Detecting the Functional Similarities Between Tools Using a Hierarchical Representation of Outcomes

Jivko Sinapov and Alexadner Stoytchev
Developmental Robotics Lab
Iowa State University
{*jsinapov, alexs*}*@iastate.edu*

Abstract— **The ability to reason about multiple tools and their functional similarities is a prerequisish for intelligent tool use. This paper presents a model which allows a robot to detect the similarity between tools based on the environmental outcomes observed with each tool. To do this, the robot incrementally learns an adaptive hierarchical representation (i.e., a taxonomy) for the types of environmental changes that it can induce and detect with each tool. Using the learned taxonomies, the robot can infer the similarity between different tools based on the types of outcomes they produce. The results show that the robot is able to learn accurate outcome models for six different tools. In addition, the robot was able to detect the similarity between tools using the learned outcome models.**

Index Terms— **Developmental Robotics, Autonomous Tool Use, Robot Manipulation.**

I. Introduction

Tool use is one of the hallmarks of intelligence and is fundamental to human life. Many animals have also been observed to use tools [1], indicating that such an ability is a general adaptation mechanism that helps overcome physical limitations imposed by an organism's anatomy.

For a robot to adapt to human environments, it needs to be able to recognize, reason, and learn the functional properties of different tools it encounters. More specifically, a robot needs to be able to distinguish between similar and different tools based on their functional properties. While object categorization based on visual features is a well studied problem, this paper introduces a model which allows the robot to detect functional similarity between tools based on the robot's interactive experience with them.

To detect the functional similarity between tools, a robot needs to model the types of environmental changes (i.e., outcomes) it can induce and detect through actions with each tool. This paper makes two contributions toward solving this problem. First, this paper introduces a framework in which the robot incrementally learns and uses an *adaptive hierarchical taxonomy* for the types of outcomes observed as a result of the robot's interaction with a tool. The proposed method allows the robot to form novel classes of outcomes as a result of ongoing experience with the tool.

Second, this paper shows how a robot can estimate the functional similarity between tools based on the learned compact representations for outcomes that the tools produce.

This allows the robot to compare tools based on what it can do with them as opposed to comparing the tools based on their visual features (e.g., shape, color, etc.).

II. Related Work

In one of the earliest examples of autonomous tool use, Bogoni [2] presents and evaluates a system in which the robot identifies functional features of objects involved in cutting and piercing operations. The robot uses a superquadratic model of the tool's shape in order to discover visual features that are characteristic of successful tools (e.g., tools that can pierce). In addition, several methods have been proposed for object recognition based on functionality using computer vision and 3D laser scanners [3], [4], [5]. However, these systems try to categorize the objects (typically human-made tools) without active autonomous exploration by a robot.

More recently, Kemp and Edsinger [6] explored how a robot can identify task-relevant features of human-made tools and showed how a robot can learn to detect and control the tip of objects (e.g., the tip of a brush). In previous work, Stoytchev [7], and Sinapov *et al.* [8] demonstrate how robots can solve tool-using tasks using an affordance representation for the tools.

In some tasks, the outcomes that the robot detects as a result of its behaviors are high-dimensional. Sahin *et al.* [9] and Montesano *et al.* [10] propose to solve this problem by clustering an initial set of observations into k clusters, each representing a class of observed effects. The formation of classes allows the robot to use machine learning methods designed for discrete data (e.g., Support Vector Machines in [9] and Baysean Networks in [10]). However, with both of these methods the robot cannot discover novel classes of observed effects as a result of new experience.

The method described here overcomes this problem by presenting a framework in which the robot incrementally learns and uses an adaptive hierarchical taxonomy to describe the types of changes it can induce and detect in its environment through its own actions with a tool. Using the learned representations, the robot is able to infer how similar two tools are in terms of what the robot can do with them.

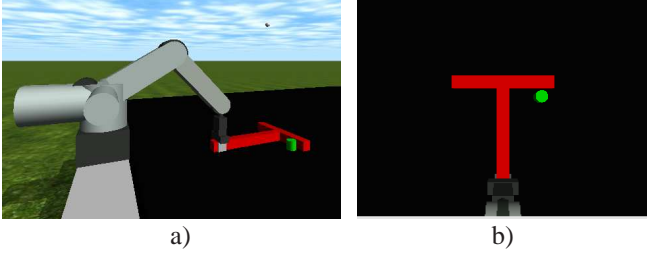a)   b)

Fig. 1. a) Snapshot from the dynamics simulator showing the robot arm;
b) View from the robot's simulated camera.

Fig. 2. The six different tools used by the robot. From left to right: T-Stick,
L-Stick, Stick, L-Hook, Y-Stick, Arrow.

III. EXPERIMENTAL SETUP

All experiments were performed using a robot simulator based on the Open Dynamics Engine [11] developed in-house. The robot is a simulated CRS+ A251 arm with 6 degrees of freedom: a slider joint at the base, waist roll, shoulder pitch, elbow pitch, wrist pitch, and wrist roll. The robot also has a gripper attached to the wrist. A snapshot of the simulated robot arm is shown in Fig. 1.a. Six different tools are used by the robot: *T-Stick, L-Stick, Stick, L-Hook, Y-Stick*, and *Arrow* (see Fig. 2). The last object in the simulation is a small cylindrical puck which can be moved by the tool when the robot performs an action.

A. Sensory Input, Behaviors and Perceptual Cues

The robot's sensory input is extracted from a camera positioned directly overhead and looking downward. Fig. 1.b show a sample visual input image. The robot's set of behaviors, $\mathcal{B}$, consists of 6 exploratory behaviors with the tool: *push, pull, slide-left, slide-right, rotate-left* and *rotate-right*. Fig. 3.a and 3.b show the view from the robot's camera before and after a behavior has been executed.

The robot's perceptual cues, C_i, are derived from the camera frames which are retinally mapped to a 30x30 image centered on the green puck, as shown in Fig. 3.c. Formally, $C_i \in \mathbb{R}^{30 \times 30 \times 3}$, i.e. C_i contains the RGB values of each pixel in the retinal image. The retinal mapping method that was used is described in [12].

B. Outcome Detection

After the robot executes a behavior, it tracks the puck's displacements over time. Let t be the time at which the robot executes a behavior B_i while observing cues C_i. The perceived outcome is defined as $O_i = [dx_{t+1}, dy_{t+1}, \cdots, dx_{t+k}, dy_{t+k}]$
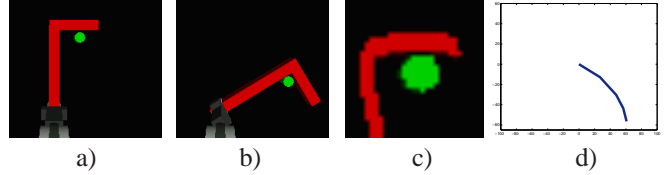

a)   b)   c)   d)

Fig. 3. One sample trial with the L-Stick tool and the *rotate-right* behavior.
a) Configuration of the robot, the tool, and the puck at the start of the trial; b) End of trial configuration after the *rotate-right* behavior has been performed; c) Retinal image used as cue at beginning of the trial; d) Observed outcome plotted as the trajectory of the puck in retinocentric coordinates. The trajectory is plotted relative to the puck's starting location.

such that dx_j and dy_j are the horizontal and vertical displacements of the puck between times $j - 1$ and j as observed in the robot's camera image. Each outcome vector O_i can be visualized as the trajectory of the puck's movements, as shown in Fig.3.d. Each behavior is executed for 3000 simulator time steps and the robot samples the position of the puck in the input camera image every 600 time steps. Thus, each outcome consists of a 10-dimensional feature vector describing the vertical and horizontal movement of the puck across 5 different points in time.

C. Data Collection

During each trial, the tool is positioned in front of the robot and the puck is randomly placed in the vicinity of the tool. The robot first grasps the tool and then randomly selects a behavior $B_i \in \mathcal{B}$ for execution. Fig. 3 shows a trial in which the robot applies its *rotate-right* behavior with the L-Stick. Once the behavior B_i has been executed, the robot acquires the triple (B_i, C_i, O_i), indicating that outcome O_i was observed after executing behavior B_i while detecting perceptual cues C_i. The new data point is then used to update the robot's model for the given tool, as described below.

IV. THEORETICAL MODEL

For many tasks (e.g., prediction of outcomes given behaviors and cues), it is important for the robot to form concept classes describing the types of outcomes it observes. In both [9] and [10], for example, the robot clusters an initial set of observations and treats each cluster as a discrete class of outcomes. This paper presents an alternative approach in which the robot incrementally learns a hierarchy of outcome classes, which allows it to discover novel concepts and to model the observed outcomes at different levels of abstraction.

A. Taxonomy of Outcome Classes

In this work, the robot learns and uses a taxonomy describing the possible *classes* of outcomes that it is able to induce and detect in its environment. Formally, a taxonomy, $\mathcal{T}$, is a tree defined over outcome classes (i.e., nodes) $v_0, \ldots, v_M$. Let $O_j^{mean} \in \mathbb{R}^m$ denote the outcome *prototype* for the

observed outcomes that belong to node v_j, where m is the dimensionality of each observed outcome O_i.

Given a taxonomy $\mathcal{T}$ and an observed outcome O_i, the robot can classify the outcome according to the learned taxonomy. Let $P_i = [v_{root}, \ldots, v_l]$ be a path from the root node v_{root} to some leaf node v_l, describing how O_i relates to $\mathcal{T}$, such that O_i belongs to all class nodes v_j on the path.

Fig. 4 shows a simple example of an acquired taxonomy in which each observation O_i is sampled from a 1-D mixture of gaussians distribution. The taxonomy was constructed from 1000 data points (sampled from the distribution shown in Fig. 4.b) using the method described below. The shaded nodes in Fig. 4.a show how an example outcome is classified according to the learned taxonomy.

B. Learning the Taxonomy

An incremental hierarchical top-down clustering approach was used to learn a taxonomy of detected outcomes for each tool. Given a newly observed outcome, O_i, the (possibly empty) taxonomy, $\mathcal{T}$, is updated as follows:

1) Let $P_i = [v_0, \ldots, v_l]$ be the classification path of O_i according to $\mathcal{T}$.
2) For each class $v_j \in P_i$, recompute the estimate of O_j^{mean} using the outcomes that fall within v_j.
3) Add O_i to the leaf node v_l. If a splitting criterion is met, cluster the outcomes in v_l into k clusters and for each add a child node of v_l to the taxonomy $\mathcal{T}$.

The X-Means clustering algorithm [13] was used in step 3. X-Means is an extension to the standard K-Means clustering algorithm with an added efficient estimation of the number of clusters. An attempted split is performed on a leaf node if the number of outcomes that fall into it exceeds a threshold γ. In all experiments, the threshold γ was set to 300 for the root node v_0. For all subsequent nodes v_j in the taxonomy $\mathcal{T}$, γ was set to 70.

In step 1, the robot uses a top-down classification rule to classify an outcome O_i according to $\mathcal{T}$. Starting at the root, the robot estimates the child outcome class v_j for O_i such that $d(O_i, O_j^{mean}) < d(O_i, O_c^{mean})$ for all other child outcome classes v_c of the root (i.e., standard K-means classification rule with Euclidean distance function d). If v_j is not a leaf node, the same rule is recursively applied until the full path from the root to a leaf is constructed.

C. Comparing Tools Using their Outcome Taxonomies

The learned hierarchical representation for the types of outcomes produced by each tool can be used in order to infer how similar or different two tools are. The distance measure that was used by the robot takes into account the functional properties of the tools, i.e., two tools that produce similar outcomes should be considered similar and vice versa.

The problem is formulated as follows. Given a set of N tools, and the learned taxonomies $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N$ for each
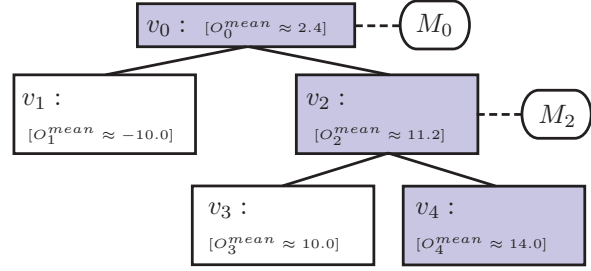
a) A hierarchical taxonomy constructed for outcomes sampled from the mixture of Gaussians distribution in b).
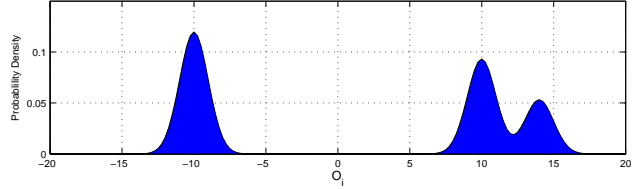
b) Probability distribution of the observed outcomes.

Fig. 4. An example hierarchical outcome taxonomy $\mathcal{T}$. The outcome is a 1-D value sampled from a mixture of gaussians distribution shown in b). The shaded nodes represent an example path, P_i, from the root to a leaf node which shows how some given outcome O_i (e.g., $O_i = 14.5$) is classified according to $\mathcal{T}$. M_0 and M_2 are the predictive models (described in section IV.D) associated with the two non-leaf nodes in $\mathcal{T}$.

tool, compute an N by N distance matrix A such that each matrix element a_{sR} is a measure indicating the distance between tools S and R in terms of their functional properties.

Given an outcome taxonomy $\mathcal{T}_S$ (obtained through experience with tool S), let $\mathcal{L}_S = [O_1^S, O_2^S, \ldots, O_{m_S}^S]$ be the set of leaf outcome class prototypes in $\mathcal{T}_S$. Furthermore, let $\mathcal{L}_R = [O_1^R, O_2^R, \ldots, O_{m_R}^R]$ be the set of leaf class prototypes of some other taxonomy $\mathcal{T}_R$ constructed after experience with tool R. By comparing the leaf outcome classes in the two taxonomies $\mathcal{T}_S$ and $\mathcal{T}_R$, the robot can estimate the distance between tools S and R in terms of the types of outcomes that they produce.

More specifically, given an outcome prototype O_i^S in $\mathcal{L}_S$, let the function $BestMatch(O_i^S, \mathcal{L}_R)$ return the leaf prototype O_j^R such that O_j^R is the prototype in $\mathcal{L}_R$ that is most similar to O_i^S. In other words, $d(O_i^S, O_j^R) < d(O_i^S, O_p^R)$ for all other prototypes $O_p^R \in \mathcal{L}_R$, where $d(x, y)$ is the Euclidean distance function.

Following, two distance measures are defined which compare two taxonomies $\mathcal{T}_S$ and $\mathcal{T}_R$ by taking into account the leaf outcome class prototypes (i.e., $\mathcal{L}_S$ and $\mathcal{L}_R$) of each taxonomy. The first function, $\mathcal{D}_1$, is defined as:

$$\mathcal{D}_1(\mathcal{T}_S, \mathcal{T}_R) = \frac{1}{|\mathcal{L}_S|} \sum_i^{|\mathcal{L}_S|} d(O_i^S, BestMatch(O_i^S, \mathcal{L}_R))$$

Intuitively, the function $\mathcal{D}_1$ can be interpreted as asking the

question of whether tool R produces the same outcomes as tool S. If $\mathcal{T}_S$ and $\mathcal{T}_R$ are identical, then $\mathcal{D}_1(\mathcal{T}_S, \mathcal{T}_R) = 0.0$. However, the distance function $\mathcal{D}_1$ is not symmetric, i.e., $\mathcal{D}_1(\mathcal{T}_S, \mathcal{T}_R) \neq \mathcal{D}_1(\mathcal{T}_R, \mathcal{T}_S)$, which is why another function, $\mathcal{D}_2$, was defined and used to compute the distance between two outcome taxonomies:

$$\mathcal{D}_2(\mathcal{T}_S, \mathcal{T}_R) = \frac{1}{2}\mathcal{D}_1(\mathcal{T}_S, \mathcal{T}_R) + \frac{1}{2}\mathcal{D}_1(\mathcal{T}_R, \mathcal{T}_S)$$

The distance measure $\mathcal{D}_2$ is symmetric and only takes into account the outcome class prototypes at the leaves in each taxonomy. $\mathcal{D}_2$ was used in all experiments to compute the distance between two taxonomies acquired through experience with two different tools.

D. Prediction of Outcomes Using The Taxonomy

While the task of prediction is not the central theme investigated in this paper, we briefly overview how the learned taxonomy of outcomes $\mathcal{T}$ can be incorporated into a learning framework that allows the robot to anticipate the outcomes of its actions with the tool.

Let $X_i = (B_i, C_i)$ be defined as an input data point indicating that the robot is executing behavior B_i while detecting perceptual cues C_i. The task of the robot is to learn a predictive model $\mathcal{M}(X_i) \rightarrow \hat{P}_i$ such that for a given data point X_i, the model returns $\hat{P}_i$ which is the predicted path from the root node to a leaf node in the taxonomy $\mathcal{T}$. This path indicates how the yet unobserved outcome O_i will be classified into the taxonomy. Once the robot executes the behavior B_i and observes the outcome O_i, the predicted path $\hat{P}_i$ can be compared with the actual path P_i and the quality of the prediction can be evaluated. In the machine learning literature, this problem is known as *hierarchical classification*, since the class labels are hierarchically structured [14].

While there are many algorithms developed to address the incremental hierarchical classification problem (see [14] for a review), the framework presented here uses a simple solution: each non-leaf node v_j in the taxonomy $\mathcal{T}$ has an associated predictive model M_j that is trained to predict the child outcome class of the (yet unobserved) outcome O_i associated with X_i. Formally, for a non-leaf node v_j, $M_j(X_i) \rightarrow \hat{v}_k$ where $\hat{v}_k$ is a child node of v_j in $\mathcal{T}$. For example, the root node in Fig. 4.a contains a model, M_0, which given an input data point X_i, predicts whether the outcome O_i falls within v_1 or v_2. Thus, applying a recursive top-down prediction routine results in a predicted path from the root node to a leaf node in the tree.

Each model M_j is realized by an ensemble of classifiers for incremental learning as proposed in [15], with a C4.5 decision tree for each classifier in the ensemble [16]. The performance of $\mathcal{M}$ is reported in terms of the normalized H-Loss function as defined by Cesa-Bianchi *et al.* in [14]. The intuition behind the normalized H-Loss function is that wrong predictions should be penalized according to the depth in the
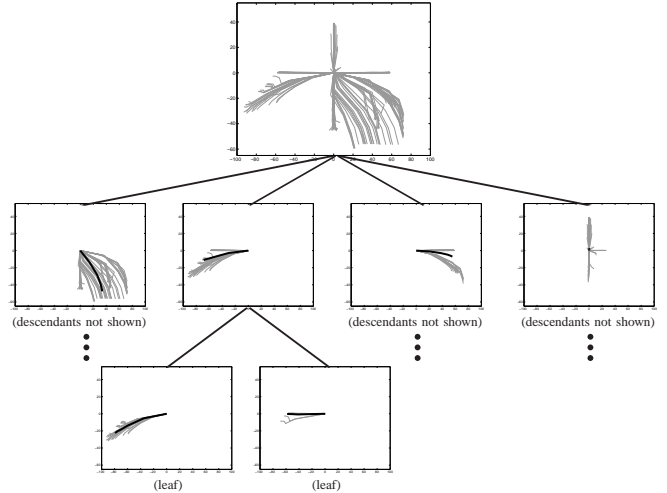

Fig. 5. Partial visualization of the learned outcome taxonomy for the L-Stick tool after 1200 trials. For each outcome class v_j the darker trajectory denotes the outcome prototype O_j^{mean}, while the lighter trajectories visualize the observed outcomes from the test set of trials that fall within v_j.
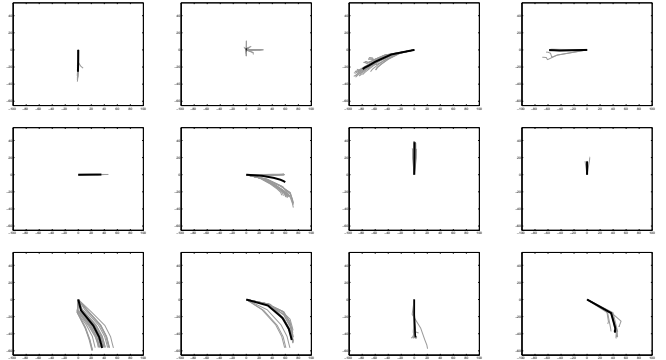

Fig. 6. All twelve leaf outcome classes of the learned taxonomy for the L-Stick tool (shown in Fig. 5). The dark trajectory shows the outcome prototype for each leaf class in the learned taxonomy, while the lighter trajectories visualize the observed outcomes that fall within v_j.

taxonomy at which they occur. An H-Loss of 1.0 indicates that the estimated path $\hat{P}_i$ diverges from the true path P_i at the very root of the taxonomy $\mathcal{T}$, while an H-Loss of 0.0 indicates perfect path prediction. For more details on the precise mathematical formulation of this function, see [14].

V. EXPERIMENTAL RESULTS

A. Exploring Individual Tools

In the first experiment the robot performs 1200 trials with the L-Stick tool and incrementally updates the outcome taxonomy $\mathcal{T}$ and predictive model $\mathcal{M}$ after each trial. Fig. 5 shows a partial visualization of the acquired taxonomy of outcomes after all 1200 trials are completed. Each trajectory is plotted relative to the puck's starting location which is randomly chosen in relation to the tool during each trial. The first level of the taxonomy (i.e., the four child nodes of the root) is created after the 300th trial.
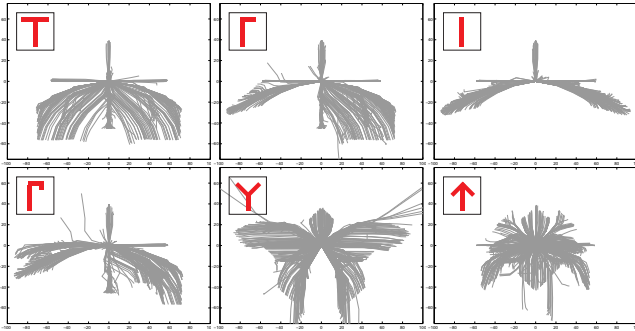
Fig. 7. A visualization of the types of movement trajectories of the puck that the robot can induce with its behaviors for each of the 6 tools. All trajectories are plotted relative to the puck's starting location, which is chosen randomly for each trial.

The root node contains all observed outcomes, each plotted as a trajectory of the puck's detected movement relative to its starting position as detected in the robot's camera image. The visualization of the outcomes shows that with the L-Stick tool the robot can push the puck forward, pull it backward, as well as slide it left and right. When applying the *rotate-right* behavior, the robot is able to bring the puck closer. However, when rotating the tool to the left, the puck moves mostly sideways. In roughly half of the trials the puck does not move at all due to initial configurations in which the robot's behavior with the tool does not affect the puck.

The learned outcome taxonomy for the L-Stick tool contains 12 leaf classes, shown in Fig. 6. The acquired leaf concept classes show that similar movements of the puck are indeed grouped together. In addition, the robot is able to form concept classes for outcomes that represent little or no movement of the puck.

The same experiment was repeated with all six tools. Fig. 7 visualizes the movement trajectories that the robot can induce on the puck with each tool. Table I shows the number of leaf concept classes in each taxonomy. The comparison shows that different tools produce taxonomies of varying complexity, e.g., the Stick tool produces the most simple taxonomy while the Y-Stick and the T-Stick tools produce the most complex ones.

Table I also shows the performance measures of the models $\mathcal{M}$ for each tool, which are obtained by evaluating each model on 600 novel trials (with the same tool) not previously seen by the robot. For most tools, the normalized H-Loss [14] is low, indicating that the robot is able to form accurate predictive models that can anticipate outcomes. Simpler taxonomies result in better prediction performance, e.g., the Stick tool produces the most predictable outcomes.

B. Estimating the Functional Similarity Between Tools

After performing 1200 trials with each of the six tools, the robot uses the previously defined distance measure, $\mathcal{D}_2$, to

SUMMARY OF THE LEARNING RESULTS FOR A TAXONOMY $\mathcal{T}$ AND A PREDICTIVE MODEL $\mathcal{M}$ FOR EACH OF THE SIX TOOLS.

Tool	Number of leaf outcome classes in $\mathcal{T}$	Normalized H-Loss of the predictive model $\mathcal{M}$
T-stick	18	0.165
L-stick	12	0.084
Stick	7	0.013
L-Hook	14	0.085
Y-Stick	19	0.198
Arrow	14	0.133

infer the functional similarity between different tools. Fig. 8 shows the computed distance matrix for all pairs of tools.

As expected, Fig. 8 shows that the inferred distance between identical tools is 0.0. The two most similar tools are the L-Stick and the L-Hook, which is not surprising considering that their shapes are almost identical. Their functional similarity is also evident from Fig. 7 which shows that the L-Stick and the L-Hook tools produce nearly identical outcomes. The two most distant tools are the Stick and the Y-Stick. The Y-Stick tool is distant from almost all tools, with the exception of the Arrow tool, with which it is most similar.

Fig. 9 visualizes the estimated distance measurements between the tools, by embedding the distance matrix onto a two-dimensional plane using the Isomap method [17]. The figure shows that the differences between the T-Stick, L-Stick, L-Hook, and Stick tools, estimated with the $\mathcal{D}_2$ distance measure, can be accurately described using a single dimension, i.e., the data points for these tools lie on a line in the 2-D embedding. Unlike those four tools, the Y-Stick and Arrow tool contain diagonal segments, which might be why they do not lie on the same line in the 2-D embedding.

	T	Γ	I	Γ	Y	↑
T	0.0	9.73	17.69	10.44	21.95	16.39
Γ	9.73	0.0	14.96	8.72	25.76	19.98
I	17.69	14.96	0.0	13.11	29.58	25.04
Γ	10.44	8.72	13.11	0.0	23.71	18.06
Y	21.95	25.76	29.58	23.71	0.0	16.90
↑	16.39	19.98	25.04	18.06	16.90	0.0

Fig. 8. Distance matrix computed by applying the $\mathcal{D}_2$ distance measure between each pair of taxonomies acquired by the robot from experience with each of the six tools.
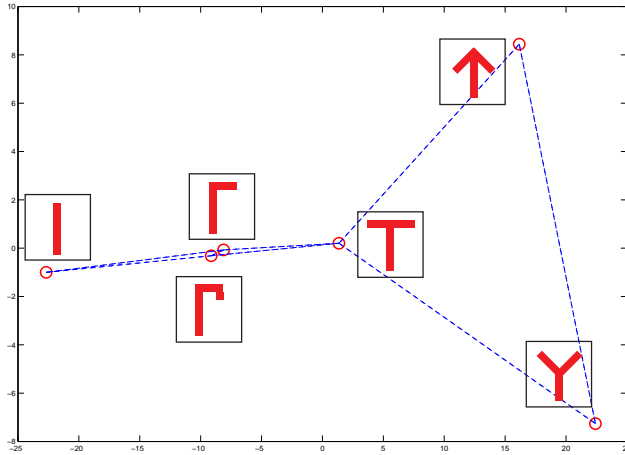
Fig. 9. Two dimensional isomap embedding (with neighborhood graph) of the distance matrix shown in Fig. 8 which describes the similarity between the six different tools.

VI. CONCLUSIONS AND FUTURE WORK

This paper presented a framework in which the robot incrementally learns an adaptive hierarchical representation for the types of outcomes it can induce on an environmental object through its actions with different tools. The model allows the robot to form discrete outcome classes *without* a priori knowledge of the underlying distribution of outcomes. Unlike previous work, the model is adaptive, allowing the robot to update the outcome class prototypes as well as to form novel classes of outcomes as a result of new experience.

The results showed that the robot can learn accurate and compact models for the types of outcomes observed with each tool. The compact outcome model for each tool allows the robot to use standard machine learning methods for prediction. With this ability the robot can select a behavior in order to achieve some desired outcome with the tool.

The learned outcome models also allowed the robot to infer the functional similarity between different tools. The robot was able to detect tools that were very similar (e.g., the L-Stick and the L-Hook tools) and tools that are very different in terms of the outcomes they produce. The distance measure between two tools took into account the functional properties of the tool and the results indicate that there is a strong relation between the shape and the functional similarity between two tools.

There are several directions which may be pursued for future work. First, the ability of the robot to infer the similarity between tools can be used to estimate what a novel tool affords the robot. For example, given a set of explored tools, the robot can start to relate common perceptual features of the tools (e.g., shape, color, etc.) to common functional properties. This will allow the robot to estimate the functional similarity between a familiar tool and a novel tool based on relevant visual features.

Second, the ability to incrementally form concept hierarchies can be extended to the robot's behaviors and cues. This would allow the robot to learn a model that captures how the learned concept classes of outcomes, behaviors, and cues relate to each other. The taxonomy learning algorithm can also be improved by considering more substantial changes to the structure of the taxonomy as a result of new data (e.g., node merge, sibling addition, etc.).

Finally, the learned models for each tool can be compared in a way that allows the robot not only to estimate a distance measure between two tools, but also to infer what these differences are. While in the current framework the robot compares the tools based on the environmental outcomes they produce, the comparison can be generalized to include other factors, such as the behavioral and perceptual aspects of the acquired model for each tool.

REFERENCES

[1] B. B. Beck, *Animal Tool behavior: The use and manufacture of tools by animals*. New York: Garland STMP Press, 1980.
[2] L. Bogoni, "Identification of functional features through observation and interactions," Ph.D. dissertation, U. of Pennsylvania, 1995.
[3] M. Sutton, L. Stark, and K. Bowyer, "GRUFF-3: generalizing the domain of a functiona-based recognition system," *Pattern Recognition*, vol. 27, no. 12, pp. 1743–1766, 1994.
[4] E. Rivlin, S. Dickinson, and A. Rosenfeld, "Recognition by functional parts," *Comp. Vision and Img. Understanding*, vol. 62, no. 2, 1995.
[5] G. Froimovich, E. Rivlin, and I. Shimshoni, "Object classification by functional parts," in *Proceedings to First International Symposium on 3D Data Processing, Visualization and Transmission*, 2002.
[6] C. C. Kemp and A. Edsinger, "Robot manipulation of human tools: Autonomous detection and control of task relevant features," in *Proc. of 5th IEEE Int. Conf. on Development and Learning (ICDL)*, 2006.
[7] A. Stoytchev, "Behavior-grounded representation of tool affordances," in *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2005, pp. 3071–3076.
[8] J. Sinapov and A. Stoytchev, "Learning and generalization of behavior-grounded tool affordances," in *Proc. 7th IEEE Int. Conf. on Development and Learning (ICDL)*, 2007.
[9] E. Sahin, M. Cakmak, M. Dogar, E. Ugur, and G. Ucoluk, "To afford or not to afford: A new formalization of affordances toward affordance-based robot control," *Adaptive Behavior*, vol. 15, no. 4, pp. 447–472, 2007.
[10] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: From sensory-motor coordination to imitation," *IEEE Transactions on Robotics*, vol. 24, no. 1, 2008.
[11] R. Smith, "The open dynamics engine (ODE) user guide," http://ode.org/, 2003.
[12] W. Schenck and R. Moller, *Anticipatory Behavior in Adaptive Learning Systems*. Springer, 2007, ch. Training and Application of a Visual Forward Model for a Robot Camera Head, pp. 153–169.
[13] D. Pelleg and A. W. Moore, "X-means: Extending k-means with efficient estimation of the number of clusters," in *17th Int. Conf. on Machine Learning*, 2000, pp. 727–734.
[14] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Incremental algorithms for hierarchical classification," *Journal of Machine Learning Research*, vol. 7, pp. 31–54, 2006.
[15] M. D. Muhlbaier and R. Polikar, "An ensemble approach for incremental learning in nonstationary environments," *Lecture Notes in Computer Science*, vol. 4472, pp. 490–500, 2007.
[16] J. R. Quinlan, *C4.5: programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.
[17] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.