To appear in *Proceedinges of the Twelfth International Joint Conference on Artificial Intelligence* (1991). Sydney: Morgan Kaufmann.

Learning to Classify Observed Motor Behavior

Wayne Iba*
AI Research Branch, Mail Stop 244-17
NASA Ames Research Center
Moffett Field, CA 94035, USA

Abstract

We present a representational format for observed movements The representation has a temporal structure relating components of a single complex movement. We also present OXBOW, an unsupervised learning system, which constructs classes of these movements. Empirical results indicate that the system builds abstract movement concepts with appropriate component structure allowing it to predict the latter portions of a partially observed movement.

1 Introduction

Motor control is a necessary facet of an intelligent agent's interaction with the environment. Until recently, this topic has been largely ignored. We are encouraged by the interest demonstrated by a number of recent efforts aimed at learning sequences of operators that can control effectors external to the learning agent (Laird, Hucka, Yager, & Tuck, 1990; Mason, Christiansen, & Mitchell, 1989; Moore, 1990). However, we believe that a general phenomenon in human motor behavior is that generation is limited by understanding; that skilled movement behavior must be understood or recognized before it can be generated. In this paper we present OXBOW, a system that acquires knowledge structures that facilitate the recognition of observed movements. This system is the recognition component of MÆANDER, a larger model of motor-skill learning that also involves the generation of movements (Iba & Langley, 1987).

We view OXBOW within the paradigm of *concept formation* (Fisher & Pazzani, in press), in that it is required to construct "concepts" for particular classes of movements it has experienced. The current work is a descendent of Gennari's CLASSIT (1990) and, in turn, of Fisher's COBWEB (1987). CLASSIT extended COBWEB to deal with attributes having continuous values and OXBOW extends CLASSIT to deal with structured objects with differing numbers of components. The common thread in OXBOW, CLASSIT, and COBWEB is the control structure of the learning algorithms and the operators used to modify the knowledge structures. Also, they all use conditional probabilities in calculating the *category utility* function (Gluck & Corter, 1985). This paradigm assumes that storing the probabilities for a set of instances is sufficient to summarize the set and to distinguish among members of different classes.

*Also affiliated with RECOM Technologies.

2 The OXBOW System

When discussing a system that learns from movements, one must necessarily consider the interface between the learning system and the environment where action occurs. In this paper we assume that this action is generated by a jointed limb, and that the positional information for each of the joints is available to the system. We assume that a sensory process observes continuous motion and parses this into a *motor schema* (described below). Our parsing process is based on Rubin and Richards' (1985) theory of elementary motion boundaries.[1] We also assume that time is discretized at a suitably fine level of granularity. Since in this paper we focus on learning to recognize movements, we do not address the motor interface that causes effectors in the environment to move according to stored representations of movement. However, the analog to the motor interface for our larger system, MÆANDER, allows it to simulate the execution of stored movements in the "mind's eye". This provides the basis for the performance measures in the experiments discussed later.

Given this framework for the system, the recognition task is: given a motor schema representing a parsed movement, find the class of previously stored schemas that has the most similar trajectory to the observed movement. Learning involves updating and restructuring memory to facilitate this process; measuring performance involves determining the similarity between the indexed class and the observed movement.

2.1 Representing movements as probabilistic motor schemas

We define a *motor schema* to be a sequence of state descriptions that describe the status of the arm at particular times. In turn, a state description specifies the positions and velocities for each of the joints of an arm at a specific time (relative to the time of the movement). These state descriptions are within a two-dimensional Cartesian frame with the origin located at the base of

[1] Iba and Gennari (in press) provide a more detailed description of OXBOW, including our parsing, performance, and learning mechanisms.

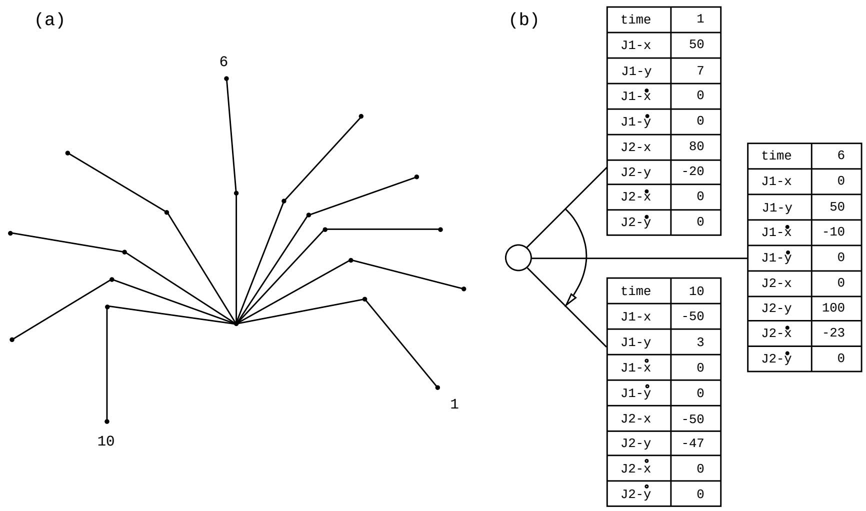

Figure 1. A pictorial rendition of (a) a simple movement and (b) OXBOW's representation of the associated motor schema.

the arm. The sequence of state descriptions comprising a motor schema is *temporally ordered* according to the time values in each state description.

Figure 1 shows a pictorial rendition of a movement observed in the environment along with a corresponding motor schema. The positions of the arm in Figure 1(a) are sampled from equal time slices during the course of the movement. Note that the movement shows the position of the arm at every time step. In contrast, motor schemas specify arm positions only at a few time steps for a given movement. Thus, the schema shown in Figure 1(b) *represents* the movement shown in Figure 1(a), but only *specifies* information for the arm at three times. In our framework, movements and schemas are closely related. Although movements occur in the environment, OXBOW classifies and stores the parsed form of movements – motor schemas.

When motor schemas are combined to form abstractions or generalizations, we think of the resulting structures as *concepts*. One way to represent concepts in this type of model is to use probabilities (Smith & Medin, 1981). A *generalized motor schema* includes a probability distribution for each attribute describing the state of the arm, as well as a probability for each entire state. That is, OXBOW stores the conditional probability distribution of an attribute given a particular state description, and stores the conditional probability of each state description given a particular schema concept.

Given this format for representing motor schemas, these structures must be organized to allow retrieval of stored schemas. In OXBOW, motor schemas are organized into an IS-A hierarchy. Nodes in this hierarchy are partially ordered according to their generality, with concepts lower in the hierarchy being more specific than their ancestors. Thus, the root node summarizes all instances that have ever been observed, terminal nodes often correspond to single instances, and intermediate nodes summarize clusters of observations. Fisher and Langley (1990) review arguments for organizing probabilistic concepts in such a hierarchy.

It is important to note that our representation of motor schemas as a sequence of state descriptions implies a structure on schema concepts. Each state description is viewed as "part of" the entire motor schema. This significantly complicates the concept formation task. As a further complicating factor, two motor schemas will often have a different number of states. In partial response to this problem, OXBOW stores and organizes state descriptions in an internal hierarchy of state descriptions within each schema node. Thus, each node in the schema hierarchy has its own private state hierarchy. The top level of this hierarchy represents the PART-OF relations between the multiple states and the schema as a whole. That is, the classes of states at the top level of the state hierarchy are the state descriptions comprising the motor schema and are ordered according to the values for the time attribute in each respective state description.

Figure 2 shows a possible hierarchy of baseball-pitching schemas. The leaf nodes of the tree represent the motor schemas from specific observed pitches. The node labeled as "overhand" represents a generalization of the three specific throws stored below it in the hierarchy. This generalization is also a motor schema, but instead of specific values, the generalization stores means and variances for each of the attributes in its state descriptions. Looking more closely at the OVERHAND schema in Figure 2, we see the internal state hierarchy that captures the structure of the abstract schema. We also show the first state of this generalized motor schema where means and standard deviations have been stored for each of the attributes. These values summarize the states that have been classified together. Now, let us turn to a description of how these structures are constructed and modified based upon experience.

2.2 Forming movement concepts

The performance task stated above is closely related to OXBOW's learning task: given an experienced movement, modify the current knowledge base according to the resulting classification. Movements are observed sequen-

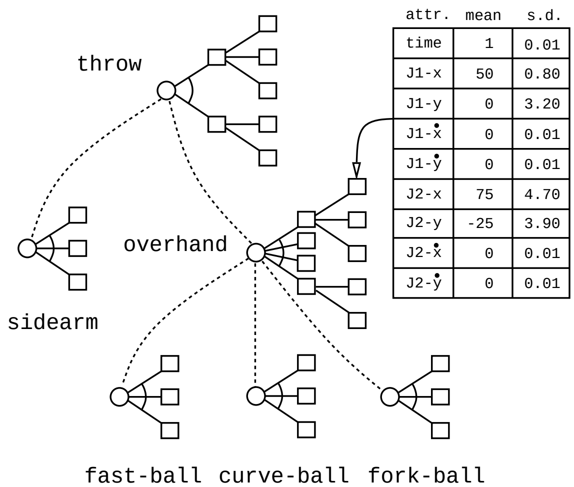

Figure 2. A hierarchy of motor schemas for baseball pitches, with one node shown in detail.

tially and are not labeled; this corresponds to an incremental, unsupervised learning task. The distinction between performance and learning in OXBOW (and similar systems) is slight and only serves to clarify a means for evaluating the overall system's ability to acquire and organize movement information. We will return to this in Section 3; here we focus on the learning method.

2.2.1 The classification mechanism

Table 1 presents the basic OXBOW learning algorithm; at this level of description it is functionally equivalent to that used in Fisher's (1987) COBWEB and Gennari's (1990) CLASSIT. In these concept formation systems, the classification and hierarchy formation processes are tightly coupled. Upon encountering a new instance I, the system starts at the root and sorts the instance down the hierarchy, using an evaluation function (described below) to decide which action to take at each level. This recursive algorithm terminates when the instance has been recognized. This occurs either when the current node is a leaf (has no children), or when the evaluation function already has a sufficiently high value that further efforts are deemed unnecessary.[2] When an instance has been recognized, the current node is returned as the classification value for the instance.

At a given node N where the instance I is still unrecognized, the algorithm decides among four options. First, the instance is temporarily incorporated into a child of N, one at a time. The resulting partitions are evaluated and the best is selected. Second, the instance is placed in a new singleton class and this new partition is also selected. The third and fourth candidates generated by the *merge* and *split* operators are intended to allow the system to recover from poor hierarchies that may result from peculiar training orders; Fisher (1987) gives details on these operators. The algorithm uses its evaluation function to determine which of the resulting four partitions is "best", and then either continues by recursively classifying the chosen best, or stopping and

[2] This decision is based upon the *recognition criterion* described in Gennari (1990).

Table 1. The OXBOW algorithm.

```
Function Oxbow(instance, concept),
if leaf?(concept) or recognized?(instance,concept)
   then stop and return(concept)
else for each child of concept
   compute score for Incorporate(instance,child)
   let best be the child with the best score
   let second be similarly the second best
compare best with the evaluations of:
      create new child with instance and call it
self
      merge(best, second)
      split(best)
let selected-child be the best respective
evaluation
if selected-child = self
   then stop and return(concept)
else call oxbow(instance, selected-child)
```

returning the current node as the classification whenever the instance is placed in a new child.

2.2.2 The evaluation function

We have mentioned that OXBOW uses an evaluation function to determine the appropriate branch to sort new instances down during classification. Since a major goal of concept formation is to let the agent categorize new experience and make predictions, the system employs category utility – an evaluation function that attempts to maximize predictive ability. Gluck and Corter (1985) originally derived this measure from both game theory and information theory in order to predict basic-level effects in psychological experiments, and Fisher (1987) adapted it for use in his COBWEB model of concept formation. The measure assumes that concept descriptions are probabilistic in nature, and it favors clusterings that maximize a tradeoff between intra-class similarity and inter-class differences. Initially formulated for nominally valued domains, Gennari (1990) extended the equation to handle attributes with continuous values. In turn, we have extended the equation to handle representations where objects consist of differing numbers of components. This version of category utility for real-valued attributes and structured objects can be stated as

$$\frac{\sum_k^K P(C_k) \sum_j^J P(S_{kj}) \sum_i^I \frac{1}{\sigma_{kji}} - \sum_m^M P(S_{pm}) \sum_i^I \frac{1}{\sigma_{pmi}}}{K} ,$$

where $P(C_k)$ is the probability of class C, K is the number of classes at the current level of the hierarchy, $P(S_{kj})$ is the probability of the jth state description in C_k, $P(S_{pm})$ is the probability of the mth state description in the parent of the current partition, σ_{kij} is the standard deviation of attribute i in the jth state of the kth class, and σ_{pmi} is the standard deviation for attribute i in the mth state of the parent node.[3]

[3] The value of $1/\sigma$ is undefined for a single instance with zero variance so an *acuity* parameter serves as the minimum variance. Gennari (1990) presents an empirical study of the this parameter's impact on performance.

2.2.3 Incorporating motor schemas

Every concept formation system must address the issue of how to create an abstraction from two items – in this case a new instance and an existing class. The abstraction, or the result of incorporating a number of instances, allows useful predictions to be made in the future. Because COBWEB assumed that every instance had values for every attribute, incorporating a new instance was a simple matter of incrementing appropriate attribute-value counts in the concept node according to the values in the instance. CLASSIT extended this to allow objects made up of multiple components, but each had to have the same number. That is, there was a single structure for the objects in every class. The problem for OXBOW is that motor schemas may have differing numbers of components, and neither COBWEB or CLASSIT have satisfactory mechanisms for handling this variation of the task. In this situation, there is not necessarily a one-to-one correspondence between states from one schema to the next. Therefore, it is not possible to uniquely associate the attributes (at the state description level) from one schema to another.

OXBOW's solution to this correspondence problem uses category utility to form a hierarchy of state descriptions based only on the time attribute. By treating each state in a schema instance separately in this fashion, an effective mapping is established between states in the instance and states stored in memory. This mapping may be many-to-one or one-to-none.[4]

Thus, the system classifies both schemas and states, using the same algorithm given in Table 1, but with two important caveats. First, if the instance is a schema, then each of its states is incorporated in the hierarchy of state descriptions associated with the current schema node. But if the instance is a state, then there is no correspondence problem and the incorporation is done the same as in CLASSIT or COBWEB. The second distinction is that if the instance is a state, then the evaluation function used in the algorithm is simplified. Because we are interested in capturing the temporal structure that is present in the schema instances, only the time attribute is considered instead of summing over all the attributes to determine the score. However, all of the attributes that describe a state are updated when a new state is incorporated.

Since schemas are made up of the top-level nodes in the state hierarchy, we may think of the first level of this hierarchy as capturing the PART-OF structure for the schema. The PART-OF relation refers to an outer context, in this case the entire motor schema. Because OXBOW's motor schemas are presented as parsed structures that are related temporally, it can break the schema into its parts, treat the parts separately, and then pop back up to the context of the schema where the work of matching the parts has already been done.

[4]A one-to-none mapping occurs when a state in the instance does not correspond to any existing states in the concept and a new state is added to the schema concept.

3 The Performance of OXBOW

OXBOW provides a method for representing jointed limb movements and for acquiring a concept hierarchy of movement concepts. Naturally, before we can make conclusions about the usefulness of such a system we need to know how well the system operates and improves on some performance task. In this section we present our performance measure and several experiments demonstrating that OXBOW can recognize (and improve its recognition of) observed movements.

The performance task for OXBOW is to *classify* a newly presented movement given some existing concept hierarchy. As discussed above in the context of learning, this involves associating the new instance with a node in the hierarchy representing previously observed movements that are most similar to the new movement. In humans, classification and learning occur simultaneously. However, we have implemented OXBOW to allow classification without modifications to the concept hierarchy. That is, we use a trimmed version of the learning algorithm which does not consider tree modification operators and which does not alter the contents of the nodes in the tree.

We evaluate the system's performance on the above task by comparing a test instance to the movement described at the node of the schema hierarchy where the test instance is classified. This comparison is performed by finding the Euclidian distance between corresponding joints of the arm at a given time. We average over the joints of the arm and over the testing movement period. The error scores we report in the following experiments reflect this averaging over joints and simulated time slices. The units are for an arm with two joints operating in a reachable workspace of 100 unit radius.

3.1 Experiment 1: Learning single movement concepts

Recall that OXBOW's representation of movements (as schemas) consists of parts (states) and the temporal relationships among them. The learning algorithm considers schemas first as a set of individual states at the state hierarchy level and then as a sequence at the schema hierarchy level. One of the first things we should verify is that the inner treatment – the determination of the PART-OF structure for the movement concept at large – is behaving appropriately. Therefore, in our first experiment we trained the system on instances from a single concept, so as to control for possible confusions between movements of different types. This should also let us determine how sensitive the system is to variance in the observed data.

To this end, we created four artificial movements that correspond to a slap, a throw, a wave, and a salute. Observed movements in this experiment (and those following) were produced by motor schemas instantiated from the templates. The time, position, and velocity values in a motor schema used for training were drawn from normal distributions given in a template. We adjusted the variance of the distributions by a scalar to produce data sets with different levels of variability.

In this experiment, a single run tested one movement

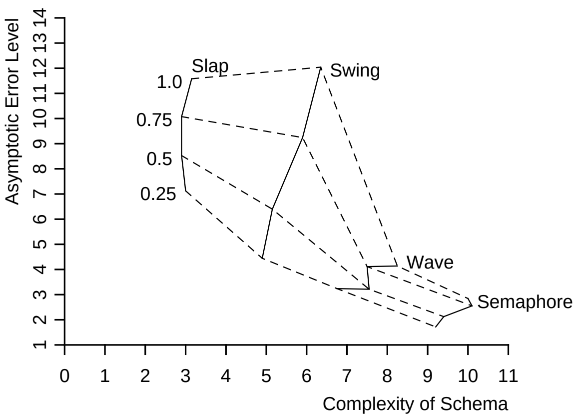

Figure 3. A comparison between asymptotic error rates when trained on single movement types, and the complexity of input data at different levels of variation.

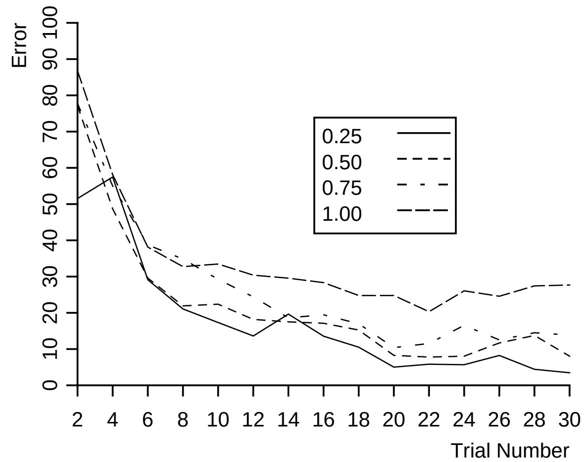

Figure 4. Learning curves when trained on instances from all four classes at four levels of input variation.

type over 20 blocks with 15 learning trials in each block. We repeated runs at four different levels of variation for each of the four movement types and measured the system's performance after every other learning trial. The performance metric compares the prototype with the schema stored at the root node of the schema hierarchy. Because there is only one movement type presented in a run, and the root represents the summary over all the observed instances, this comparison lets us control for possible retrieval problems. The results indicated that error decreased as a function additional movement observations for each of the schema types at all levels of variation. However, we note that the asymptotic levels increase as a function of variability. These results indicate that Oxbow has trouble finding the central tendency in domains with high variance. Because the data comes from a single prototype, we would expect that the prototype would be recoverable. This could either be due to problems determining the values within the states of the learned motor schemas, finding the correct structure of states themselves, or a combination of both.

To help clarify this issue, Figure 3 shows the asymptotic error levels for each movement type as a function of the structural complexity inherent in the data. Additionally, it shows how the asymptote and complexity changes for the different levels of variation in the data. We define complexity as the number of states in a parsed description of an observed movement. For a given movement type and a single level of noise, we computed the average complexity over 20 randomly generated movement instances.

From this graph we see that changing the variability in the generated movements does not cause large changes in the structure of the parsed movements. We can conclude that the increased asymptote levels as a function of increased variation are not a result of a failure to determine the appropriate part-of structure for the movement concept. but rather to problems determining the correct values within the states. We are looking at ways to improve this situation. However, this figure also reveals an unexpected result – that increasing complexity tends to decrease asymptotic error level. This non-intuitive result is not without precedent; for instance, vision researchers found that more complexity in the environment made things easier to disambiguate. This would suggest that Oxbow could scale up to more complex environments and movements. We are currently considering these issues further to determine where to assign credit and blame, and how extensible our methods are.

Overall, this experiment indicates that Oxbow is able to capture the part-of structure found in observed movements when faced with only a single type, but that its ability to form accurate state descriptions is influenced by variability. Now let us turn to the larger problem of acquiring movement concepts from observed data when presented with instances of more than one type.

3.2 Experiment 2: Learning multiple movement concepts

If we had first tested Oxbow on acquiring multiple concepts simultaneously, we would not have known whether performance errors were caused by confusions between categories when classifying an observed movement, by problems identifying the appropriate part-of structure for a particular node in the hierarchy, or both. The previous study established a baseline for comparison. We can expect that errors above and beyond those reported in the previous section are a result of classification errors.

To study such errors, we ran a second experiment in which Oxbow observed movements from all four of the classes, each with an equal likelihood. We presented 30 training instances, from which the system constructed its hierarchy of movement concepts. After every other training instance, we stopped learning and tested the system's performance as described above. We repeated this experiment at the same four levels of variation in the movement generators. Figure 4 shows the average error (over the four movement types), again as a function of experience and variance level. The errors are averaged over ten blocks of 30 training instances with random orders of the movement types.

The figure gives some indication of how well Oxbow is distinguishing between observed movements of different types, even though after 30 training instances the system has only seen (on average) one fourth that many

Table 2. Asymptotic error rates for two training modes

	Variability level			
	1.0	2.0	3.0	4.0
Separate Training	4.20	5.48	6.74	8.04
Mixed Training	3.46	8.03	13.88	26.20

training instances of each type. Table 2 shows a comparison between the performance after 30 training instances from Figure 4, under mixed training, and the performance level after seven training instances in Figure 3, under separate training. This reveals an interaction between the training method (mixed/separate) and the variability in the domain; increased variability has a much greater impact on error when learning multiple concepts than when learning single concepts. In Experiment 1, we were able to control for retrieval errors because only a single movement type was given. In this experiment, we relied on Oxbow's retrieval mechanism and therefore increased errors could be attributable to misclassifications during either training, retrieval, or both. One option to clarify this would be to repeat Experiment 1 and rely on Oxbow's retrieval mechanism, but another option is to evaluate the retrieval mechanism in isolation as we report in the next section.

3.3 Experiment 3: Predicting unseen movement

In the previous experiments, the performance measure corresponded to what has been termed *recognition* in the psychological literature. That is, the complete prototype of a particular movement class was classified and a comparison was made across the entire duration of the movement. In real life, one would more likely observe a partial movement and need to predict the continuation of the movement. Furthermore, this puts a strain only on the retrieval mechanism, allowing an assessment of its performance.

Thus, in a third experiment, we trained Oxbow as described before, but we altered the performance mechanism slightly. When testing, we presented only a portion of the prototype movement and we measured error over the remaining unobserved movement. Because we average over time when determining error, we can compare errors among different ratios of observed movement to predicted movement, even though the number of predictions changes. Any differences in errors will be attributable to classification problems during retrieval because the knowledge base is the same for each level of observation at a given point in training.

This formulation of the task suggests a prediction: as less of the movement is observed, classification should become more difficult and mistakes should lead to greater measured error. Simply stated, the more one is able to observe, the less uncertainty there should be about succeeding movement. Figure 5 shows the learning curves from an experiment in which we varied the portion of the movement to be predicted. We fixed the variation level at 0.5 and averaged the results over ten blocks of 30 training instances each.

The figure shows that when Oxbow is predicting 80%

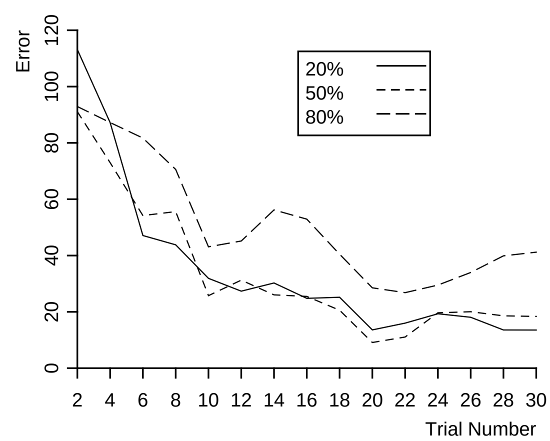

Figure 5. Learning curves showing error while varying the amount of the test movement that is missing and must be predicted.

of the movement (observing only the first 20% of the movement) the errors are consistently the highest (except very early in training, when not all the movement types have yet been seen). However, there is little difference between observing 50% of the movement and observing 80%. This result suggests that the system is not adversely affected by having less information available for classification except in extreme cases like the 80% condition. This leads to another prediction: as the training data becomes more variable, the system should require larger portions of the test movement in order to prevent the error from increasing. We intend to test this prediction in future work.

More important, this experiment holds other factors constant while varying the amount of information in the test movement, thus indicating the sensitivity of the classification process during retrieval. The results suggest that Oxbow is not readily making misclassifications when given partial structures in the input. Again, this points towards problems with the mechanism for storing state descriptions when learning from new experience. We are pursuing avenues to clearly establish and to improve this situation.

4 Discussion

We designed the Oxbow system to address issues in both concept formation and the recognition of motor behaviors. With respect to the first area, we must conclude that although Oxbow presents a novel approach to acquiring structured concepts, it has two drawbacks. Although the system shows significant improvement when learning multiple concepts, the interaction effect in Experiment 2 shows that Oxbow has difficulties forming generalized motor schemas from schemas having significant variation among them. Second, the natural extension to objects with multiple levels of structure would require prohibitive amounts of memory and processing. However, we believe that the significant structural aspect of representing movements is timing information, so that only a single level of components is necessary in this domain. With respect to motor behavior, Oxbow represents the first system that specifically addresses the recognition of movements, so it is difficult to asses its weaknesses in this area. However, the current implementation is sensitive to the values in the state descriptions with respect to scale and rotation of the observed action. We have several ideas on a normalization mechanism to address these problems and do not view them as fundamental flaws.

In addition to extending the implementation to normalize the representation of movements for differences in scale and rotation, we plan to apply OXBOW, as a movement recognition system, to the task of signature recognition. Because our approach is intimately tied to movements and not to static pattern recognition, we think there are potential advances in this direction. There also is a direct application of movement recognition in domains where humans and robots are working together; the artificial agent should be able to infer goals and plans held by the human from observing and recognizing her actions. Following on our efforts at modeling human motor behavior (Iba & Langley, 1987), we seek to connect OXBOW's predictions about movement recognition to the psychological literature on this subject. As a method for concept formation, OXBOW suggests a connection between IS-A and PART-OF relations by using a concept formation mechanism to extract the structure of a composite concept. We want to analyze this connection to determine its implications for concept formation in domains with more complex structure. As part of the ongoing experimentation mentioned in the previous section, we also intend to compare OXBOW with other approaches to forming structured concepts.

In closing, OXBOW makes an important contribution to the area of concept formation; it provides an interesting approach to the problem of acquiring concepts from objects composed of parts. The experiments indicated that the system was able to extract appropriate PART-OF structures, but it was not immune to confusions in domains with high variability. More important, as a motor skill system, OXBOW provides an approach to representing, recognizing, and organizing motor skills that are observed by an agent. In addition, the method is flexible with respect to environmental characteristics such as the number of joints in an arm or the number of arms operating in concert. Furthermore, the system's representation and organization carries over smoothly from recognition to generation of motor skills (Iba & Langley, 1987). We hope that this effort will contribute to the growing interest in motor control, but also that it will spark new interest in treating motor behavior as a broader two-part process – recognition and control.

Acknowledgements

We thank J. Allen, J. Gennari, P. Langley, K. Thompson, J. Bresina, P. Friedland, and G. Iba; each contributed by one or more of: (a) participating in discussions that led to many of the ideas here, (b) commenting on an earlier draft, and (c) providing assistance formatting the figures and tables.

References

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, *2*, 139–172.

Fisher, D. H., & Langley, P. (1990). The structure and formation of natural categories. In G. H. Bower (Ed.), *The psychology of learning and motivation: Advances in research and theory* (Vol. 26). Cambridge, MA: Academic Press.

Fisher, D. H., & Pazzani, M. (Eds.) (in press). *Concept formation: Knowledge and experience in unsupervised learning*. San Mateo, CA: Morgan Kaufmann.

Gennari, J. H. (1990). *An experimental study of concept formation*. Doctoral dissertation, Department of Information & Computer Science, University of California, Irvine.

Gluck, M. A., & Corter, J. E. (1985). Information, uncertainty, and the utility of categories. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society* (pp. 283–287). Irvine, CA: Lawrence Erlbaum.

Iba, W., & Gennari, J. H. (in press). Learning to recognize movements. In D. H. Fisher & M. Pazzani (Eds.), *Concept formation: Knowledge and experience in unsupervised learning*. San Mateo, CA: Morgan Kaufmann.

Iba, W., & Langley, P. (1987). A computational theory of motor learning. *Computational Intelligence*, *3*, 338-350.

Laird, J. E., Hucka, M., Yager, E. S., & Tuck, C. M. (1990). Correcting and extending domain knowledge using outside guidance. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 235–243). Austin, TX: Morgan Kaufmann.

Mason, M. T., Christiansen, A. D., & Mitchell, T. M. (1989). Experiments in robot learning. *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 141–145). Ithica, NY: Morgan Kaufmann.

Moore, A. W. (1990). Acquisition of dynamic control knowledge for a robot manipulator. *Proceedings of the Seventh International Conference on Machine Learning* (pp. 235–243). Austin, TX: Morgan Kaufmann.

Rubin, J. M., & Richards, W. A. (1985). *Boundaries of visual motion* (MIT-AI Memo 835). Cambridge, MA: Massachusetts Institute of Technology.

Smith, E. E., & Medin, D. L. (1981). *Categories and concepts*. Cambridge, MA: Harvard University Press.