulated motion models. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 1468–1475, 2015.

[279] Scott Niekum and Andrew G Barto. Clustering via Dirichlet process mixture models for portable skill discovery. In *Advances in Neural Information Processing Systems 24*, pages 1818–1826, 2011.

[280] Scott Niekum, Andrew G Barto, and Lee Spector. Genetic programming for reward function search. *IEEE Transactions on Autonomous Mental Development*, 2(2):83–90, 2010.

[281] Scott Niekum, Sarah Osentoski, George Konidaris, Sachin Chitta, Bhaskara Marthi, and Andrew G Barto. Learning grounded finite-state representations from unstructured demonstrations. *International Journal of Robotics Research*, 34(2):131–157, 2015.

[282] S. Otte, J. Kulick, M. Toussaint, and O. Brock. Entropy-based strategies for physical exploration of the environment's degrees of freedom. In *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 615–622, 2014.

[283] Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1(6):1 – 14, 2009.

[284] Erhan Oztop, Nina S. Bradley, and Michael A. Arbib. Infant grasp learning: a computational model. *Experimental Brain Research*, 158(4):480–503, Oct 2004.

[285] Sindhu Padakandla, Shalabh Bhatnagar, et al. Reinforcement learning in non-stationary environments. *arXiv preprint arXiv:1905.03970*, 2019.

[286] Alexandros Paraschos, Christian Daniel, Jan R Peters, and Gerhard Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems 26*, pages 2616–2624, 2013.

[287] Daehyung Park, Zackory Erickson, Tapomayukh Bhattacharjee, and Charles C. Kemp. Multimodal execution monitoring for anomaly detection during robot manipulation. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 407–414, 2016.

[288] Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, pages 763–768, 2009.

[289] Peter Pastor, Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, and Stefan Schaal. Skill learning and task outcome prediction for manipulation. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 3828–3834, 2011.

[290] Peter Pastor, Ludovic Righetti, Mrinal Kalakrishnan, and Stefan Schaal. Online movement adaptation based on previous sensor experiences. In *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 365–371, 2011.

[291] Peter Pastor, Mrinal Kalakrishnan, Ludovic Righetti, and Stefan Schaal. Towards associative skill memories. In *Proceedings of the 2012 IEEE-RAS International Conference on Humanoid Robots*, pages 309–315, 2012.

[292] Deepak Pathak, Yide Shentu, Dian Chen, Pulkit Agrawal, Trevor Darrell, Sergey Levine, and Jitendra Malik. Learning instance segmentation by interaction. In *CVPR 2018 Workshop on Benchmarks for Deep Learning in Robotic Vision*, 2018.

[293] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[294] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, 2018.

[295] Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

[296] Jan Peters, Katharina Mülling, and Yasemin Altun. Relative entropy policy search. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 1607–1612, 2010.

[297] A. Petrovskaya and O. Khatib. Global localization of objects via touch. *IEEE Transactions on Robotics*, 27(3):569–585, 2011.

[298] Pedro Piacenza, Weipeng Dang, Emily Hannigan, Jeremy Espinal, Ikram Hussain, Ioannis Kymissis, and Matei T. Ciocarlie. Accurate contact localization and indentation depth prediction with an optics-based tactile sensor. In

Proceedings of the 2017 IEEE International Conference on Robotics and Automation, pages 959–965, 2017.

[299] Sudeep Pillai, Matthew R. Walter, and Seth Teller. Learning articulated motions from visual demonstrations. In *Robotics: Science and Systems X*, Berkeley, CA, 2014.

[300] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, pages 3406–3413, 2016.

[301] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. In *Proceedings of the 2016 European Conference on Computer Vision*, pages 3–18. Springer, 2016.

[302] R. Platt, C. Kohler, and M. Gualtieri. Deictic image mapping: An abstraction for learning pose invariant manipulation policies. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 2019.

[303] Robert Platt, Russ Tedrake, Leslie Kaelbling, and Tomas Lozano-Perez. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and Systems VI*, 2010.

[304] Robert Platt, Leslie Kaelbling, Tomas Lozano-Perez, and Russ Tedrake. Efficient planning in non-Gaussian belief spaces and its application to robot grasping. In *Proceedings of the 2011 International Symposium on Robotics Research*, pages 253–269, 2011.

[305] Vitchyr Pong, Shixiang Gu, Murtaza Dalal, and Sergey Levine. Temporal difference models: Model-free deep RL for model-based control. In *Proceedings of the 2018 International Conference on Learning Representations*, 2018.

[306] J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[307] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. *Urbana*, 51(61801):1–4, 2007.

[308] Muhammad Asif Rana, Mustafa Mukadam, S Reza Ahmadzadeh, Sonia Chernova, and Byron Boots. Skill generalization via inference-based planning. In *RSS 2017 Workshop on Mathematical Models, Algorithms, and Human-Robot Interaction*, 2017.

[309] P. Ranchod, B. Rosman, and G.D. Konidaris. Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 471–477, 2015.

[310] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*, pages 63–71. Springer, 2004.

[311] Nathan D. Ratliff, Jan Issac, Daniel Kappler, Stan Birchfield, and Dieter Fox. Riemannian motion policies. *CoRR*, abs/1801.02854, 2018.

[312] L. Riano and T.M. McGinnity. Automatically composing and parameterizing skills by evolving finite state automata. *Robotics and Autonomous Systems*, 60(4):639–650, 2012.

[313] Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Van de Wiele, Volodymyr Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing-solving sparse reward tasks from scratch. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4344–4353, 2018.

[314] A. Rodriguez, D. Bourne, M. Mason, G. F. Rossano, and J. Wang. Failure detection in assembly: Force signature analysis. In *Proceedings of the 2010 IEEE International Conference on Automation Science and Engineering*, pages 210–215, 2010.

[315] D. Rodriguez and S. Behnke. Transferring category-based functional grasping skills by latent space non-rigid registration. *IEEE Robotics and Automation Letters*, 3(3):2662–2669, 2018.

[316] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

[317] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[318] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[319] Dorsa Sadigh, Eric S Kim, Samuel Coogan, S Shankar Sastry, and Sanjit A Seshia. A learning based approach to control synthesis of Markov decision processes for linear temporal logic specifications. In *Proceedings of the 53rd Annual Conference on Decision and Control*, pages 1091–1096, 2014.

[320] Erol Sahin, Maya Cakmak, Mehmet R. Dogar, Emre Ugur, and Gokturk Ucoluk. To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior*, 15(4):447–472, 2007.

[321] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4470–4479, 2018.

[322] Terence D Sanger. Neural network learning control of robot manipulators using gradually increasing task difficulty. *IEEE Transactions on Robotics and Automation*, 10 (3):323–333, 1994.

[323] Akanksha Saran, Branka Lakic, Srinjoy Majumdar, Juergen Hess, and Scott Niekum. Viewpoint selection for visual failure detection. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5437–5444, 2017.

[324] Ashutosh Saxena, Justin Driemeyer, Justin Kearns, Chioma Osondu, and Andrew Y. Ng. Learning to grasp novel objects using vision. In Oussama Khatib, Vijay Kumar, and Daniela Rus, editors, *Proceedings of the 2008 International Symposium on Experimental Robotics*, pages 33–42, 2008.

[325] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

[326] Stefan Schaal. Dynamic movement primitives—a framework for motor control in humans and humanoid robotics. In *Adaptive Motion of Animals and Machines*, pages 261–280. Springer, 2006.

[327] Stefan Schaal, Jan Peters, Jun Nakanishi, and Auke Ijspeert. Learning movement primitives. In *Proceedings of the 2005 International Symposium on Robotics Research*, pages 561–572. Springer, 2005.

[328] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1312–1320, 2015.

[329] C. Schenck, J. Sinapov, and A. Stoytchev. Which object comes next? grounded order completion by a humanoid robot. *Cybernetics and Information Technologies*, 12(3): 5–16, 2012.

[330] Connor Schenck and Dieter Fox. Perceiving and reasoning about liquids using fully convolutional networks. *International Journal of Robotics Research*, 37(4-5):452–471, 2018.

[331] Connor Schenck and Dieter Fox. SPNets: Differentiable fluid dynamics for deep neural networks. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 317–335, 2018.

[332] Connor Schenck and Alexander Stoytchev. The object pairing and matching task: Toward Montessori tests for robots. In *Humanoids 2012 Workshop on Developmental Robotics*, 2012.

[333] Connor Schenck, Jivko Sinapov, David Johnston, and Alexander Stoytchev. Which object fits best? Solving matrix completion tasks with a humanoid robot. *IEEE Transactions on Autonomous Mental Development*, 6(3): 226–240, 2014.

[334] Connor Schenck, Jonathan Tompson, Sergey Levine, and Dieter Fox. Learning robotic manipulation of granular media. In *Proceedings of the 1st Conference on Robot Learning*, pages 239–248, 2017.

[335] Tanner Schmidt, Richard A. Newcombe, and Dieter Fox. DART: dense articulated real-time tracking with consumer depth cameras. *Autonomous Robots*, 39(3):239–258, 2015.

[336] Jonathan Scholz and Mike Stilman. Combining motion planning and optimization for flexible robot manipulation. In *Proceedings of the 2010 IEEE-RAS International Conference on Humanoid Robots*, pages 80–85, 2010.

[337] Jonathan Scholz, Martin Levihn, Charles Isbell, and David Wingate. A physics-based model prior for object-oriented MDPs. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1089–1097, 2014.

[338] John Schulman, Jonathan Ho, Cameron Lee, and Pieter Abbeel. Learning from demonstrations through the use of non-rigid registration. In *Proceedings of the 2013 International Symposium on Robotics Research*, pages 339–354. Springer, 2013.

[339] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1889–1897, 2015.

[340] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[341] Max Schwarz, Anton Milan, Arul Selvam Periyasamy, and Sven Behnke. RGB-D object detection and semantic segmentation for autonomous manipulation in clutter. *International Journal of Robotics Research*, 37(4-5):437–451, 2018.

[342] Daniel Seita, Nawid Jamali, Michael Laskey, Ron Berenstein, Ajay Kumar Tanwani, Prakash Baskaran, Soshi Iba, John Canny, and Ken Goldberg. Robot bed-making: Deep transfer learning using depth sensing of deformable fabric. *arXiv preprint arXiv:1809.09810*, 2018.

[343] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 1134–1141, 2018.

[344] Stuart C. Shapiro and Haythem O. Ismail. Anchoring in a grounded layered architecture with integrated reasoning. *Robotics and Autonomous Systems*, 43(2):97 – 108, 2003.

[345] Thierry Siméon, Jean-Paul Laumond, Juan Cortés, and Anis Sahbani. Manipulation planning with probabilistic roadmaps. *International Journal of Robotics Research*, 23 (7-8):729–746, 2004.

[346] J. Sinapov, V. Sukhoy, R. Sahai, and A. Stoytchev. Vibrotactile recognition and categorization of surfaces by a humanoid robot. *IEEE Transactions on Robotics*, 27(3): 488–497, 2011.

[347] Jivko Sinapov, Taylor Bergquist, Connor Schenck, Ugonna Ohiri, Shane Griffith, and Alexander Stoytchev. Interactive object recognition using proprioceptive and auditory feedback. *International Journal of Robotics Research*, 30(10): 1250–1262, 2011.

[348] Jivko Sinapov, Connor Schenck, Kerrick Staley, Vladimir Sukhoy, and Alexander Stoytchev. Grounding semantic categories in behavioral interactions: Experiments with 100 objects. *Robotics and Autonomous Systems*, 62(5): 632–645, 2014.

[349] Satinder Singh, Michael R. James, and Matthew R. Rudary. Predictive state representations: A new theory for modeling dynamical systems. In *Proceedings of the 20th International Conference on Uncertainty in Artificial Intelligence*, pages 512–519, 2004.

[350] D. Song, C. H. Ek, K. Huebner, and D. Kragic. Multivariate discretization for Bayesian network structure learning in robot grasping. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 1944–1950, 2011.

[351] Jonathan Sorg, Richard L Lewis, and Satinder P Singh. Reward design via online gradient ascent. In *Advances in Neural Information Processing Systems 23*, pages 2190–2198, 2010.

[352] Jonathan Sorg, Satinder P Singh, and Richard L Lewis. Internal rewards mitigate agent boundedness. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1007–1014, 2010.

[353] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4732–4741, 2018.

[354] Bradly C Stadie, Pieter Abbeel, and Ilya Sutskever. Third-person imitation learning. In *Proceedings of the 2017 International Conference on Learning Representations*, 2017.

[355] Kenneth O Stanley, David B D'Ambrosio, and Jason Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212, 2009.

[356] Mike Stilman, Koichi Nishiwaki, and Satoshi Kagami. Learning object models for whole body manipulation. In *Proceedings of the 2008 IEEE-RAS International Conference on Humanoid Robots*, pages 174–179, 2008.

[357] Johannes A. Stork, Carl Henrik Ek, Yasemin Bekiroglu, and Danica Kragic. Learning predictive state representation for in-hand manipulation. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 3207–3214, 2015.

[358] Alexander Stoytchev. Behavior-grounded representation of tool affordances. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3060–3065, 2005.

[359] F. Stramandinoli, V. Tikhanoff, U. Pattacini, and F. Nori. Heteroscedastic regression and active learning for modeling affordances in humanoids. *IEEE Transactions on Cognitive and Developmental Systems*, 10(2):455–468, 2018.

[360] Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. PAC model-free reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 881–888, 2006.

[361] J. Stückler and S. Behnke. Adaptive tool-use strategies for anthropomorphic service robots. In *Proceedings of the 2014 IEEE-RAS International Conference on Humanoid Robots*, pages 755–760, 2014.

[362] Jürgen Sturm, Advait Jain, Cyrill Stachniss, Charles C Kemp, and Wolfram Burgard. Operating articulated objects based on experience. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2739–2744, 2010.

[363] Jürgen Sturm, Cyrill Stachniss, and Wolfram Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research*, 41(2):477–526, 2011.

[364] Zhe Su, Karol Hausman, Yevgen Chebotar, Artem Molchanov, Gerald E Loeb, Gaurav S Sukhatme, and Stefan Schaal. Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor. In *Proceedings of the 2015 IEEE-RAS International Conference on Humanoid Robots*, pages 297–303, 2015.

[365] Zhe Su, Oliver Kroemer, Gerald E. Loeb, Gaurav S. Sukhatme, and Stefan Schaal. Learning to switch between sensorimotor primitives using multimodal haptic signals. In *From Animals to Animats 14*, pages 170–182, 2016.

[366] Vladimir Sukhoy, Veselin Georgiev, Todd Wegter, Ramy Sweidan, and Alexander Stoytchev. Learning to slide a magnetic card through a card reader. In *Proceedings of the 2012 IEEE International Conference on Robotics and Automation*, pages 2398–2404, 2012.

[367] Balakumar Sundaralingam, Alexander Lambert, Ankur Handa, Byron Boots, Tucker Hermans, Stan Birchfield, Nathan Ratliff, and Dieter Fox. Robust learning of tactile force estimation through robot interaction. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, 2019.

[368] Jaeyong Sung and J. Kenneth Salisbury andAshutosh Saxena. Learning to represent haptic feedback for partially-observable tasks. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, pages 2802–2809, 2017.

[369] Jaeyong Sung, Seok Hyun Jin, and Ashutosh Saxena. Robobarista: Object part-based transfer of manipulation trajectories from crowd-sourcing in 3D pointclouds. In *Proceedings of the 2015 International Symposium on Robotics Research*, 2015.

[370] Jaeyong Sung, Ian Lenz, and Ashutosh Saxena. Deep multimodal embedding: Manipulating novel objects with point-clouds, language and trajectories. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, pages 2794–2801, 2017.

[371] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 1998.

[372] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 13*, pages 1057–1063, 2000.

[373] R.S. Sutton, D. Precup, and S.P. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2): 181–211, 1999.

[374] Maxwell Svetlik, Matteo Leonetti, Jivko Sinapov, Rishi Shah, Nick Walker, and Peter Stone. Automatic curriculum graph generation for reinforcement learning agents. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2590–2596, 2017.

[375] Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems 29*, pages 2154–2162, 2016.

[376] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *Proceedings of the 2014 IEEE International Conference on Robotics and Automation*, pages 1168–1175, 2014.

[377] Matthew Taylor, Peter Stone, and Yaxin Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(1): 2125–2167, 2007.

[378] Matthew E Taylor, Halit Bener Suay, and Sonia Chernova. Integrating reinforcement learning with human demonstrations of varying ability. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 617–624, 2011.

[379] Russ Tedrake, Ian R Manchester, Mark Tobenkin, and John W Roberts. LQR-trees: Feedback motion planning via sums-of-squares verification. *The International Journal of Robotics Research*, 29(8):1038–1052, 2010.

[380] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands

for robotic navigation and mobile manipulation. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pages 1507–1514, 2011.

[381] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. Grasp pose detection in point clouds. *International Journal of Robotics Research*, 36(13-14):1455–1473, 2017.

[382] Moritz Tenorth, Stefan Profanter, Ferenc Balint-Benczedi, and Michael Beetz. Decomposing CAD models of objects of daily use and reasoning about their functional parts. In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5943–5949, 2013.

[383] Brijen Thananjeyan, Ashwin Balakrishna, Ugo Rosolia, Felix Li, Rowan McAllister, Joseph E Gonzalez, Sergey Levine, Francesco Borrelli, and Ken Goldberg. Extending deep model predictive control with safety augmented value estimation from demonstrations. *arXiv preprint arXiv:1905.13402*, 2019.

[384] Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, 11 (1):3137–3181, 2010.

[385] Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2139–2148, 2016.

[386] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High confidence policy improvement. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2380–2388, 2015.

[387] Philip S Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 3000–3006, 2015.

[388] Jesse Thomason, Jivko Sinapov, Maxwell Svetlik, Peter Stone, and Raymond J Mooney. Learning multi-modal grounded linguistic semantics by playing "I Spy". In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 3477–3483, 2016.

[389] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 23–30, 2017.

[390] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4950–4957, 2018.

[391] Marc Toussaint, Kelsey Allen, Kevin A. Smith, and Joshua B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. In *Robotics: Science and Systems XIV*, 2018.

[392] Jonathan Tremblay, Thang To, Balakumar Sundaralingam, Yu Xiang, Dieter Fox, and Stanley T. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. In *Proceedings of the 2nd Conference on Robot Learning*, pages 306–316, 2018.

[393] Constantine J. Tsikos and Ruzena Bajcsy. Segmentation via manipulation. *IEEE Transactions on Robotics and Automation*, 7(3):306–319, 1991.

[394] E. Ugur and J. Piater. Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 2627–2633, 2015.

[395] E. Ugur and J. Piater. Refining discovered symbols with multi-step interaction experience. In *Proceedings of the 2015 IEEE-RAS International Conference on Humanoid Robots*, pages 1007–1012, 2015.

[396] Emre Ugur, Erol Sahin, and Erhan Öztop. Predicting future object states using learned affordances. In *Proceedings of the 24th International Symposium on Computer and Information Sciences*, pages 415–419, 2009.

[397] Emre Ugur, Erhan Öztop, and Erol Sahin. Going beyond the perception of affordances: Learning how to actualize them through behavioral parameters. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pages 4768–4773, 2011.

[398] Jur Van Den Berg, Stephen Miller, Daniel Duckworth, Humphrey Hu, Andrew Wan, Xiao-Yu Fu, Ken Goldberg, and Pieter Abbeel. Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pages 2074–2081, 2010.

[399] Herke van Hoof, Oliver Kroemer, and Jan Peters. Probabilistic segmentation and targeted exploration of objects in cluttered environments. *IEEE Transactions on Robotics*, 30(5):1198–1209, 2014.

[400] H. van Seijen, S. Whiteson, and L.J.H.M Kester. Efficient abstraction selection in reinforcement learning. *Computational Intelligence*, 30(4):657–699, 2013.

[401] Jacob Varley, David Watkins-Valls, and Peter K. Allen. Multi-modal geometric learning for grasping and manipulation. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, 2019.

[402] F. Veiga, J. Peters, and T. Hermans. Grip stabilization of novel objects using slip prediction. *IEEE Transactions on Haptics*, 11(4):531–542, 2018.

[403] F.F. Veiga, H. van Hoof, J. Peters, and T. Hermans. Stabilizing novel objects by learning to predict tactile slip. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5065–5072, 2015.

[404] A. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, , and K. Kavukcuoglu. FeUdal networks for hierarchical reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3540–3549, 2017.

[405] N.A. Vien and M. Toussaint. Touch-based POMDP manipulation via sequential submodular optimization. In *Proceedings of the 2015 IEEE-RAS International Conference on Humanoid Robots*, pages 407–413, 2015.

[406] N.A. Vien and M. Toussaint. POMDP manipulation via trajectory optimization. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 242–249, 2015.

[407] Sethu Vijayakumar and Stefan Schaal. Locally weighted projection regression: An $O(n)$ algorithm for incremental real time learning in high dimensional space. In *Proceedings of the 17th International Conference on Machine Learning*, pages 288–293, 2000.

[408] Francisco E Viña, Yasemin Bekiroglu, Christian Smith, Yiannis Karayiannidis, and Danica Kragic. Predicting slippage and learning manipulation affordances through Gaussian process regression. In *Proceedings of the 2013 IEEE-RAS International Conference on Humanoid Robots*, pages 462–468, 2013.

[409] R. Waibel, M. and Beetz, M. and Civera, J. and D'Andrea, R. and Elfring, J. and Galvez-Lopez, D. and Haussermann, K. and Janssen, R. and Montiel, J.M.M. and Perzylo, A. and Schiessle, B. and Tenorth, M. and Zweigle, O. and van de Molengraft. Roboearth—a world wide web for robots. *IEEE Robotics Automation Magazine*, 18(2):69–82, 2011.

[410] Austin S. Wang and Oliver Kroemer. Learning robust manipulation strategies with multimodal state transition models and recovery heuristics. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, 2019.

[411] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the 2019 Conference on Computer Vision and Pattern Recognition*, 2019.

[412] Zi Wang, Caelan Reed Garrett, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Active model learning and diverse action sampling for task and motion planning. In *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4107–4114, 2018.

[413] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay. In *Proceedings of the 2017 International Conference on Learning Representations*, 2017.

[414] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[415] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, 291(5504):599–600, 2001.

[416] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[417] Florentin Wörgotter, Eren Erdal Aksoy, Norbert Kruger, Justus Piater, Ales Ude, and Minija Tamosiunaite. A simple ontology of manipulation actions based on hand-object relations. *IEEE Transactions on Autonomous Mental Development*, 5(2):117–134, 2013. ISSN 1943-0604.

[418] Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in Neural Information Processing Systems 28*, pages 127–135, 2015.

[419] Jiajun Wu, Joseph J Lim, Hongyi Zhang, Joshua B Tenenbaum, and William T Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *Proceedings of the 2016 British Machine Vision Conference*, 2016.

[420] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation. In *Advances in Neural Information Processing Systems 30*, pages 5279–5288, 2017.

[421] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[422] M. Wüthrich, J. Bohg, D. Kappler, C. Pfreundt, and S. Schaal. The coordinate particle filter - a novel particle filter for high dimensional systems. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 2454–2461, 2015.

[423] Annie Xie, Avi Singh, Sergey Levine, and Chelsea Finn. Few-shot goal inference for visuomotor learning and planning. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 40–52, 2018.

[424] Danfei Xu, Suraj Nair, Yuke Zhu, Julian Gao, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Neural task programming: Learning to generalize across hierarchical tasks. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 1–8, 2018.

[425] Tianbing Xu, Qiang Liu, Liang Zhao, Wei Xu, and Jian Peng. Learning to explore via meta-policy gradient. In *Proceedings of the 35th International Conference on Machine Learning*, pages 5463–5472, 2018.

[426] A. Yamaguchi and C. G. Atkeson. Stereo vision of liquid and particle flow for robot pouring. In *Proceedings of the 2016 IEEE-RAS International Conference on Humanoid Robots*, pages 1173–1180, 2016.

[427] A. Yamaguchi and C. G. Atkeson. Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables. In *Proceedings of the 2016 IEEE-RAS International Conference on Humanoid Robots*, pages 1045–1051, 2016.

[428] A. Yamaguchi and C. G. Atkeson. Implementing tactile behaviors using fingervision. In *Proceedings of the 2017 IEEE-RAS International Conference on Humanoid Robots*, pages 241–248, 2017.

[429] Akihiko Yamaguchi and Christopher G Atkeson. Differential dynamic programming with temporally decomposed dynamics. In *Proceedings of the 2015 IEEE-RAS International Conference on Humanoid Robots*, pages 696–703, 2015.

[430] Yezhou Yang, Yi Li, Cornelia Fermuller, and Yiannis Aloimonos. Robot learning manipulation action plans by "watching" unconstrained videos from the world wide web. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 3686–3692, 2015.

[431] Tian Ye, Xiaolong Wang, James Davidson, and Abhinav Gupta. Interpretable intuitive physics model. In *Computer Vision - ECCV 2018*, pages 89–105, 2018.

[432] Kuan-Ting Yu, Maria Bauzá, Nima Fazeli, and Alberto Rodriguez. More than a million ways to be pushed. A high-fidelity experimental dataset of planar pushing. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 30–37, 2016.

[433] Tianhe Yu, Pieter Abbeel, Sergey Levine, and Chelsea Finn. One-shot hierarchical imitation learning of compound visuomotor tasks. *arXiv preprint arXiv:1810.11043*, 2018.

[434] Zhen Zeng, Zheming Zhou, Zhiqiang Sui, and Odest Chadwicke Jenkins. Semantic robot programming for goal-directed manipulation in cluttered scenes. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 7462–7469, 2018.

[435] Fangyi Zhang, Jürgen Leitner, Michael Milford, Ben Upcroft, and Peter Corke. Towards vision-based deep reinforcement learning for robotic motion control. *arXiv preprint arXiv:1511.03791*, 2015.

[436] Tianhao Zhang, Zoe McCarthy, Owen Jowl, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, pages 1–8, 2018.

[437] Kemin Zhou, John Comstock Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice Hall, 1996.

[438] Yilun Zhou, Benjamin Burchfiel, and George Konidaris. Representing, learning, and controlling complex object interactions. *Autonomous Robots*, 42(7):1355–1367, 2018.

[439] Brian D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, 2010.

[440] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 1433–1438, 2008.