Re-Using Schematic Grasping Policies

Robert Platt, Roderic A. Grupen
Laboratory for Perceptual Robotics
Department of Computer Science
University of Massachusetts, Amherst
{rplatt, grupen}@cs.umass.edu

Andrew H. Fagg
Symbiotic Computing Lab
School of Computer Science
University of Oklahoma
fagg@ou.edu

***Abstract*— It can be difficult to generalize the solutions to grasping and manipulation problems because even small differences in problem context can require qualitatively different solutions. For example, small changes in the shape of an object to be grasped can necessitate different grasp strategies. In this paper, we introduce the *action schema* framework that represents generalized skills in a functional way such that all viable ways of accomplishing a task are represented as instantiations of the generalized skill. We also propose an on-line algorithm for learning how to instantiate the skill in a context-appropriate way. We test this approach with a robotic grocery bagging task where a dexterous humanoid robot learns to make correct qualitative decisions regarding how to grasp everyday grocery items and drop them in a paper bag.**

I. Introduction

A key problem for robotic grasping systems is that it is difficult to generalize a single grasp solution to all the different grasp scenarios a robot is likely to encounter in an open environment. This is because relatively similar grasping problems often require kinematically different solutions. For example, small differences in object shape can have a big effect on how many fingers the grasp should use. Similarly, small differences in object location have significant bearing on whether a humanoid robot should use its left or right hand and whether an over-hand or under-hand grasp is more appropriate. A robot must be able to evaluate these different grasp strategies as equally valid alternatives, and be able to select the correct strategy in any given situation. In this paper, we propose a control-based framework that corresponds functionally similar behaviors with a single generalized solution. We propose an algorithm that learns instantiations of the generalized solution that are appropriate in different problem contexts.

Humans select grasps from a large repertoire based primarily on object characteristics and task requirements. With respect to object characteristics, grasp selection is related to object shape, size, and weight. For example, Schlesinger identifies a grasp posture known as "spherical prehension" for grasping cylinders and "hook prehension" for grasping heavier objects [1]. Cutkosky enumerates sixteen different types of power and precision grasps and associates them with different objects and tasks [2].

The close relationship between human grasps and object and task characteristics suggests an approach to automated robot grasping whereby the robot assesses object and task characteristics and responds by producing the corresponding grasp. Using this approach, Cutkosky and Howe proposed an expert system that associated desired grasp characteristics with a grasp label based on human grasp associations [3]. In an attempt to develop an association that generalizes to new objects, Iberall proposed using a neural network to learn the association between object and grasp type and an appropriate selection of palmar or (finger) pad opposition [4].

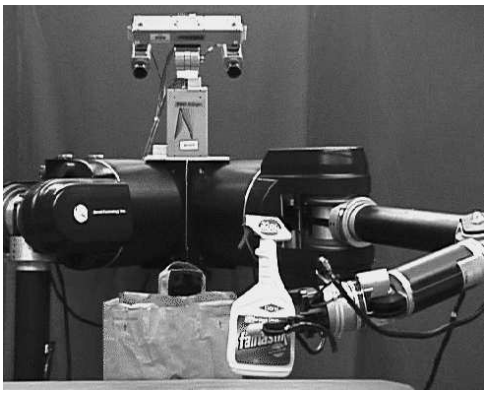


Fig. 1. Dexter prepares to drop a spray bottle containing multi-purpose cleaning solution into a paper grocery bag.

Elements of this approach to grasping have been tested in the context of real robot experiments. Kamon, Flash, and Edelman proposed approximating the relationship between grasp parameters and resulting grasp quality using a nearest-neighbor approach [5]. Moussa proposed a similar approach where the system learns an object-centric homogeneous transform that correctly positions the gripper based on trial-and-error experience [6]. Both of these approaches make all grasping decisions in a single step: qualitative grasp decisions are made by the same mechanism that resolves the fine details of contact placement. In addition, both of these approaches use vision as their only sensory modality; it is not clear how non-visual information could be integrated into these systems.

In this paper, we present an approach to learning to apply general skills in new situations known as the *action schema* framework, and apply it to the grasping problem. The action schema specifies generalized behaviors in terms of abstract actions derived from "natural" classes of controllers. Because each abstract action is associated with a set of different controllers, the generalized behavior expressed by the action

schema can be instantiated in many different ways. Using an on-line algorithm, the system learns to associate problem context with the corresponding instantiation. We demonstrate this approach in a grocery bagging task where a humanoid robot reaches to and grasps various grocery items and drops them into a paper bag. The system learns to associate general visual features such as blob height and width with correct instantiations of a localize-reach-grasp action schema. This approach decomposes the problem into a high-level part where qualitatively correct actions are learned and a low-level part where closed-loop controllers specialize in solving sub-tasks.

This paper is organized as follows. Section II reviews a grasp control technique and the control basis approach. Section III proposes the action schema framework for learning context appropriate instantiations of generalized grasping behaviors. Last, Section IV describes experimental work that demonstrates the utility of this approach.

II. Background

This paper is predicated on a control-based approach whereby closed-loop controllers are combined concurrently and sequentially to generate and represent robot behavior. We draw upon previous work in the areas of grasp control and the control basis [7]–[11].

A. Grasp Control

In the control-based approach to grasp synthesis, a grasp controller displaces contacts toward better grasp geometries as part of a closed-loop process. These approaches use tactile feeback to calculate an error gradient and displace grasp contacts on the object surface. After making light contact with the object using sensitive tactile load cells, the controller displaces contacts toward minima in the grasp error function using discrete probes [7] or a continuous sliding motion [10].

Grasp controllers descend an artificial potential ϕ_g derived from wrench error

$$\epsilon_w = \vec{\rho}^T \vec{\rho}, \quad \vec{\rho} = \sum_{1 \leq i \leq n} \vec{w}_i \tag{1}$$

where $\vec{w}_i$ is the contact wrench at the i^{th} contact calculated using the point contact without friction model [7], [8]. The control law converges when the contacts have been displaced to locations where the net applied wrench is minimized. If the minimum corresponds to zero net wrench, then, in the presence of friction, such a grasp achieves necessary conditions for wrench closure because it fulfills the conditions for non-marginal equilibrium. Non-marginal equilibrium requires the contact forces achieving net zero force lie strictly inside their corresponding friction cones and has been shown to be a sufficient condition for wrench closure [12]. This is equivalent to the grip Jacobian having a non-trivial nullspace.

B. The Control Basis Approach

When using a control-based approach to solve multi-step tasks, a framework is needed that allows controllers to be sequenced in an organized way. The *control basis* framework accomplishes this by organizing the set of viable controllers and providing a robust way of evaluating system state [11].

1) Controller Synthesis: The control basis can systematically specify an arbitrary closed-loop controller by matching an artificial potential function with a *sensor transform* and *effector transform*. The potential function specifies controller objectives, the effector transform specifies what degrees of freedom the controller uses, and the sensor transform implements the controller feedback loop. For example, consider a reach controller. The sensor transform specifies the goal configuration of the end-effector. The effector transform specifes what degrees of freedom are used to accomplish the task.

In general, the control basis realizes a complete controller by selecting one potential function from a set $\Phi = \{\phi_1, \phi_2, \ldots\}$, one sensor transform from a set $\Sigma = \{\sigma_1, \sigma_2, \ldots\}$, and one effector transform from a set $\Upsilon = \{\tau_1, \tau_2, \ldots\}$. Given Φ, Σ, and Υ, the set of controllers that may be generated is $\Pi \subseteq \Phi \times \Sigma \times \Upsilon$. When specifying a fully-instantiated controller, the notation $\phi_i|_\tau^\sigma$ denotes the controller constructed by parameterizing potential function ϕ_i with sensor transform σ and effector transform τ.

In this paper, we will utilize four reach controllers that reach to the top or the side of an object using either the left or the right hand. The set of sensor transforms compatible with the reach potential function ϕ_r is $\Sigma = \{top, side\}$. The set of compatible effector transforms is $\Upsilon = \{lh, rh\}$ (standing for "left hand" and "right hand."). Therefore the following reach controllers are possible: $\phi_r|_{lh}^{top}$ (reach to the object's top with the left hand), $\phi_r|_{rh}^{top}$ (reach to the object's top with the right hand), $\phi_r|_{lh}^{side}$ (reach to the object's side with the left hand), and $\phi_r|_{rh}^{side}$ (reach to the object's side with the right hand).

We will also utilize four grasp controllers that grasp an object using either two or three fingers on either the left or right hands. The set of sensor transforms is $\Sigma = \{l12, l123, r12, r123\}$; the elements of this set stand for "left hand, fingers one and two," "left hand, fingers one, two, and three," "right hand, fingers one and two," etc. Assuming that each of the grasping sensors is actuated, there is a correspondance between sensor and effector transforms: $\Upsilon = \{l12, l123, r12, r123\}$. After eliminating instantiations of ϕ_{grasp} that do not make sense, the four possible grasp controllers are: $\phi_g|_{l12}^{l12}$ (grasp with two fingers on the left hand), $\phi_g|_{l123}^{l123}$ (grasp with three fingers on the left hand), $\phi_g|_{l12}^{l12}$ (grasp with two fingers on the right hand), and $\phi_g|_{l123}^{l123}$ (grasp with three fingers on the right hand).

2) Discrete Controller State: The control basis approach measures system state in terms of controller dynamics. At any point in time, the instantaneous error and the instantaneous gradient of error can be evaluated. Although the more general system dynamics can be treated [13], in this paper, we will consider only controller convergence (convergence is characterized by a low error and error gradient) to establish system state. For example, the state of having grasped an object with some effector is represented by the convergence status of a grasp controller parameterized by that effector transform.

Controller error is calculated by evaluating the controller's

potential function ϕ for a particular sensor transform σ. Let $\mathcal{R}$ be the set of compatible potential functions and sensors: $\mathcal{R} \subseteq \Phi \times \Sigma$. Then system state can be defined to be the elements of $\mathcal{R}$ that are converged:

$$s_k = \{(\phi_i, \sigma_j) \in \mathcal{R} | \phi_i \text{ is converged for } \sigma_j \text{ with low error}\}. \quad (2)$$

The set of all states that can be represented this way is the power set $2^{\mathcal{R}}$. For example, if the system is in state $s_k \subseteq \mathcal{R}$, then $(\phi_i, \sigma_j) \in s_k$ when ϕ_i is converged for σ_j with a low error. If $(\phi_i, \sigma_j) \notin s_k$, then either ϕ_i is not converged for σ_j, or it converged with a high error.

III. Generalized Grasping Policies

Successful grasping requires the robot to be able to handle a wide variety of slightly different problem contexts such as differences in object size, shape, and location. A key difficulty is that similar grasping problems can require relatively different grasping solutions. For example, a tall object might be grasped from the side while a shorter object might be better grasped from the top. Rectangular objects are probably best grasped by opposing two fingers while round object may be better grasped with three or more fingers. Whether the robot grasps using the left or right hand should depend on the location of the object. While these grasp problems differ only in degrees, they can require the robot to behave in geometrically different ways. In order to solve grasping problems like this, it is insufficient to adapt a solution to different problem contexts by just making small tweaks to end-effector trajectories. Instead, the focus should be on developing a representation of *functionally* similar grasping policies that generalize well.

A. Equivalence Classes of States and Actions

Our representation of generalized behavior is based on equivalence classes of actions and states. We introduce *abstract actions* and *abstract states* that roughly correspond to the notion of generalized states and actions and develop a mapping from functionally similar controllers to abstract actions and from functionally similar states to abstract states. These two mappings will allow us to represent the set of functionally similar behaviors.

By specifying controllers in terms of the triple (ϕ, σ, τ), the control basis approach can be considered to factor the control objective, (ϕ), from sensor and effector parameterizations to create a partition on Π. Controllers that share the same potential function are therefore in the same equivalence class.

Recall that the set of possible controllers Π is a subset of $\Phi \times \Sigma \times \Upsilon$. The function

$$g : \Pi \to \Phi, \quad g((\phi_i, \sigma_j, \tau_k)) = \phi_i \quad (3)$$

maps the space of controllers into the space of potential functions by ignoring sensor and effector parameterization. For example, consider two reach controllers: $\phi_r|_{lh}^{top}$ and $\phi_r|_{rh}^{side}$ (where lh is short for "left hand" and rh is short for "right hand.") These two controllers are in the same class because they both share the reach potential function ϕ_r. We also define g^{-1} to be the inverse of g:

$$g^{-1}(a') = \{a \in A | g(a) = a'\}. \quad (4)$$

The partition on controllers induced by g immediately suggests a set of abstract actions

$$A' \subseteq \Phi. \quad (5)$$

Assuming that any controller that can be executed is considered an action, $A \subseteq \Pi$, then Equation 3 can be described as $g : A \to A'$. This equation maps the *underlying actions* A onto abstract actions A'.

Controller equivalence classes induce equivalence classes of states. A state is defined to be the set of pairs $(\phi_i, \sigma_j) \in \mathcal{R}$ that are converged. Two states are considered to be in the same class when the system is converged for the same potential functions in both states. This is defined formally by mapping states $S \subseteq 2^{\mathcal{R}}$ onto a set of abstract states

$$S' \subseteq 2^{\Phi} \quad (6)$$

as follows:

$$f : S \to S', \quad (7)$$
$$f(s_k) = \{\phi_i \in \Phi | \exists \sigma_j \in \Sigma \text{ s.t. } (\phi_i, \sigma_j) \in s_k\}.$$

This equation maps *underlying states* S onto abstract states S'. For example, two underlying states $s_1 = \{(\phi_1, \sigma_1), (\phi_2, \sigma_2)\}$ and $s_2 = \{(\phi_1, \sigma_2), (\phi_2, \sigma_3)\}$ are in the same class because $f(s_1) = f(s_2) = \{\phi_1, \phi_2\}$. The abstract state $\{\phi_1, \phi_2\}$ describes the condition that ϕ_1 is currently converged *for some* sensor transform and ϕ_2 is also converged for some (possibly different) sensor transform.

B. An action schema

Abstract states and actions are used to represent functionally similar behaviors. We introduce the *action schema* framework that encodes a generalized behavior that can be mapped onto a set of functionally similar behaviors.

The action schema is defined in terms of a set of abstract states $S' \subseteq 2^{\Phi}$, abstract actions $A' \subseteq \Phi$, an abstract policy π', an abstract transition function T', and an abstract goal state s'_g. It is defined formally as the tuple:

$$\mathcal{S} = \left\langle S', A', \pi', T', s'_g \right\rangle. \quad (8)$$

The abstract policy $\pi' : S' \to A'$ is a rule for selecting the abstract action to execute given the current abstract state (notice that π' is defined differently than Π). This policy is assumed to be deterministic. It may be hand-coded, it may come from an observation of a successful teleoperated sequence, or it may derive from an autonomously learned policy. The deterministic transition function $T' : S' \times A' \to S'$, $T'(s'_t, a') = s'_{t+1}$ specifies what abstract state s'_{t+1} the schema expects to arrive in when a' is taken from s'_t. The goal abstract state $s'_g \in S'$ encodes the objective of the policy.

The right side of Figure 2 illustrates an action schema that represents a generalized grasping behavior. The schema policy

and transition function specify that (starting from abstract state s'_1) the visual localize abstract action, $\phi_{localize}$, is to be taken followed by the abstract reach action, ϕ_{reach}, and the abstract grasp action, ϕ_{grasp}. The four circles represent abstract states $s'_1, s'_2, s'_3, s'_4 \in S'$ where s'_4 is the goal state and the arrows represent abstract actions.

C. Instantiations of the action schema

The generalized behavior encoded by the action schema can be instantiated by multiple functionally similar behaviors. This subsection describes an approach to instantiating generalized behavior whereby the system first determines which abstract action is supposed to execute next and then enumerates all the functionally similar instantiations of that action.

Recall that $S' \times A'$ is the *abstract space* and $S \times A$ (where $A \subseteq \Pi$ and $S \subseteq 2^{\mathcal{R}}$) is the *underlying* space. Generalized behavior encoded by the abstract policy π' maps to a set of policies $\pi(s_t) \in B(s_t)$ in the underlying space where

$$B(s_t) = g^{-1}(\pi'(f(s_t))). \tag{9}$$

In this equation, f and g^{-1} are the state and inverse action mappings defined in Equations 7 and 4 that map states and actions, respectively.

In Equation 9, $f(s)$ maps the underlying state onto an abstract state. The schema's abstract policy π' returns an abstract action associated with this abstract state. Finally, g^{-1} projects this abstract action onto a set of underlying actions that are consistent with the schema policy. Any action in this set is a valid instantiation of the action schema's policy.

This process of policy projection is illustrated in Figure 2. Suppose the system is in underlying state $s_2 \in S$. The mapping f projects s_2 onto abstract state $s'_2 \in S'$. The abstract policy π' specifies that the abstract reach action, ϕ_{reach}, is to be taken from state s'_2. Finally, the inverse mapping g^{-1} projects ϕ_{reach} onto a set of reach actions $a_1, a_2, a_3, \ldots, a_n \in A$. ϕ_{reach} is instantiated by one of the four reach controllers $\phi_r|^{top}_{lh}$, $\phi_r|^{top}_{rh}$, $\phi_r|^{side}_{lh}$, or $\phi_r|^{side}_{rh}$ described in Section II.

In addition to the instantiations of ϕ_{reach}, we allow one instantiation of $\phi_{localize}$ and four instantations of ϕ_{grasp}. In principle, $\phi_{localize}$ can be instantiated by many different localize controllers that are parameterized with different sensor and effector transforms. Depending upon which instantiation is selected, the system can selectively localize different objects using different sensory modalities including visual, auditory, or haptic. In the experiments presented in the paper, we allow a single instantiation of the localize controller that visually determines the location, height, and width of whatever object is placed in front of the robot. This controller segments blobs that correspond to the object in two stereoscopic image planes. Object location is determined by triangulating on the centroid of the two blobs. The horizontal and vertical extent of one of the blobs is used to roughly approximate object shape. While this approach does not yield precise shape information, we expect simple blob statistics such as these to generalize well to new objects.

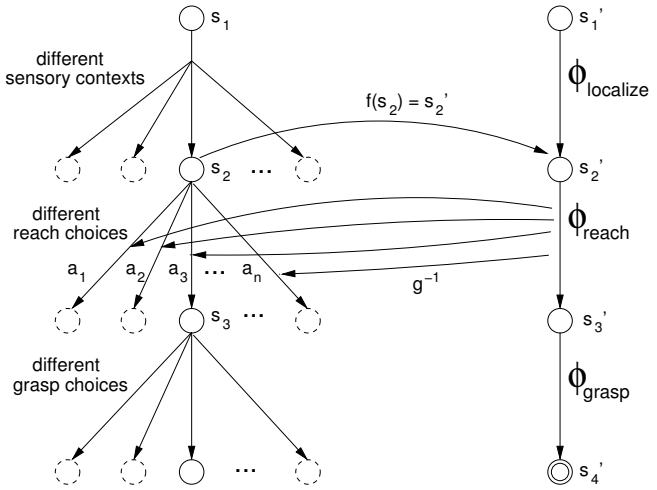


Fig. 2. Right: the localize-reach-grasp action schema. Left: possible instantiations of the action schema.

Similarly, ϕ_{grasp} is instantated by one of the four grasp controllers $\phi_g|^{l12}_{l12}$, $\phi_g|^{l123}_{l123}$, $\phi_g|^{r12}_{r12}$, or $\phi_g|^{r123}_{r123}$. Although this paper only uses physical grasp contacts, we note that grasp controllers can also be parameterized by *virtual contacts* [4], [9]. By adjusting the set of virtual contacts used by the grasp controller, ϕ_{grasp} can be instantiated by almost any type of grasp, including "whole body" grasps [9].

D. Context Differentiation

Although it may be possible to instantiate the action schema with a number of functionally similar behaviors, not all instantiations may be appropriate in a given problem context. In particular, it is desirable to select an instantiation of the action schema that will enable the system to reach the goal state. We refer to the process of selecting instantiations of the action schema based on problem context as *context differentiation.*

The goal of context differentiation is to find a policy π among the instantiations of the action schema policy that maximizes the chance of reaching a schema goal state. Let $V(s_t)$ be the probability of reaching a goal state starting from $s_t \in S$. If the system is already at a goal state, then $V(s_t)$ is trivially equal to one. Otherwise, the probability of reaching a goal state can be calculated by using a form of one-sweep dynamic programming that recursively evaluates $V(s)$ using the following equation:

$$V(s_t) = \max_{a \in B(s_t)} Q(s_t, a), \tag{10}$$

where

$$Q(s_t, a) = \sum_{s_{t+1} \in N(s_t, a)} V(s_{t+1}) P(s_{t+1} | s_t, a). \tag{11}$$

In Equation 11, $N(s_t, a)$ is the set of next states that are consistent with schema constraints and can be reached from s_t by taking a. The recursive process terminates when a goal state is reached.

Optimal actions can be discovered by enumerating all possible schema-consistent actions and evaluating how each

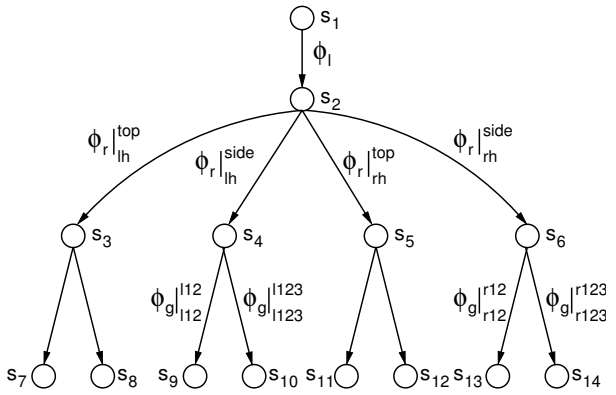


Fig. 3. Instantiations of the localize-reach-grasp action schema used in this paper's experiments.

of them will affect the probability of reaching the goal state. If the system is in state s_t, then the action that yields the maximum probability of reaching a schema goal state is:

$$a_{max} = \arg\max_{a \in B(s)} Q(s_t, a). \quad (12)$$

This approach of evaluating different action choices in terms of the probability of reaching a schema goal state is the basis for an on-line context differentiation algorithm. This algorithm explores the space of actions that adhere to schema constraints and discovers instantiations of the generalized policy that maximize the chances of reaching a schema goal state. In the process of repeatedly executing instantiations of the schema policy, this algorithm acquires relevant experience and improves the likelihood that it will reach a schema goal state. At each step, all possible schema-consistent actions are evaluated to determine which is most likely to lead to a schema goal state using Equation 12. This algorithm is outlined below:

FUNCTION DIFFERENTIATE ACTION SCHEMA
1. Repeat
2. Get current state $s_t \in S$
3. Let $B(s_t) = g^{-1}(\pi'(f(s_t)))$
4. Evaluate $a_{max} = \arg\max_{a \in B(s_t)} Q(s_t, a)$
5. Execute a_{max}
6. Get next state $s_{t+1} \in S$
7. Update transition model $P(s_{t+1}|s_t, a)$
8. While $f(s_t)$ is not the Schema goal state.

In the absence of any experience, the DIFFERENTIATE ACTION SCHEMA algorithm selects random instantiations of the abstract reach and grasp actions. As experience accrues, better estimates of transition probabilities become available. In the case of the localize-reach-grasp action schema, these transition probabilities correspond to estimates of the chances of success of different reach or grasp controllers given the current state of the system and the set of visually localized object properties.

For example, Figure 3 illustrates the possible instantiations of the localize-reach-grasp action schema used in the present experiments. The probability $P[s_{14}|\phi_g|^{r123}_{r123}, s_6, blob]$ represents the probability of a three-fingered grasp succeeding given that the robot has already reached to the side of the object with the right hand and given blob parameters such as object position and horizontal scale. Similarly, the probability $P[s_6|\phi_r|^{top}_{rh}, s_2, blob]$ encodes the chances of being able to reach to the top of an object given an object's position (where position is a component of blob parameters.) As these estimates of transition probabilities improve, the actions selected by Equation 12 in the DIFFERENTIATE ACTION SCHEMA algorithm are more likely to be successful reaches and grasps. This should produce monotonic improvement in average localize-reach-grasp schema performance measured in terms of the frequency of grasp success.

IV. EXPERIMENTAL VALIDATION: BAGGING GROCERIES

We performed a set of experiments in order to demonstrate the ability of the action schema approach to differentiate reach and grasp contexts appropriately. In our experiments, a humanoid robot learned to reach to and grasp grocery items and drop them in a paper bag as shown in Figure 1. The main experimental questions were: (1) does the action schema learn to appropriately differentiate reach and grasp behavior based on context, and (2) does the action schema improve its performance with experience?

A. Method

Dexter, the UMass bi-manual humanoid robot, was used to test our approach [14]. Dexter consists of a 4-DOF bisight head and two whole-arm manipulators (Barrett Technologies, Cambridge MA), each Barrett WAM equipped with a 3-finger, 4 DOF Barrett Hand. Mounted on the tip of each Barrett hand finger is a 6-axis force-torque sensor.

The localize-reach-grasp action schema was trained using four items that are representative of a large class of items that can be purchased in a grocery store: a 828mL bottle of laundry detergent, a 946mL spray bottle containing multi-purpose cleaning solution, a 300ft roll of all-purpose Jute Twine, and a 13Oz container of Vaseline. These objects were placed at arbitrary locations on a table in front of Dexter within reach of its left or right hand. The robot grasped these objects and placed them in a paper bag.

We tested our approach using the localize-reach-grasp action schema. Dexter was capable of reaching to the side or top of the object using either the left or right arm. In addition, Dexter could grasp the object using either two or three contacts on either hand. The localize control action observed the position and horizontal and vertical scale of whatever object was placed in front of the robot. After grasping the object, the action schema applied a grasping force (hold), lifted the object, and placed it in a grocery bag.

The DIFFERENTIATE ACTION SCHEMA algorithm was tested by repeatedly attempting to grasp (starting from a default robot configuration) different objects placed at an arbitrary location on a table in front of the robot. We conducted two experiments, each consisting of 67 attempts to reach and

grasp an object. At the beginning of both experiments, the system had no experience and reach and grasp actions were randomly selected from among the valid mappings of the schema. However, as experience accumulated, the transition model was updated and performance of the action schema improved.

B. Results

Figure 4a shows the action schema's rate of success as a function of time. The horizontal axis is the episode number and the vertical axis is the probability of success. "Probability of success" is a moving average with a five-episode window where each point is the average success (where "success" is defined as either 1 or 0) of the last two episodes, the current episode, and the next two episodes.

Figures 4b and 4c plot the action schema's estimate that each of four instantiations of the schema policy will succeed versus the number of training episodes for two different object types. Figure 4b illustrates how the probability that four possible action schema instantiations are selected evolves as a function of time when small objects (the twine or the Vaseline) are placed on the table. Figure 4c illustrates the evolution of the same probabilities when tall objects (the cleaning solution squirt bottle and the laundry detergent) are placed to the robot's left. Notice that the four probabilities need not sum to one because it is possible for more than one policy instantiation to reliably grasp a particular class of objects.

C. Discussion

With no experience, the DIFFERENTIATE ACTION SCHEMA algorithm initially selects reach and grasp actions randomly, and often fails to place the object in the bag. As experience accrues, actions that are more likely to reach the schema goal are selected. Figure 4a shows that as experience accrues, the system becomes more likely to succeed by placing objects in the bag.

A more detailed view is portrayed in Figures 4b and 4c. As the system gathers more experience, its estimate that each of four policy instantiations will succeed for the two different object types changes. For both large and small objects, all policy instantiations are initially assumed to succeed. The probability that a left handed top grasp succeeds quickly goes to zero for small objects located on the robot's right; this reflects the fact that the left hand cannot reach to the top of these objects. Nevertheless, it *is* possible to use the left hand to reach to the *side* of objects on the right. However, since side grasps do not work well for small objects, Figure 4b shows that these grasps succeed only about 65% of the time. The only grasp that works well for small objects on the right is the right-handed top grasp. Figure 4c shows that tall objects located on the left can be grasped with either hand using a side grasp. Due to the geometry of the detergent and squirt bottle, top grasps always fail on these objects. The graph also shows that the system has difficulty reaching all the way to the left with the right hand and that the system therefore prefers using the left hand to grasp these items.

V. CONCLUSION

The variation in the size, shape, and weight of everyday objects requires different objects to be grasped in different ways. Therefore, a dexterous robot capable of grasping everyday objects must have a large number of different reach and grasp choices available and must be able to select a context-appropriate reach-grasp policy. This paper proposes a generalized representation of a skill that leads to a number of functionally similar behaviors. Through the process of context differentiation proposed in this paper, the system autonomously discovers context-appropriate instantiations of this generalized skill. We show how this approach applies to localize-reach-grasp problems and develop a humanoid grocery-bagging robot based on these principles. The robot learns to associate general visual features regarding location and shape with the correct reach and grasp strategy for several grocery items.

A desirable characteristic of any system that learns to grasp is the ability to generalize to new objects. Although this ability has not yet been tested with our system, we expect reasonable generalization because reach and grasp decisions are conditioned on generic visual characteristics of the object such as location and horizontal and vertical scale. In the future, we hope to verify this in the context of a richer set of blob descriptors by testing performance with a different set of objects than was used during learning.

ACKNOWLEDGMENT

The authors would like to thank Andrew Barto, Oliver Brock, Rachel Keen, Balaraman Ravindran, and the members of the Dexterous Robotics Laboratory at NASA JSC for their contributions to this work. This work was supported by NASA grant NNJ05HB61A, ARO grant DAAD 19-03-R-0017, and NASA GSRP Fellowship NNJ04jf76H.

REFERENCES

[1] G. Schlesinger, *Ersatzglieder und Arbeitshilfen fur Kriegsbeschadigte und Unfallverletzte*. Berlin: Springer, 1919, ch. The mechanical structure of artificial limbs, pp. 21–600.

[2] M. Cutkosky, "On grasp choice, grasp models, and the design of hands for manufacturing tasks," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 269–279, June 1989.

[3] M. Cutkosky and R. Howe, *Dextrous robot hands*. NY: Springer-Verlag, 1990, ch. Human grasp choice and robotic grasp analysis, pp. 5–31.

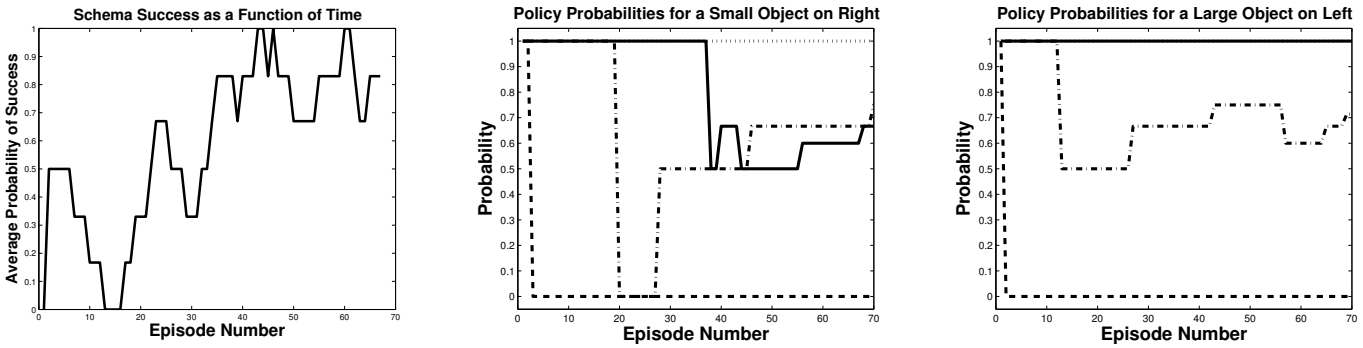
[4] C. MacKenzie and T. Iberall, *The Grasping Hand*. North-Holland, 1994.

[5] I. Kamon, T. Flash, and S. Edelman, "Learning to grasp using visual information," Mathematics and Computer Science, Weizmann Institue Of Science, Tech. Rep. CS94-04, 1994.

[6] M. Moussa, "Combining expert neural networks using reinforcement feedback for learning primitive grasping behaviour," *IEEE Transaction on Neural Networks*, 2003.

[7] J. Coelho and R. Grupen, "A control basis for learning multifingered grasps," *Journal of Robotic Systems*, 1997.

[8] R. Platt, A. H. Fagg, and R. A. Grupen, "Nullspace composition of control laws for grasping," in *IEEE Int'l Conf. on Intelligent Robots and Systems*, 2002.



(a) Learning curve.

(b) Small object / right side policy probabilities.

(c) Large object / left side policy probabilities.

Fig. 4. (a) shows how the probability of schema success rises as learning progresses (probabilities are an average over two experiments.) (b) and (c) show how the probability of selecting four reach-grasp policy instantiations changes during learning. Only policies that use two-contact grasps are shown. The single-dashed line represents reaching to the top of the object with the left hand. The solid line represents reaching to the side with the left hand. The dotted line represents reaching to the top of the object with the right hand. The dash-dot line represents reaching to the side of the object with the right hand.

[9] R. Platt, A. Fagg, and R. Grupen, "Extending fingertip grasping to whole body grasping," in *IEEE Int'l Conference on Robotics and Automation*, 2003.

[10] R. Platt, A. H. Fagg, and R. A. Grupen, "Manipulation gaits: Sequences of grasp control tasks," in *IEEE Int'l Conf. Robotics Automation*, 2004.

[11] M. Huber, "A hybrid architecture for adaptive robot control," Ph.D. dissertation, U. Massachusetts, 2000.

[12] J. Ponce, S. Sullivan, A. Sudsang, J. Boissonnat, and J. Merlet, "On computing four-finger equilibrium and force-closure grasps of polyhedral objects," *Int. J. Rob. Res.*, 1996.

[13] J. Coelho, J. Piater, and R. Grupen, "Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot," *Robotics and Autonomous Systems Journal, special issue on Humanoid Robots*, vol. 37, no. 2-3, November 2001.

[14] R. Platt, O. Brock, A. H. Fagg, D. Karupiah, M. Rosenstein, J. Coelho, M. Huber, J. Piater, D. Wheeler, and R. Grupen, "A framework for humanoid control and intelligence," in *Proceedings of the 2003 IEEE International Conference on Humanoid Robots*, October 2003.