An Intrinsic Reward for Affordance Exploration

Stephen Hart

***Abstract*—In this paper, we present preliminary results demonstrating how a robot can learn environmental *affordances* in terms of the features that predict successful control and interaction. We extend previous work in which we proposed a learning framework that allows a robot to develop a series of hierarchical, closed-loop manipulation behaviors. Here, we examine a complementary process where the robot builds probabilistic models about the conditions under which these behaviors are likely to succeed. To accomplish this, we present an intrinsic reward function that directs the robot's exploratory behavior towards gaining confidence in these models. We demonstrate how this single intrinsic motivator can lead to artifacts of behavior such as "novelty," "habituation," and "surprise." We present results using the bimanual robot Dexter, and explore these results further in simulation.**

***Index Terms*—Affordances, Intrinsic Motivation, Developmental Robotics, Control-Based Representations.**

I. Introduction

Gibson proposed a "theory of affordances" in which organisms perceive their environment in terms of their ability to interact with it [6]. Recent work in the robotics literature has examined how to learn affordance models for robot grasping and pushing [5], [13], [16], [18], mobile robot navigation [15], and tool use [19]. Although this work appropriately grounds a robot's perception in its actions, these affordances were learned under conditions strongly dictated by the programmer. We believe, however, that a robot should be *self-motivated* to learn its models of behavioral affordances. Recent attention in the machine learning literature has investigated computational means of self- or *intrinsic*-motivation that allow agents to gain competence and acquire skills without programming in task-specific rewards and goals [1], [14].

Intrinsic motivators for behavior have long been studied in psychology. Hull introduced the concept of "drives" such as hunger, pain, sex, or escape in human behavior, in terms of deficits that the organism wishes to reduce to achieve homeostatic equilibrium [11]. Berlyne proposed a number of other intrinsically motivating factors—what he called the *collative variables*—such as novelty, habituation, curiosity, surprise, challenge, and incongruity [2]. Festinger suggested that there is a drive to reduce the cognitive dissonance between internal knowledge structures and the current perception of the organism's world [4]. Primary motivators to reduce the discrepancy between cognitive structures and experience were introduced by Kagan [12].

In this paper, we provide an intrinsic motivation function that attempts to bring together much of the above psychology work. This function rewards a robot for taking actions that reduce the uncertainty in its models of which environmental contexts lead to controllable interactions. We characterize this uncertainty by examining how the variance of these models dynamically changes with experience. We call these models "affordance models," and show how they can be acquired in a reinforcement learning framework [20]. Although other work has used variance reduction and measures of model improvement in intrinsically motivated learning systems (c.f., [3], [17]), our work explicitly links the learning of affordance-based memory structures and techniques for intrinsic motivation in real-robot systems.

Our technique for affordance modeling extends previous work examining how a robot can autonomously develop hierarchical control programs [7], [9], [8]. The intrinsic reward function presented in this paper allows a robot to explore the conditions in which such programs are likely to succeed until it achieves sufficient confidence in its internal models. The result is a process complementary to the mechanisms for acquiring the behavior in the first place (as presented in [9], [8]) that can guide a robot towards an enhanced understanding of how it can interact with its environment.

The remainder of this paper is organized as follows. In the next section, we introduce the *control basis* framework for state and action that allows for hierarchical and co-articulated behavior. We then explain the "multi-modal imperative" with which we endow our robots to assemble behavioral programs in this framework. Next, we introduce a formal definition for affordance models. We explain how a robot can acquire and refine such models using a novel intrinsic reward function in learning situations we call *model exploration programs*. We conclude with three illustrative demonstrations showing how model exploration programs result in behavior that could be interpreted as habituation, novelty, and surprise.

II. The Control Basis Framework

The control basis framework provides a combinatoric means of constructing hierarchical and multi-objective closed-loop programs from a robot's sensory and motor resources, and is diagrammed in Figure 1. Although this framework is discussed in more detail in previous publications (c.f., [7], [9], [8]), we briefly summarize its main components here in order to emphasize the representational analogy between the state/action spaces of the robot's control processes and the state/action spaces of its cognitive processes as described in Section III.

The control basis framework supports principled mechanisms for the following:

1) Parameterizable Control Actions:: Primitive actions in the control basis framework are closed-loop feedback controllers constructed by combining a potential function $\phi \in \Omega_\phi$,

S. Hart is with the Laboratory for Perceptual Robotics, University of Massachusetts Amherst, Amherst, MA, 01003, USA e-mail: (shart@cs.umass.edu)

978-1-4244-4118-1/09/$25.00

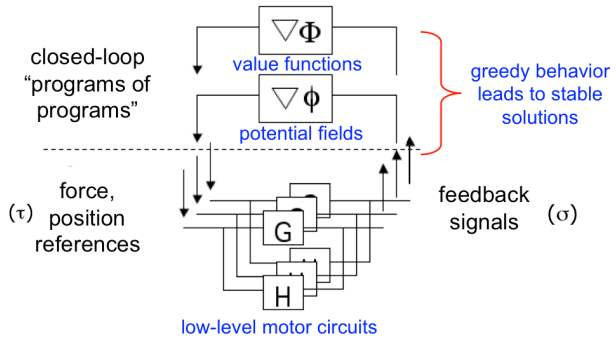


Fig. 1. This diagram shows the hierarchical, control basis architecture. Control actions perform greedy descent on potential fields, ϕ, that are free of local-minima. These actions send reference inputs to low-level motor units (with H and G representing feedback and feedforward transfer functions, respectively) that guarantee stable performance at the hardware level. Programs written using these control expressions ascend value functions, Φ, that produce adaptive-optimal performance.

with a feedback signal $\sigma \in \Omega_\sigma$, and motor variables $\tau \in \Omega_\tau$, into a control action $c(\phi, \sigma, \tau)$. In any such configuration, $\phi(\sigma)$ is a scalar potential function defined to satisfy properties that guarantee asymptotic stability. The sensitivity of the potential to changes in the value of motor variables is captured in the task Jacobian, $J = \partial\phi(\sigma)/\partial\tau$. Reference inputs to lower-level motor units are computed by controllers $c(\phi, \sigma, \tau)$, such that $\Delta\tau = J^{\#}\phi(\sigma)$, where $J^{\#}$ is the Moore-Penrose pseudoinverse of J.

Multi-objective control actions that support co-articulated behavior are constructed by combining control primitives in a prioritized manner. Concurrency is achieved by projecting subordinate/inferior actions into the nullspace of superior actions, and is denoted $c_{inf} \triangleleft c_{sup}$. The combinatorics of potentials Ω_ϕ, and resources Ω_σ and Ω_τ defines all closed-loop, multi-objective actions $a \in \mathcal{A}$ that a robot can employ.

2) State Estimation:: The dynamics $(\phi, \dot{\phi})$ created when a controller interacts with the task domain supports a natural discrete abstraction of the underlying continuous state space [10]. One simple discrete state definition based on *quiescence events* and controller relevance was proposed in [9]. Quiescence events occur when a controller reaches an attractor state in its potential. We define a predicate $p(\phi, \dot{\phi})$ associated with controller $c(\phi, \sigma, \tau)$, such that:

$$p(\phi, \dot{\phi}) \quad = \quad \begin{cases} X & : \quad \phi(\sigma) \text{ state is unknown} \\ - & : \quad \phi(\sigma) \text{ has undefined reference} \\ 0 & : \quad |\dot{\phi}| > \epsilon_\phi, \text{ transient response} \\ 1 & : \quad |\dot{\phi}| \leq \epsilon_\phi, \text{ quiescence,} \end{cases} \tag{1}$$

where ϵ_ϕ is a small, positive constant. The "$-$" condition means that no target stimuli is present in the feedback signal, σ, and the environment does not afford that control action at that time. The unknown "X" condition occurs when a controller is not running and has no dynamics. The "0" occurs during the transient response of c_i as it descends the gradient of its potential, and "1" represents quiescence. Given a collection of n distinct primitive control actions, a discrete state-space $\mathcal{S} \equiv (p_1 \cdots p_n)$ can be formed.

3) Hierarchical Programming:: Sensorimotor programs are learned in the control basis framework given the state and action spaces $\mathcal{S}$ and $\mathcal{A}$ defined by the set $\{\Omega_\phi, \Omega_\sigma, \Omega_\tau\}$ and a reward function $\mathcal{R}$. Formulating the learning problem as a Markov Decision Process (MDP), allows a learning agent to estimate the value, $\Phi(s, a)$, of taking an action a in a state s using reinforcement learning (RL) techniques [20]. Representing behavior in terms of a value function provides a natural hierarchical representation for control basis programs where attractor states of the value function, Φ, capture quiescence events in the policy. As a result, the state of a program can be captured using the same state-predicate representation as above, even though that program may have its own complex transition dynamics.

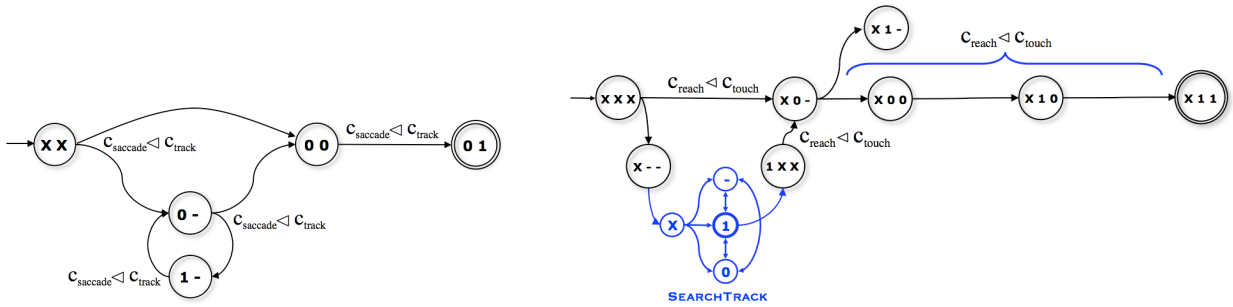
4) A Multi-Modal Imperative:: In previous work, we introduced a reward function, r_{mmi}, called the "multi-modal imperative" (or MMI) that allows a robot to learn hierarchical control basis programs [9]. The MMI focuses a robot's attention around aspects of its environment that exhibit a degree of controllability and stable interaction, regardless of that aspect's "mode" (visual, auditory, tactile, etc.). Specifically, the MMI provides a unit of reward to the robot whenever it can configure and run to quiescence any control action that responds to a feedback signal, $\sigma \in \Omega_\sigma$, directly measuring information from the robot's environment.

The MMI has successfully been used to teach the robot Dexter (Figure 3(c)) a number of manipulation behaviors—many of which employ other of these same behaviors hierarchically—labelled with the intuitive names SEARCHTRACK, REACHTOUCH, VISUALINSPECT, HANDTRANSFER, and PICKANDPLACE. In each program—or behavioral *schema*—a policy was learned that orients the robot to respond to a single rewarding control event by means of the MMI. Figure 2 provides the learned policies for two of these behaviors, SEARCHTRACK and REACHTOUCH, both of which are used in the experiments in Section IV. It is important to note that although these behaviors appear to be used as atomic actions in these experiments, each is a complex closed-loop control program with its own internal transition dynamics.

In the remainder of this paper, we examine how an intrinsically motivated system exploring this suite of behaviors can build models of the conditions under which MMI reward is achieved. The result is a framework for learning probabilistic models of affordances that can be acquired and adapted over the course of a robot's lifetime.

III. AFFORDANCE-BASED MEMORY

In [8], we examined how a robot could generalize control basis schema to environmental contexts different from those in which they were initially learned. This was achieved by finding decision boundaries in the input signals to its control actions and determining how to employ different resources (e.g., a robot's left arm vs. its right arm) that probabilistically predicted rewarding outcomes. Formally, the robot learned distributions of the form $Pr(r_{mmi}|f, a_i)$ capturing the likelihood of achieving MMI-type reward, r_{mmi}, for taking action $a_i \in \mathcal{A}$ after observing the environmental context $f \in \mathcal{F}$.



(a) (b)

Fig. 2. In this figure we show two behavioral schema, both of which are described in more detail in [9]. (a) shows a learned policy for a SEARCHTRACK program using the multi-objective control action $c_{saccade} \triangleleft c_{track}$. This program moves Dexter's pan/tilt head in search of a particular hue-, saturation-, or intensity-space visual feature the robot can then track. The state space for this program is $\mathcal{S}_{st} = (p_{saccade}\ p_{track})$. The robot begins in state (XX) and takes the composite action. If the target stimuli to be tracked is seen in the robot's cameras, the robot continues running the action until c_{track} quiesces in state (01), and the robot receives reward from the MMI. If the stimuli is not initially present the robot enters state $(0-)$ from which it repeatedly saccades by moving its pan/tilt cameras until the stimuli appears causing the robot to enter state (00). (b) shows a learned policy for a REACHTOUCH program that allows Dexter to visually seek out objects that it can then reach out and touch. This program has state space $\mathcal{S}_{rt} = (p_{st}\ p_{reach}\ p_{touch})$ and uses SEARCHTRACK hierarchically (as demonstrated by the SEARCHTRACK icon in the transition graph), with corresponding state predicate p_{st}. It also employs two other primitive control actions c_{reach} and c_{touch} in a composite control law. The robot begins by trying to reach out and touch a visual feature of a certain type. If no visual feature of that type is found (causing c_{reach} to be undefined), the robot invokes SEARCHTRACK until such a feature is found and tracked. The robot, at this point, runs $c_{reach} \triangleleft c_{touch}$ until the rewarding touch action becomes defined and quiesces, or the reach quiesces but the robot is not in contact with any object (it is out of reach). For some simple objects within reach of the robot, this policy will result in controlled touch response that results in a primitive "grab." Note that both behaviors can be parameterized by visual σ's of a certain type. For example SEARCHTRACK(blue) will search for and track blue features; REACHTOUCH(red) will reach out and grab small red objects in the robot's workspace.

In general, the values of $\mathcal{F}$ could capture any information that might predict MMI reward. We chose to look exclusively at the values of the feedback signals, $\sigma \in \Omega_\sigma$, used in the control action set of the schema in order to restrict a possible infinite feature space to a tractable collection of variables. Furthermore, we only examined the means, velocities, and spatial volumes of each σ measured *immediately* before each action was run.

Examining the feature set $\mathcal{F}$ allows a robot to use its experience to build models of contexts that are likely to lead to MMI reward if a given program is run. We believe such models best capture the likely affordance of a particular action at any given time. We therefore call such models "affordance models" defined such that:

$$\mathcal{M}(a_i) \quad \equiv \quad Pr(f|r_{mmi}, a_i). \tag{2}$$

It is our goal to create learning situations in which a robot may build reliable affordance models that it can store in its memory as implicit, behavioral knowledge. We next propose how this might be done.

A. Model Exploration Programs

We define a class of MDPs called *model exploration programs*. These MDPs have action set $\mathcal{A}$, and an $|\mathcal{A}|$-predicate state representation set that captures the robot's "confidence" of each of its affordance models. We define this confidence in terms of *model variance quiescence*. The quiescence of each model is defined by a 4-valued predicate logic as follows:

$$p(\dot{\Sigma}(\mathcal{M}_i)) \quad = \quad \begin{cases} X & : \quad \mathcal{M}_i \text{ is unknown} \\ - & : \quad \mathcal{M}_i \text{ has undef features } f \\ 0 & : \quad |\dot{\Sigma}(\mathcal{M}_i)| > \epsilon_m \\ 1 & : \quad |\dot{\Sigma}(\mathcal{M}_i)| \leq \epsilon_m, \end{cases} \tag{3}$$

where ϵ_m is a small, positive constant, $\Sigma(\mathcal{M}_i)$ is the variance of model $\mathcal{M}(a_i)$, and $\dot{\Sigma}(\mathcal{M}_i)$ is its rate of change as experience is gathered.

Before the robot has experience with an action a_i, its affordance model will be unknown, and $p(\dot{\Sigma}(\mathcal{M}_i)) =$ "X". If the environment does not afford a behavior at a given time the corresponding predicate value will be undefined, and $p(\dot{\Sigma}(\mathcal{M}_i)) =$ "$-$". This occurs when no relevant input signals to the action are present on the robot's sensors, and its input can not be evaluated. However, in the cases where the robot can take a particular action, the predicate value for that action's model will be 0 or 1, depending on the dynamics of that model. It is worth noting the similarity of this state representation to the state representation in control basis programs; one captures dynamic properties of a robot's actions, the other captures dynamic properties of its internal knowledge structures.

We now define an *intrinsic* reward function for model exploration programs. In this function, reward is received at time t for taking action a_i if the robot received new information that changes its "confidence" of the corresponding affordance model, such that:

$$r_{aff}(t) \quad = \quad |\Sigma(\mathcal{M}(a_i, t)) - \Sigma(\mathcal{M}(a_i, t-1))| - \rho, \tag{4}$$

where $\Sigma(\mathcal{M}(a_i, t))$ is the variance of affordance model $\mathcal{M}(a_i)$ at time t, and ρ is a small positive constant constituting a cost for performing each action. In cases where the models are multi-dimensional, the magnitude of the variance across all dimensions can be used to compute this reward.

We assume that the variance of a given affordance model grows more stable as more experience is gathered. Therefore, we would expect a robot using reinforcement learning to explore model exploration programs will build stable affordance

models over time. Moreover, we expect the robot to initially get large amounts of reward for engaging novel contexts, "habituating" quickly to those contexts that are relatively stable. We also expect the robot to spend more time exploring more variable contexts and to react strongly to contexts that change in "surprising" and unexpected ways.

We allow each exploration program to contain a NOOP action in its action set that has no cost. We expect the robot to select this action over its other behaviors as the cost for taking them eventually outweighs the expected information gained.

It should be noted that although each model exploration program is defined as an MDP, the goal is not to find a fixed, optimal policy for that MDP. Because the reward is non-stationary this is not feasible. Instead, we expect that an agent acting greedily to maximize reward at every time step will spend time taking actions that gather the most information in terms of the underlying affordance models. However, formulating model exploration programs as MDPs that can be submitted to reinforcement learning algorithms provides a convenient way for the robot to habituate on its models in an adaptive-optimal manner.

IV. DEMONSTRATION

To provide a demonstration of how model exploration programs may be used in practice, we now present three simple examples. Because it is our goal in this paper to examine the qualitative and quantitative performance of model exploration programs, the state/action set for each progam is provided by the programmer *a priori*. It should be noted, however, that the author is currently investigating means for the robot to create its own exploration problems autonomously, with less explicit guidance from a human programmer.

In each of the following three demonstrations, the robot uses a model exploration program to guide its behavior in estimating multi-dimensional Gaussian affordances models for each of the program's actions. The models are initialized with zero mean and unit variance. For each, a single trial was initially performed on the robot Dexter, and twenty-four additional trials (for a total of twenty-five) were performed using a realistic Dexter simulator. The simulation experiments used the probabilistic models learned in the real trial (the likelihood of success and the estimated affordance model for each action) to generate realistic samples. The empirical results presented in the following are averaged over all twenty-five trials.

A. Experimental Setup

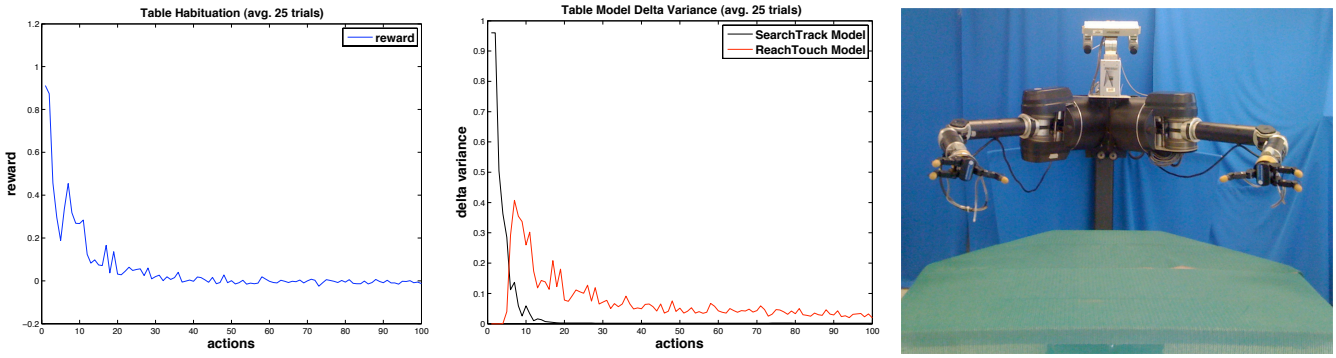
For all three experiments, a model exploration program is instantiated and explored using Q-learning with ϵ-greedy exploration ($\epsilon = 0.05$)[1] [20], and the action penalty, $\rho = 0.05$. At the beginning of each trial, the robot has no experience and thus undefined affordance models. As the robot performs actions it receives reward by reducing the change in variance of these models as more samples are added. If the rate of change of the variance between successive executions of an action drops below $\epsilon_m = 0.1$, we treat the model as having quiesced, as per Equation 3. Each model is updated after the execution of the corresponding action if reward from the multi-modal imperative is received. The models are not updated when the action does not succeed, accumulating no MMI-reward, only cost. Performance is analyzed over 25 trials (1 real, 24 simulated) of 100 actions each. The change in variance after each action was normalized to the maximum value observed during each trial across all affordance models in the program. We describe each of the three experiments next.

1) Dexter's Table: In the first example, Dexter's "first" object—the green table seen in Figure 3(c)—is placed in the robot's workspace. We let the robot explore two behaviors and the "no-op" action, such that $\mathcal{A}_1 = \{$SEARCHTRACK(green), REACHTOUCH(green), NOOP$\}$. From the two behaviors, we can create a model exploration program that explores the affordances for Dexter being able to visually track the table (with SEARCHTRACK), and reach out and touch it (with REACHTOUCH). The feature space for the SEARCHTRACK model is a five dimensional vector, $f_s = [\mathbf{u}(t)\ \dot{\mathbf{u}}(t)\ \gamma_u(t)]$, containing the centroid $\mathbf{u}(t)$ in $\mathbb{R}^2$ of green pixel regions on the robot's left camera image at time t, its area $\gamma_u(t)$, and its velocity, $\dot{\mathbf{u}}(t)$. The feature space for the REACHTOUCH model is a seven dimensional vector, $f_r = [\mathbf{x}(t)\ \dot{\mathbf{x}}(t)\ \gamma_x(t)]$, containing the Cartesian location $\mathbf{x}(t)$ in $\mathbb{R}^3$ of the table found by triangulating the view of the green pixel regions in both of the robot's camera images, that region's estimated 3D volume, $\gamma_x(t)$, and its velocity, $\dot{\mathbf{x}}(t)$.

2) Three Moving Objects: In the second example, three different colored objects (a red, a yellow, and a blue ball) are placed on the green table in front of the robot. The robot is given an action set $\mathcal{A}_2 = \{$REACHTOUCH(red), REACHTOUCH(yellow), REACHTOUCH(blue), NOOP$\}$, resulting in a three-predicate state-space for the model exploration program—one predicate for each corresponding affordance models. The robot is allowed to explore this action set to learn the affordance models for each. During the trial, the objects are placed in random locations on the table. After each action, the positions of the red, yellow, and blue balls on the table are moved to locations that differ from their original position with (approximate) variances of $0.01m$, $0.05m$, and $0.25m$, respectively. We hypothesize that the robot will explore the objects in proportion to each object's variance. In other words, it will engage the blue object more than the yellow or red object, and the yellow object more than the red.

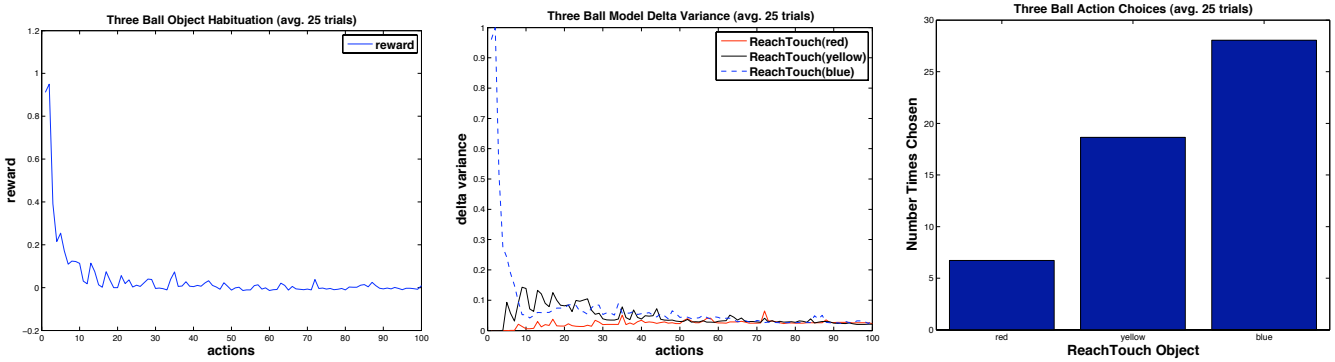
3) Novel & Surprising Objects: In the third example, a two-predicate model exploration program is instantiated in which Dexter explores the action set $\mathcal{A}_3 = \{$REACHTOUCH(red), REACHTOUCH(blue), NOOP$\}$ and their corresponding affordance models. For the first 25 actions, a red ball is placed in locations on the table with a small positional variance. After the robot takes 25 actions, a blue ball is placed on the table also with small positional variance. After 25 additional actions are taken, the position of the red ball is moved to a different location on the table 25 centimeters from its original location. In this example, we hypothesize two results: 1) Dexter will engage the novel, blue object after it has habituated on the

[1] A small exploration constant was chosen to inject some stochasticity into the action selection process. Exploration arises naturally from the non-stationary aspects of the intrinsic reward function.



(a) (b) (c)

Fig. 3. (a) shows the reward (averaged over 25 trials) received after each action for the model exploration program that tracks and touches the green table. Reward becomes negligible after 20-30 actions. (b) shows the change in model variance for the two affordance models used in the program.



(a) (b) (c)

Fig. 4. (a) shows the reward (averaged over 25 trials) received after each action for the model exploration program that reaches out and grabs the three colored balls. Reward becomes negligible after 30-40 actions. (b) shows the change in model variance for the three affordance models used in the program. (c) shows the number of times each action is taken before the corresponding affordance model quiesces.

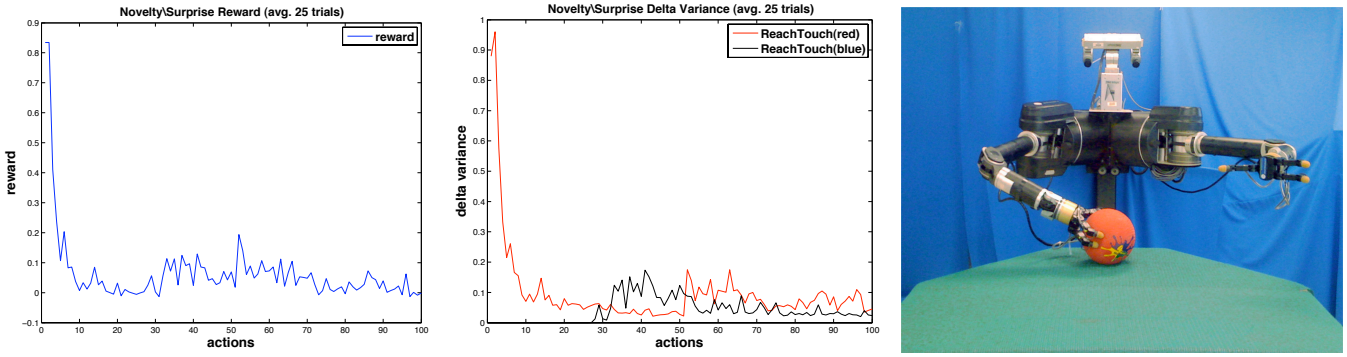
affordance model for the red object because it will produce more reward, and 2) Dexter will exhibit a form of "surprise" by re-engaging the habituated red object when its location varies from its estimated model.

B. Results

Figure 3 shows the performance of the model exploration program for the table experiment. We see in Figure 3(a) the reward received after each action is taken (averaged over the 25 trials). This figure shows that after 20-30 actions, the robot receives negligible reward for taking either of its two actions. Figure 3(b) shows the change in variance for each of the affordance models in the program. We can clearly see in this plot the switch that occurs after about five or six actions when the change in variance of the REACHTOUCH affordance model starts to outweigh the habituating SEARCHTRACK affordance model.

Figure 4 shows the performance of the model exploration program for the three moving objects experiment. In these experiments, it takes the robot about 30-40 actions to habituate on its three affordance models, as seen in Figure 4(a). As expected, we see in Figure 4(b) that it takes each of the three models an amount of time to quiesce that grows with the variance of the locations that the balls are placed in. Figure 4(c) shows the average number of actions taken to explore each of the colored objects before model quiescence, averaged over the 25 trials. It clearly shows that it takes increasingly more actions to quiesce on the more variable objects. On average about 7 actions are taken before the robot habituates on the red object, about 20 for the yellow object, and about 27 for the blue object.

The final experiment shows that model exploration programs—in addition to exhibiting "habituation" type behavior—also results in behavior that is typically described as reacting to "novelty" and being "surprised." Figure 5 shows the reward and variance plots for this experiment. The robot quickly habituates on the red object, switching to the "novel" blue object when it appears after 25 actions, in turn habituating on that object model after about 25 more actions. Also, when the position of red object moves it "surprises" the robot, temporarily creating more uncertainty in its affordance model for REACHTOUCH(red), and causing the robot to re-engage that object, and to receive reward until it habituates once again.



(a) (b) (c)

Fig. 5. (a) shows the reward (averaged over 25 trials) received after each action for the model exploration program for the third experiment in which the second object only becomes available after 25 actions, and the first object changes location after 50 actions. Reward becomes negligible during the first 5-10 actions as the robot explores the first object, but increases when the novel object appears, and again when the robot is surprised by the change of position of the first object. (b) shows the change in model variance for the two affordance models used in the program. We see the change in variance decrease for the first (red) object initially, but increase when its position is moved after 50 actions. We also see the change in variance become apparent when the novel (blue) object appears, decreasing as the robot explores that object. (c) shows Dexter after running REACHTOUCH on the red ball.

V. DISCUSSION

In this paper, we provided a framework for intrinsically motivated affordance modeling. This framework is built upon a grounded, control-based architecture that allows a robot to organize its sensory and motor resources into behavioral programs according to a "multi-modal imperative," and then explore the environmental conditions under which these behaviors succeed. We provided three demonstrations in which the robot Dexter learns models of objects in its environment: first about a table we place in the robot's workspace, then about objects that appear on that table, and finally about objects that come and go in interesting ways.

Although the empirical demonstrations in this paper were performed in simple contexts with simple objects, we expect the methodology to apply in more sophisticated situations. In particular, we limited the examples to allow the robot only to learn affordance models for trackability and touchability in terms of an object's location and size, but we expect model exploration programs to perform comparably for more interesting affordances such as graspability, liftability, or assembly, given that the robot has a program for accomplishing such actions. We are currently investigating how the robot can learn these behaviors by means of the multi-modal imperative.

ACKNOWLEDGEMENTS

This work was supported by NSF award SGER IIS-0847895. The author would like to thank Shiraj Sen for his useful feedback in its development.

REFERENCES

[1] A. Barto, S. Singh, and N. Chentanez, "Intrinsically motivated learning of hierarchical collections of skills," in *Proceedings of the International Conference on Development and Learning (ICDL)*, LaJolla, CA, 2004.
[2] D. E. Berlyne, *Conflict, Arousal, and Curiosity*. McGraw-Hill, 1960.
[3] M. Duff, "Design for an optimal probe," in *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, Washington, DC, 2003.
[4] L. Festinger, *A Theory of Cognitive Dissonance*. Evanston, Row, Peterson, 1957.
[5] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini, "Learning about objects through action: Initial steps towards artificial cognition," in *IEEE International Conference on Robotics and Automation*, 2003.
[6] J. J. Gibson, "The theory of affordances," in *Perceiving, acting and knowing: toward an ecological psychology*. Hillsdale, NJ: Lawrence Erlbaum Associates Publishers, 1977, pp. 67–82.
[7] S. Hart and R. Grupen, "Natural task decomposition with intrinsic potential fields," in *Proceedings of the 2007 International Conference on Intelligent Robots and Systems (IROS)*, San Diego, California, 2007.
[8] S. Hart, S. Sen, and R. Grupen, "Generalization and transfer in robot control," in *8th International Conference on Epigenetic Robotics (Epirob08)*, 2008.
[9] ——, "Intrinsically motivated hierarchical manipulation," in *Proceedings of the 2008 IEEE Conference on Robots and Automation (ICRA)*, Pasadena, California, 2008.
[10] M. Huber and R. Grupen, "Learning to coordinate controllers - reinforcement learning on a control basis," in *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 1997.
[11] C. Hull, *Principles of Behavior: An Introduction to Behavior Theory*. New York, NY: Appleton-Century-Croft, 1943.
[12] J. Kagan, "Motives and development," *Journal of Personality and Social Psychology*, vol. 22, pp. 51–66, 1972.
[13] J. Modayil and B. Kupiers, "Autonomous development of a grounded object ontology by a learning robot," in *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07)*, 2007.
[14] P. Oudeyer and F. Kaplan, "How can we define intrinsic motivation?" in *8th International Conference on Epigenetic Robotics (Epirob08)*, 2008.
[15] E. Şahin, M. Çakmak, M. Doğar, E. Uğur, and G. Üçoluk, "To afford or not to afford: A formalization of affordances toward affordance-based robot control," *Adaptive Behavior*, vol. 4, no. 15, pp. 447–472, 2007.
[16] A. Saxena, J. Driemeyer, and A. Ng, "Robotics grasping of novel objects using vision," *International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2007.
[17] J. Schmidhuber, "Adaptive curiosity and adaptive confidence," Institut fur Informatik, Technische Universitat Munchen, Tech. Rep. FKI-149-91, 1991.
[18] M. Stark, P. Lies, M. Zillich, J. Wyatt, and B. Schiele, "Functional object class detection based on learned affordance cues," in *Sixth International Conference on Computer Vision Systems, Vision for Cognitive Systems*, Santorini, Greece, 2008.
[19] A. Stoytchev, "Toward learning the binding affordances of objects: A behavior-grounded approach," in *Proceedings of the AAAI Spring Symposium on Developmental Robotics*, Stanford University, 2005.
[20] R. Sutton and A. Barto, *Reinforcement Learning*. Cambridge, Massachusetts: MIT Press, 1998.