A Method for Clustering the Experiences of a Mobile Robot that Accords with Human Judgments

Tim Oates, Matthew D. Schmill and Paul R. Cohen
Computer Science Building
University of Massachusetts, Box 34610
Amherst, MA 01003-4610
{oates,schmill,cohen}@cs.umass.edu

Abstract

If robotic agents are to act autonomously they must have the ability to construct and reason about models of their physical environment. For example, planning to achieve goals requires knowledge of how the robot's actions affect the state of the world over time. The traditional approach of hand-coding this knowledge is often quite difficult, especially for robotic agents with rich sensing abilities that exist in dynamic and uncertain environments. Ideally, robots would acquire knowledge of their environment and then use this knowledge to act. We present an unsupervised learning method that allows a robotic agent to identify and represent qualitatively different outcomes of actions. Experiments with a Pioneer-1 mobile robot demonstrate the utility of the approach with respect to capturing the structure and dynamics of a complex, real-world environment, and show that the models acquired by the robot correlate surprisingly well with human models of the environment.

Introduction

If robotic agents are to act autonomously they must have the ability to construct and reason about models of their physical environment. In all but the simplest, static domains, such models must represent the dynamics of environmental change. For example, because the effects of actions are not instantaneous, planning to achieve goals requires knowledge of how the robots actions affect the state of the world over time. The traditional approach of hand-coding this knowledge is often quite difficult, especially for robotic agents with rich sensing abilities that exist in dynamic and uncertain environments. Ideally, agents would acquire knowledge of their environment and then use this knowledge to act.

This paper presents an unsupervised method for learning models of environmental dynamics based on clustering multivariate time series. An unsupervised learning approach to this problem is desirable because, as noted previously, hand-coding models of dynamic, stochastic environments is a difficult task, and inadequacies of the encoding undermine the agent's autonomy. Experiments with a Pioneer-1 mobile robot demonstrate the utility of the method and show that the models acquired by the robot correlate surprisingly well with human models of the environment.

Individual time series are obtained by recording the output of a subset of an agent's sensors. We call these time series *experiences*. An example of a sensor subset on the Pioneer-1 robot is its array of seven sonars. Each sonar returns the distance to the closest object in the direction that it points. Recording of time series is usually triggered by events, such as the initiation of a particular action. Each time a given event occurs, the time series that was recorded is added to a bucket associated with that event. Once a sufficient number of experiences are recorded, clusters can be formed. Clustering requires a measure of similarity between multivariate time series. One such measure that is particularly appropriate for this problem is Dynamic Time Warping (DTW) (Sankoff & Kruskal 1983). (We discuss DTW in detail in a later section.) Each cluster can then be represented by a prototype, either the cluster centroid or an average of its members. This process is depicted graphically in Figure 1.

Cluster prototypes formed in this manner are useful for a variety of purposes. If the event driving the collection of time series was the initiation of an action, cluster prototypes correspond to qualitatively different outcomes of engaging in that action. As such, they can be used for off-line planning and for on-line prediction by finding the best partial match among the prototypes to current sensor readings.

The remainder of the paper is organized as follows. The next section describes our method for clustering time series in detail, including a discussion of Dynamic Time Warping, the particular clustering algorithm used, and prototype formation. We then present an evaluation of the method as applied to the Pioneer-1 mobile robot. The last two sections review related work, pointing out the connection between Dynamic Time Warping and Hidden Markov Models, and outline future research, respectively.

Clustering Experiences

This section presents our method for unsupervised learning of models of environmental dynamics based on clustering of multivariate time series. To ground the discussion, consider the Pioneer-1 mobile robot. Its sensors include, among others, a bump switch on the end of each gripper paddle that indicates when the gripper hits an object, an infrared break beam between the gripper paddles that indicates when an object enters the gripper, and wheel encoders that measure

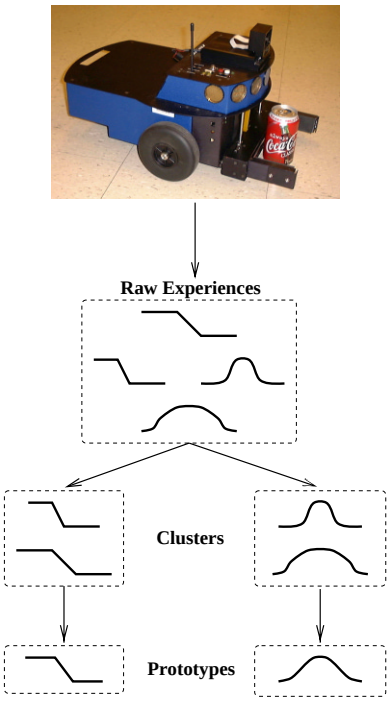

Figure 1: The formation of prototypes of qualitatively difference experiences.

the rate at which the wheels are spinning.

Suppose the robot is moving forward at a fixed velocity. Collectively, the values returned by the sensors mentioned above can discriminate many different situations. For example, if the robot runs into a large immovable object, such as a wall, the bump sensors go high and the wheel velocities abruptly drop to zero. If it bumps into a trash can, which is large but movable, the bump sensors go high and the wheel velocities remain constant. If it comes across an object that can be grasped, the break beam goes high when the object enters the gripper and there is no change in wheel velocity. As observers of the robot's actions, we can label and categorize its experiences. Our goal is to provide mechanisms that will allow the robot to perform that task by itself.

Let E denote an experience, a multivariate time series containing n measurements from a set of sensors such that $E = \{e_t | 1 \quad t \quad n\}$. The e_i are vectors of values containing one element for each sensor. Given a set of m experiences, we want to obtain, in an unsupervised manner, a partition into subsets of experiences such that each subset corresponds to a qualitatively different type of experience. Given such a partition, reasoning with entire sets of experiences is unwieldy, so a simpler representation such as the average experience in a subset is required.

If an appropriate measure of the similarity of two time series is available, clustering followed by prototype extraction is a suitable unsupervised learning method for this problem (see Figure 1). Finding such a measure of similarity is difficult because experiences that are qualitatively the same may be quantitatively different in at least two ways. First, they may be of different lengths, making it difficult or impossible to embed the time series in a metric space and use, for example, Euclidean distance to determine similarity. Second, within a single time series, the rate at which progress is made can vary non-linearly. For example, the robot may move slowly or quickly toward a wall, leading to either a slow or rapid decrease in the distance returned by its forward-pointing sonar. In each case, though, then end result is the same, the robot bumps into the wall. Such differences in rate make similarity measures such as cross-correlation unusable.

The measure of similarity that we use is Dynamic Time Warping (DTW) (Sankoff & Kruskal 1983). It is ideally suited for the time series generated by a robot's sensors. DTW is a generalization of classical algorithms for comparing discrete sequences (e.g. minimum string edit distance (Corman, Leiserson, & Rivest 1990)) to sequences of continuous values. It was used extensively in speech recognition, a domain in which the time series are notoriously complex and noisy, until the advent of Hidden Markov Models which offered a unified probabilistic framework for the entire recognition process (Jelinek 1997).

Given two experiences, E_1 and E_2 (more generally, two continuous multivariate time series), DTW finds the warping of the time dimension in E_1 that minimizes the difference between the two experiences. Consider the two univariate time series shown in Figure 2. Imagine that the time axis of E_1 is an elastic string, and that you can grab that string at any point corresponding to a time at which a value was recorded for the time series. Warping of the time dimension consists of grabbing one of those points and moving it to a new position on the time axis. As the point moves, the elastic string (the time dimension) compresses in the direction of motion and expands in the other direction. Consider the middle column in Figure 2. Moving the point at the third time step from its original location to the seventh time step causes all of the points to its right to compress into the remaining available space, and all of the points to its left to fill the newly created space. Of course, more complicated warpings of the time dimension are possible, as with the third column in Figure 2 in which four points are moved.

Given a warping of the time dimension in E_1, yielding a time series that we will denote E_1', one can measure the similarity of E_1' and E_2 by determining the area between the two curves. That area is shown in gray in the bottom row of Figure 2. Note that the first warping of E_1 in which a single point was moved results in a poor match, one with a large area between the curves. However, the fit given by the second, more complex warping is quite good. In general, there are exponentially many ways to warp the time dimension of E_1. DTW uses dynamic programming to find the warping that minimizes the area between the curves in time that is a low order polynomial of the lengths of E_1 and E_2, i.e. $O(|E_1||E_2|)$.

DTW returns the optimal warping of E_1, the one that minimizes the area between E_1' and E_2, and the area associated with that warping. The area is used as a measure of similarity between the two time series. Note that this measure of similarity handles nonlinearities in the rates at which ex-

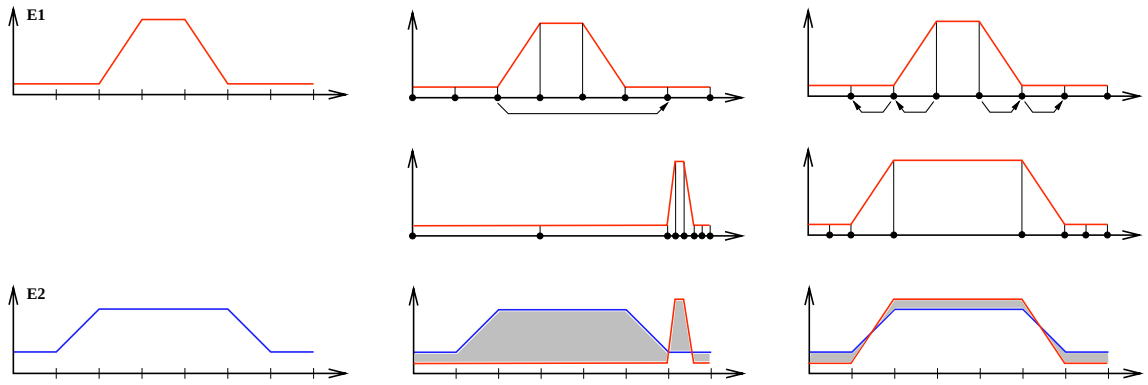

Figure 2: Two time series, E_1 and E_2, (the leftmost column) and two possible warpings of E_1 into E_2 (the middle and rightmost columns).

periences progress and is not affected by differences in the lengths of experiences. In general, the area between E_1' and E_2 may not be the same as the area between E_2' into E_1. We use a symmetrized version of DTW that essentially computes the average of those two areas based on a single warping (Kruskall & Liberman 1983). Although a straightforward implementation of DTW is more expensive than computing Euclidean distance or cross-correlation, there are numerous speedups that both improve the properties of DTW as a distance metric and make its computation nearly linear in the length of the time series with a small constant.

Given m experiences, we can construct a complete pairwise distance matrix by invoking DTW $m(m-1)/2$ times (the factor of 2 is due to the use of symmetrized DTW). We then apply a standard hierarchical, agglomerative clustering algorithm that starts with one cluster for each experience and merges the pair of clusters with the minimum average intercluster distance (Everitt 1993). Without a stopping criterion, merging will continue until there is a single cluster containing all m experiences. To avoid that situation, we do not merge clusters for which the mean intercluster distance is significantly different from the mean intracluster distance as measured by a t-test.

Finally, for each cluster we select a prototype. Two methods commonly used are to choose the cluster member that minimizes the distance to all other members of the cluster, or to simply average the members of the cluster. The advantage of the latter method is that it smooths out noise that may be present in any individual data item. Unfortunately, it is only workable when the cluster elements are embedded in a metric space (e.g. Cartesian space). Although we cannot embed experiences in a metric space, DTW allows us to use a combination of the two methods as follows. First, we select the time series that minimizes distance to all other time series in a given cluster. Then we warp all other patterns into that centroid, resulting in a set of patterns that are all on the same time scale. It is then a simple matter to take the average value at each time point over all of the series and use the result as the cluster prototype.

Evaluation

We are interested in the results of our clustering algorithm for two key reasons. First, for the purposes of planning, we would like clusters to map to action outcomes, so that each cluster prototype can serve as the basis for an operator model. Second, we would like agents to be able to acquire a believable ontology of activity. That is, we would like our agents to be able to differentiate actions as a human would so that their representations of outcome are in accordance with our own. As such, our primary means of evaluating cluster quality is to compare the clusters generated by our automated system against clusters generated manually by the experimenter who designed the experiences they comprise.

Data were collected for 4 sets of experiences: 102 experiences with the robot moving in a straight line while collecting data from the velocity encoders, break beams, and gripper bumper (which we will call the *tactile* sensors), 102 move experiences collecting data from the Pioneer's vision system, including the X and Y location, area, and distance to a single visible object being tracked (which we will call the *visual* sensors), 50 experiences with the robot turning in place collecting tactile data, and 50 turn experiences collecting visual data. In each experience, the robot attempted to move or turn for a duration between 2 and 8 seconds in the laboratory environment. Visible objects and objects that impeded or obstructed the robot's path were present in many of the trials.

The labels given to the hand-built clusters generated are summarized in table 3. In the visual tracking problems, the clusters correspond to visible objects' relations to the agent during activity; the object may move across the visual field while turning or it may loom while being approached. In the tactile problems, clusters correspond to the Pioneer's velocity and the types of contact made with objects in the environment during the activity; heavy objects halt the Pioneer's progress, and are labeled "crash", while light, small objects merely trigger the break beams and are labeled "push".

We evaluate the clusters generated by DTW and agglomerative clustering with a 2×2 contingency table called an

move/tactile	turn/tactile	move/visual	turn/visual
+250 unobstructed	+100 unobstructed	no object	no object
+100 unobstructed	+100 never stops	heavy noise	can't move
-100 unobstructed	+100 bump	approach on right	pass left to right
-250 unobstructed	+100 blocked	approach disappear	pass right to left
+250 temporary bump	+100 temporary bump	discover left reverse	discover right
+100 temporary bump	+100 blocked bump	vanish on right	discover left
+250 push delayed bump	-100 unobstructed	vanish on left	left to right
+250 delayed bump	-100 temporary bump	retreat left	vanish off right
+100 delayed bump	-100 impeded turn	discover right	vanish off left
+250 crash beam1	-100 blocked	approach ahead	
+250 squash		approach, gets big	
+250 push blocked		approach on left	
+250 push		approach, stays small	
+100 push			
+100 push shallow			
+100 blocked			
-100 blocked			

Figure 3: Outcome labels given to the hand built clusters for each of the 4 experience sets.

accordance table. Consider the following table:

	t_e	$:t_e$
t_t	n_1	n_2
$:t_t$	n_3	n_4

We calculate the cells of this table by considering all pairs of experiences e_j and e_k, and their relationships in the target (hand-built) and evaluation (DTW) clusterings. If e_j and e_k reside in the same cluster in the target clustering (denoted by t_t), and e_j and e_k also reside in the same cluster in the evaluation clustering (denoted by t_e), then cell n_1 is incremented. The other cells of the table are incremented when either the target or evaluation clusterings places the experiences in different clusters ($:t_t$ and $:t_e$, respectively).

Cells n_1 and n_4 of this table represent the number of experience pairs in which the clustering algorithms are in accordance. We call $n_1 + n_4$ the number of *agreements* and $n_2 + n_3$ the number of *disagreements*. The *accordance ratios* that we are interested in are $\frac{n_1}{n_1+n_2}$, accordance with respect to t_t, and $\frac{n_4}{n_3+n_4}$, accordance with respect to $:t_t$.

Table 4 shows the breakdown of accordance for the combination of dynamic time warping and agglomerative clustering versus the ideal clustering built by hand. The column labeled "#" indicates the difference between the number of hand-built and automated clusters. In each problem, the automated algorithm clustered more aggressively, resulting in fewer clusters. The columns that follow present the accordance ratios for experiences grouped together, apart, and the total number of agreements and disagreements.

The table shows very high levels of accordance. Ratios ranged from a minimum of 82.2% for experiences clustered together (t_t) in the move/visual set to 100% for experiences clustered together in the turn problems. For the turn problems, the aggressive clustering may account for the high t_t accuracy, causing slightly lower accuracy in the $:t_t$ case.

The disparity in the number of clusters suggests that tuning the parameters of the clustering algorithm to produce more clusters might boost $:t_t$ accuracy while preserving the t_t accuracy. The table for this condition is omitted for the sake of brevity, but our findings were that tuning the clustering algorithm in this way leads to a reduction in accuracy in all but the turn/tactile dataset, whose $:t_t$ accuracy increased 7 points.

The failure of this strategy to increase $:t_t$ accuracy by tuning the clustering algorithm to terminate with more clusters indicates that it is not simply a matter of the number of clusters. Exploration of the t_t disagreements in the move/visual data, the problem with the highest error rate, indicates that 132 out of the 156 errors can be traced to two clusters in the automatically generated set that were distributed differently in the target set. The target clusters were "no object" (no visible object being tracked, some minor noise) and "heavy noise" (noise makes it unclear whether anything was being tracked). The automated set had made the split differently; experiences with any noise were grouped together from those that had none. The remaining 24 errors were covered by a handful of six or seven experiences that were also attracted into clusters by experiences the hand builder did not feel were similar.

The problem is rooted in the tendency of greedy clustering algorithms to suffer from *ordering effects* (Fisher, Xu, & Zard 1992). In clustering schemes based on sorting, the order in which instances are considered biases the clusters that result. In agglomerative clustering algorithms, the clusters that result are biased by the algorithm's greedy choice of always considering merging the lowest distance clusters. Figure 5 illustrates how the ordering effect works on a 2d representation of the move/visual data. Because two of the noisy data are very similar (distance=d_2), they are clustered together early in the clustering process. This early decision creates two cluster centers that individually attract members based on the local greedy policy, where a global view (like our hand-builder's) would cluster them together.

Fortunately, optimization techniques exist that can re-

	#	t_t	$t_t \wedge t_e$	%	$: t_t$	$: t_t \wedge : t_e$	%	Agree	Disagree	%
Move visual	-5	876	720	**82.2**	4275	4125	**96.4**	4845	306	**94.0**
Move tactile	-7	443	378	**85.3**	4708	4468	**95.0**	4846	305	**94.0**
Turn visual	-5	262	262	**100.0**	599	571	**95.3**	833	28	**96.7**
Turn tactile	-6	163	163	**100.0**	698	593	**85.0**	756	105	**87.8**

Figure 4: Accordance statistics for automated clustering against the hand built clustering.

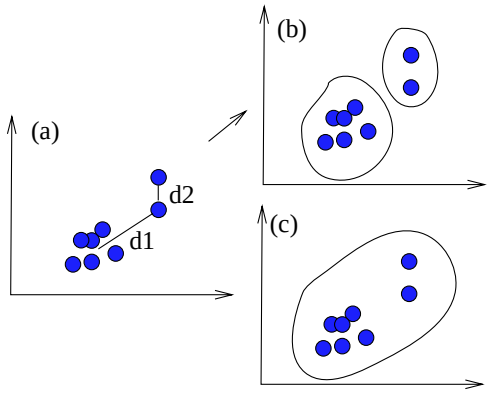

Figure 5: (a) A 2d representation of experiences. (b) The ordering effect of greedily merging based on the shorter distance d2 than the group average distance d1. (c) The most desirable clustering.

fine initial clusterings to better reflect a global view (Fisher 1996). We have implemented a simple optimization technique which iteratively reassigns experiences to neighboring clusters if a cluster is found with a smaller group average distance than to the one the experience is in. After applying this optimization technique to the clusters used to generate table 4, many of the errors in the t_t cases disappeared: accordance climbed to 91.9% or better in all cases except the $: t_t$ case of turn/tactile, which decreased to below 80%, which reflects the disparity between the number of clusters generated by our algorithm and the hand built clustering.

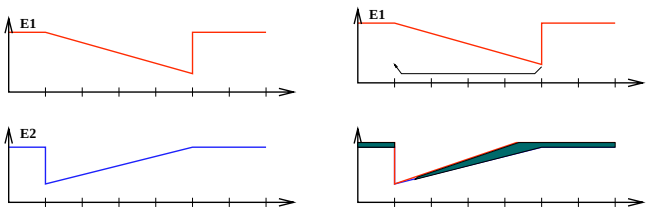

Figure 6: Two time series, E_1 and E_2, and a possible warping of E_1 into E_2 that obscures the salient difference.

The remaining few percent of misses appear to be related to dynamic time warping's ability to manipulate the time dimension. Figure 6 illustrates two time series that correspond to the horizontal location of an object on the Pioneer's visual plane. In experience e_1, the object comes into view from the right, passes across, and disappears off the left side of the visual plane. Experience e_2 represents an object moving in the opposite direction across the visual plane. Clearly, this is a salient distinction for many purposes, including planning, but it is one that DTW is able to obscure by sliding a single point of e_1 backward in time.

Related Work

The use of DTW as a measure of similarity between multivariate time series dates back a number of years to early work in speech recognition (Sakoe & Chiba 1978), although it was ultimately displaced by HMM's (Jelinek 1997). HMM's are actually a powerful generalization of DTW, and recent years have seen renewed interest in DTW for applications where the full power of HMM's may not be required (Berndt & Clifford 1994). That fact notwithstanding, it is unclear how one would apply standard HMM algorithms (such as Baum-Welch and Viterbi) directly to clustering time series. One recent attempt (Smyth 1997) at that problem is much more complex, both computationally and descriptively, than our application of DTW and requires a priori knowledge of the number of clusters (although a method for attempting to determine that number is presented).

Other approaches to measuring similarity between continuous time series have been proposed in the literature (Agrawal *et al.* 1995; Keogh & Pazzani 1998). However, these approaches are limited to univariate time series and are therefore not applicable to our problem, in which one sensor alone is insufficient to discriminate between experiences.

Conclusion

We have presented an approach to clustering the experiences of an autonomous agent acting in a complex, stochastic environment. Using Dynamic Time Warping as a measure of the similarity between time series sensor data, we produced clusters based on the dynamics of experiences, rather than static features. We evaluated the effectiveness of the unsupervised clustering algorithm by measuring the amount of *accordance* between the clusters it generated and cluster sets generated by hand as an answer key. Using only Dynamic Time Warping and agglomerative clustering on 150 trials of real Pioneer data in a variety of experiences, we measured 82-100% accordance between the automated and hand-built clusterings. By applying a simple iterative optimization algorithm to the initial clusterings, accordance measures increased to 91.9% and better.

Still, pathological cases exist where Dynamic Time Warping was able to find a temporal mapping that glossed over significant differences in time series exist. Though these cases cover only a small percentage of the robot's practical

experiences, it is possible to constrain DTW so that these differences will be felt through the distance metric.

Future work will extend the approach described above in three ways. First, we plan on extending the autonomy of our system by utilizing cluster prototypes as bases for planning models, which will allow the Pioneer-1 agent to create basic action sequences to achieve sensorimotor goals. Second, rather than using each experience in its entirety, we will develop methods for identifying subsequences within the experiences that are relevant to the clustering process. Finally, we intend to leverage the relationship between DTW and HMM's to develop a method of clustering time series in which the output is a set of HMM's, one for each cluster.

References

Agrawal, R.; Lin, K.; Sawhney, H. S.; and Shim, K. 1995. Fast similarity search in the presence of noise, scaling and translation in time series databases. In *Proceedings of the 21st International Conference on Very Large Databases*.

Berndt, D. J., and Clifford, J. 1994. Using dynamic time warping to find patterns in time series. In *Working Notes of the Knowledge Discovery in Databases Workshop*, 359–370.

Corman, T. H.; Leiserson, C. E.; and Rivest, R. L. 1990. *Introduction to Algorithms*. MIT Press.

Everitt, B. 1993. *Cluster Analysis*. John Wiley & Sons, Inc.

Fisher, D. H.; Xu, L.; and Zard, N. 1992. Ordering effects in clustering. In *Proceedings of the 9th Annual Conference on Machine Learning*, 163–168.

Fisher, D. 1996. Iterative optimization and simplification of hierarchical clusterings. *Journal of Artificial Intelligence Research* 4:147–179.

Jelinek, F. 1997. *Statistical Methods for Speech Recognition*. MIT Press.

Keogh, E., and Pazzani, M. J. 1998. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *Working Notes of the AAAI-98 workshop on Predicting the Future: AI Approaches to Time-Series Analysis*, 44–51.

Kruskall, J. B., and Liberman, M. 1983. The symmetric time warping problem: From continuous to discrete. In *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley.

Sakoe, H., and Chiba, S. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transaction on Acoustics, Speech and Signal Processing* 26:143–165.

Sankoff, D., and Kruskal, J. B., eds. 1983. *Time Warps, String Edits, and Macromolecules: Theory and Practice of Sequence Comparisons*. Reading, MA: Addison-Wesley Publishing Company.

Smyth, P. 1997. Clustering sequences with hidden markov models. In *Advances in Neural Information Processing 9*.