# The Importance of Being Discrete: Learning Classes of Actions and Outcomes through Interaction

Gary King & Tim Oates

Computer Science Department, LGRC
University of Massachusetts, Box 34610
Amherst, MA 01003-4610
gwking@cs.umass.edu, oates@ai.mit.edu
(413) 577-0669

**Abstract.** A robotic agent experiences a world of continuous multivariate sensations and chooses its actions from continuous action spaces. Unless the agent is able to successfully partition these into functionally similar classes, its ability to interact with the world will be extremely limited. We present a method whereby an unsupervised robotic agent learns to discriminate discrete actions out of its continuous action parameters. These actions are discriminated because they lead to qualitatively distinct outcomes in the robot's sensor space. Once found, these actions can be used by the robot as primitives for further exploration of its world. We present results gathered using a Pioneer 1 mobile robot.

## 1 Introduction

To live sucessfully in a world, robotic agents must be able to derive meaning from continuous state spaces and select actions from continuous ranges of possibility. In order to thrive agents must be able to find discrete classes of states and actions that enable them to achieve their goals. For example, the Pioneer 1 mobile robot has a pair of independent drive wheels and a variety of sensors including seven sonars and a CCD camera. To move, the robot must select a speed for its right and left wheels from an infinite range of possible parameters. While it acts, the values returned from its sonars, camera and other sensors will transition through a subset of an infinite number of states. As far as the robot can tell, every one of its possible wheel speed settings is a different action and every one of its distinct sensor readings is a different state. If the robot had to use these as primitives, it would be unable to understand the world or determine what actions to take in any reasonable amount of time.

Of course, many of these wheel speed settings lead to qualitatively similar outcomes. The robot will go forward, backwards, turn left or right or not move at all. We can examine the robot's behavior and categorize its actions because *we* have already categorized these continuous domains into discrete chunks. However, providing a robot with knowledge of our categories by hand-coding primi-

tive actions and states is tedious, error prone, and must be tuned to each particular model of robot. Lastly, since the robot's sensing and effecting abilities are not equivalent to our own, we may be unable to provide distinctions which are optimally effective for the robot as it attempts to interact with and control its environment.

The problem of learning action models for the purpose of planning is studied in a variety of forms. Much of this work focuses on simulated domains and assumes discrete state and action spaces and deterministic outcomes of actions [5, 15], though some allows for the possibility of probabilistic outcomes [2, 9]. One notable exception is [11], which describes a method for learning action models given continuous state and action spaces for a simulated robot with noisy sensors. Though [16] and others explore the discovery of regimes in time series, their regimes are approximated by normal distributions and their method does not address the case where the time series are (at least partially) caused by an agent.

In stochastic domains with continuous states and discrete actions, reinforcement learning methods can learn reactive control policies [7], and recent work in this area addresses the case in which both the state and action spaces are continuous [13]. Reinforcement learning has also proven to be effective both in simulated domains and with physically embodied robots. Our work differs from these approaches in that the goal is to learn a declarative action model suitable for use by symbolic planning algorithms (and other cognitive tasks such as natural language generation and understanding [10]), not opaque, non-symbolic policies.

Our representation of outcomes as prototypical time series is based on earlier work on clustering time series [8]. Several other recent approaches to identifying qualitatively different regimes in time series data include [1, 3, 6].

Below, we present a method whereby an unsupervised robotic agent can learn qualitively distinct regions of the parameters that control its actions. In our model, the robot begins with a finite number of distinct controllers, each of which is parameterized over zero or more dimensions. Using our method, a robot will be able to learn for itself which regions of the parameter spaces of its controllers lead to what sensory outcomes. These regions can then become the discrete primitive actions which the robot can use to plan. The layout of the paper is as follows: we first describe our robotic agent–the Pioneer 1 mobile robot–and the primitive controllers we created for it; then we describe our method and the experimental results that validate it. Lastly, we discuss the future work involved in turning the prototype actions discovered by our algorithm into planning operators.

## 2 Method

We can view the sensor data collected by the robot as being generated by distinct activities or processes. For example, a process may involve the robot going forward, turning to the right, spinning sharply left or doing nothing at all. Our problem falls into two pieces. The first is to take the set of continuous multivari-

ate time series generated by the robot and discover the distinct activities which created it and which activity generated which time series. In essence we want to discover how many different kinds of things the robot did and which thing goes with which time series. The second problem is to use this information to divide the parameter space(s) of the controller(s) that generated each activity into possibly overlapping regions. These regions build upon the robot's innate controllers and we use them to form the robot's primitive actions.

### 2.1 Framework

Although the method we propose is quite general, we explicate it in the context of our experimental work with the Pioneer mobile robot. We provide a robot with three distinct controllers:

- $\Psi_{RL}(r, l)$–a left-right wheel speed controller. By varying $r$ and $l$, the robot sets its right and left wheel speeds.
- $\Psi_{\emptyset}$–a null controller that sets all the robots effectors to their resting states. Though perhaps perverse, this 'controller' lets the system differentiate action from inaction.
- $\Psi_{OS}$–a controller designed to seek out and move the robot towards open space. It does this by rotating to find the direction with the least amount of sonar clutter, moving in that direction and then orienting towards a randomly chosen heading.

We then let the robot randomly select controllers and parameters and execute them for a brief time–typically between 10 and 20 seconds. The data recorded by the robot during each experience is saved along with the controller type and its parameters, if any. We call the complete set of robot experiences $\mathcal{E}$. Note that qualitatively distinct controller/parameter settings should generate trajectories of qualitatively distinct sensor readings as outcomes. For example, going forward will typically cause the forward facing sonar's distances to go down, the sizes of objects in the visual field to grow and the translational velocity to be positive. Other actions will produce very different readings. The next section describes how we can learn which of these sensor time series are associated with the different kinds of activities in which the robot engages.

### 2.2 Learning Distinctive Outcomes for a Controller

Given $\mathcal{E}$, we search for distinctive outcomes by first uniformly sampling fixed length subsequences of length $L$, called L-sequences, from the data. We then form $k$ clusters from the L-sequences via hierarchical agglomerative clustering using Dynamic Time Warping (DTW) [12] to measure the distance between each sequence. DTW is a generalization of classical algorithms for comparing discrete sequences (e.g. minimum string edit distance [4]) to sequences of continuous values. The $k$ centroids of the clusters found, $C_i$, partition the space of L-sequences, with each centroid standing in for all of the L-sequences that are most similar

to it. In effect, the centroids discretize the continuous sensor space and form an alphabet which can be used to tokenize any other experience.

We next divide $\mathcal{E}$ into two sets for each controller: one set contains experiences that occurred while the controller was running; the other experiences that occurred while some other controller was running. For each centroid, we can determine the probability that $C_i$ occurred when the controller was running, $p(C_i|\Psi)$, and the probability that $C_i$ occurred when the controller was not running, $p(C_i|\overline{\Psi})$. If $p(C_i|\Psi)$ is significantly different from $p(C_i|\overline{\Psi})$ then the centroid is *distinctive* for $\Psi$. Centroids that occur more frequently than by chance (under the null hypothesis that the occurrence does not depend on the controller) are called positively distinctive centroids for $\Psi$ and are denoted by $\Psi(C_i)^+$. Centroids that occur less frequently are negatively distinctive centroids and are denoted by $\Psi(C_i)^-$. Centroids which are neither positively nor negatively distinctive are said to be neutral with respect to the controller. As positively distinctive centroids occur more often in the presence of $\Psi$, we infer that $\Psi$ causes them: that the sensor trajectories similar to $\Psi(C_i)^+$ are the outcomes of running $\Psi$. Typically, the inference that $a$ causes $b$ requires that $a$ and $b$ covary, that $a$ occurs before $b$ and that other potential causes of $b$ are controlled [14]. As our method does not account for the last item, some of the causal inductions will be incorrect and further effort will need to go into resolving them.

### 2.3 From Distinctive Outcomes to Distinctive Actions

For each centroid in $\Psi(C_i)^+$, we examine the experiences in $\mathcal{E}$ and see if the centroid occurs more frequently than by chance. We accomplish this by comparing the number of occurrences of L-sequences similar to the centroid in the experience to that expected given the overall population density of the centroid in $\mathcal{E}$. If $C_i$ occurs frequently in an experience, then we say that the experience is distinctive for the centroid. The set of distinctive experiences for each centroid is $\mathcal{E}_{C_i}$. We will denote the parameters of the distinctive experiences for a centroid as $P_{C_i}$. We can plot $P_{C_i}$ for each controller colored by the centroid. For example, figure 1 shows one particular division of $\Psi_{RL}$'s parameter space. This plot shows left and right wheel speed parameters associated with data collected from the Pioneer-1 while running $\Psi_{RL}$. Each of these robot experiences is labeled with one of six distinctive centroids. For example, the experiences labeled with the small $x$'s all have wheel speeds that are generally below zero. The center portion of the plot is empty because our method did not find any distinctive outcomes for these experiences. Notice that each of the prototypical centroids is associated with a subset of the entire parameter space and that the subsets appear to be well separated.

In general, there are several possible outcomes for the distributions of controller parameters derived from individual centroids $C_i$ and from pairs of centroids $C_j$ and $C_k$. We first list the possibilities and then provide intuitions for their meanings:

1. $P_{C_i}$ has a uniform distribution across the entire parameter space.

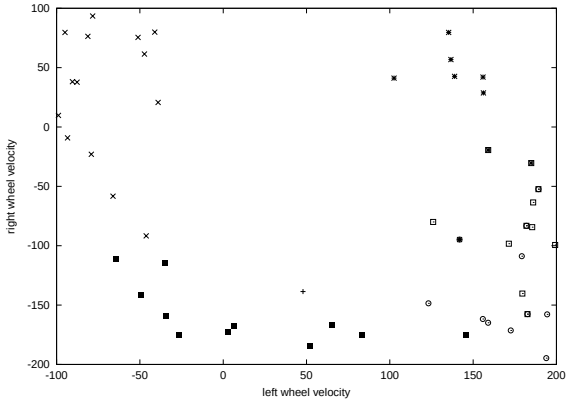


**Fig. 1.** Scatter-plot of Left and Right wheel velocities labeled by their centroid or distinctive outcome.

2. $P_{C_i}$ has a non-uniform distribution–some parameter values lead to $C_i$ more frequently than others. This distribution may be uni-modal, bimodal or more complex.
3. $P_{C_j}$ and $P_{C_k}$ are well separated (note that this can only occur if the individual distributions are non-uniform to begin with).
4. $P_{C_j}$ and $P_{C_k}$ overlap significantly.

We will formalize these notions below but the intuitions should be clear. In the concrete terms of $\Psi_{RL}(r, l)$, item 1 indicates that although the outcome occurs more frequently when $\Psi_{RL}$ is running, it does not depend on the parameters of $\Psi_{RL}$. Item 2 indicates that the occurrence of the centroid depends on $r$ and $l$. If the distribution is uni-modal, then only one range of $r$ and $l$ leads to this outcome; if it is more complex, then two or more ranges lead to it. This corresponds to a different regions of the parameter space having the same outcome.

Items 3 and 4 both require that the outcomes $C_j$ and $C_k$ depend on the choice of $r$ and $l$. If the parameter ranges for the two outcomes overlap significantly, then this corresponds to a single action leading to two (or more) different outcomes. This may be due to the context in which the two action occurs.

### 2.4 Knowing when an action is discrete

Given a distribution $P_{C_i}$, we must ask whether it is significantly different from that expected by random chance. We can divide the parameter space of a controller into uniform cells and create a histogram of the number of occurrences of $P_{C_i}$ in a cell. We can create a similar histogram of the total number of experiences with parameters in a cell regardless of centroid. We can use these histograms to form a discrete probability distribution of the probability that a given range of parameters leads to the distinctive outcome ($C_i$). The null hypothesis is that the parameter values have no effect on the outcomes and that the distribution

obtained from $P_{C_i}$ is uniform. We can test $H_0$ for each $C_i$ by building a sampling distribution of the Kullback-Leibler distances between randomly generated distributions of the number of experiences containing $C_i$. elements and the true uniform distribution. The discrete Kullback-Leibler distance or average variance measures how much one probability distribution differs from another:

$$d(p_1, p_2) = -\sum_x p_1(x) ln \frac{p_1(x)}{p_2(x)}$$

Once we have obtained the distribution of the distance measures, we can use randomization testing to see if the actual distribution derived from $P_{C_i}$ is significant.

If $P_{C_i}$ is significantly different from the non-uniform distribution, then we can use randomization testing again on each of the cells in the distribution. In this case, we build the sampling distribution for the cells of the histogram using the Kullback-Leibler distance of the probability value in each cell as compared to the uniform probability distribution. We then look for cells whose Kullback-Leibler score is significantly different from that expected under $H_0$. These cells are the ones who contribute highly to $P_{C_i}$'s significance. They define the discrete action which leads to outcome $C_i$.

### 2.5 Summary of the Method

In summary, our method is as follows. Given a set of parameterized controllers for a mobile robot and a set of sensors:

1. Randomly select a controller and run it with randomly selected parameters. While it is running, record the data that it generates and save this along with the type of controller and its parameter values.
2. Sample fixed length subsequences uniformly from the data generated and form clusters.
3. For each cluster centroid, $C_i$, and controller, $\Psi$, determine if the probability of the centroid occurring while $\Psi$ is running, $p(C_i|\Psi)$, differs significantly from the probability of the centroid occurring while $\Psi$ is *not* running, $p(C_i|\overline{\Psi})$.
4. Determine the distinctive experiences for each of $\Psi$'s positively distinctive centroids. Use these to create probability distributions for $P_{C_i}$, the parameters of the experiences that lead to outcome $C_i$.
5. Use randomization testing and the discrete Kullback-Leibler distance to find centroids that are dependent on the parameters of $\Psi$ and the regions of the parameter-space that lead to the centroid.

The regions found are ranges of parameter values that typically result in specific outcomes of sensory trajectories. They are candidates for primitive actions of the mobile robot.

# 3 Experiment

## 3.1 Method

We collected 120 experiences using $\Psi_{RL}(r, l)$ (96-experiences), $\Psi_{\emptyset}$ (12-experiences) and $\Psi_{OS}$ (12-experiences). The distribution was weighted towards $\Psi_{RL}$ as this controller was the focus of our experiment. The $r$ and $l$ parameters for $\Psi_{RL}$ were uniformly sampled between -100 and 200 so as to obtain more forward-moving experiences than backward-moving experiences. The robot operated in a convex space containing numerous small objects with which it could interact. Intervention was required once during the data collection when the robot became stuck on power conduit lines attached to one of the walls of the space.

In the analysis that follows we used the following subset of sensors: heading, right-wheel-velocity, left-wheel-velocity, translational-velocity and rotational-velocity. The Pioneer keeps track of its heading and assumed position by dead reckoning. It determines its right and left wheel velocities, translational and rotational velocities via proprioceptive feedback from its wheel encoders. The values of its sensors are recorded every 10-milliseconds.

## 3.2 Results

The algorithm described above found several statistically significant ($p < 0.01$) regions of the parameter space of $\Psi_{RL}(r, l)$ including ones that we would label roughly as "forward", "backwards", "hard-left", "slow-left" and so forth. Figure 2 below demonstrates several probability distributions linking particular setting of left and right wheel speeds and their distinctive outcomes ($C_i$). In this experiment, each distribution was decidedly non-uniform (corresponding to item 2 in the taxonomy above) and demarcated a single contiguous region of the parameter space. The regions are relatively large, however, and several overlap each other. Our assumption is that this is due in part to the small size of our data set and in part to the fact that different actions can result in the same qualitative outcome. Further experiments are underway to clarify this.

Each plot in figure 2 shows the action associated with a particular distinctive outcome. These plots are based on clusters of the data from the original 120-trials. The darker cells of each plot indicate the range of parameters that define the action. The first plot shows the action defined by high values of left and right wheel speeds and with the right wheel speed generally higher than the left wheel speed–with what we would label forward motion and turning to the right. Investigation of the distinctive centroid associated with the plot confirms this interpretation. The second plot shows actions with right wheel speeds below zero and the third shows actions with high left wheel velocities and low right wheel velocities. We might label these activities as "backwards to the left" and "forward left turn" respectively. Of course, each atomic action discovered by our method ranges over a large portion of the controller's parameter space. This is due in part to the limited amount of data collected and in part to the noisy

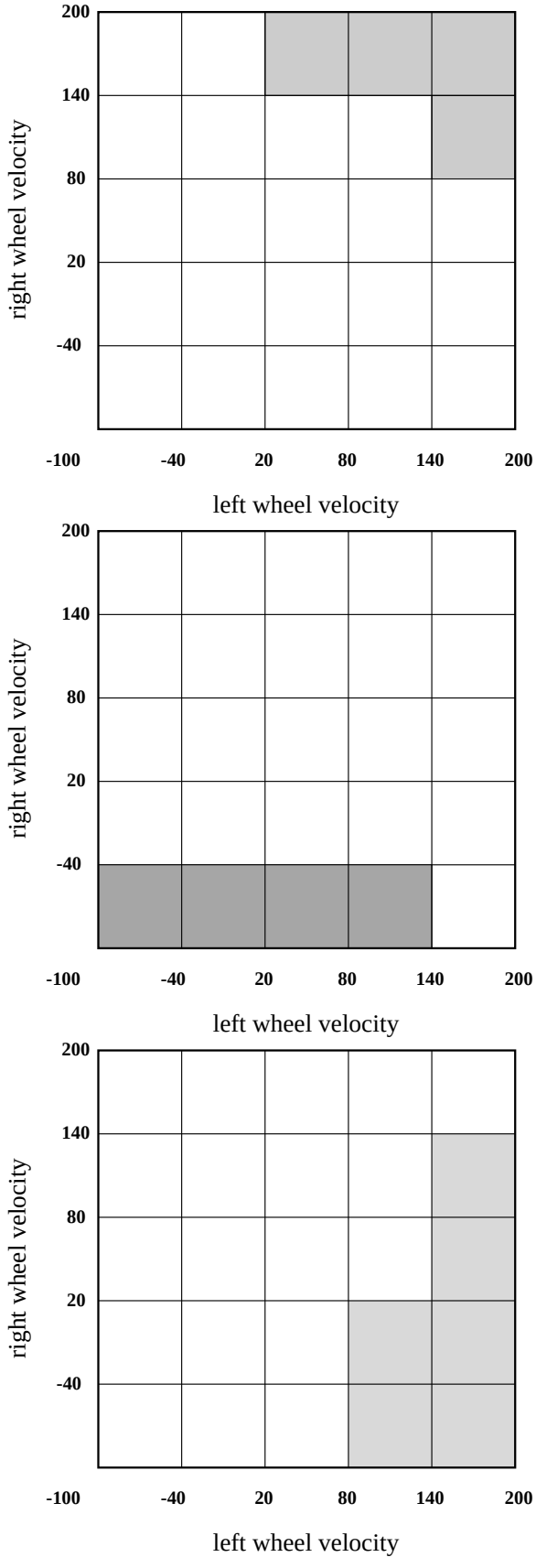


**Fig. 2.** Examples of discrete actions associated with different distinctive outcomes. They corresponds roughly to "going forward"; "going backwards or turning left"; and "turning to the left".

environment in which the robot runs. We expect that additional data would allow the atomic actions to become more precise.

We have shown that our method allows an unsupervised mobile robot to interact with its environment and learn discrete actions over the parameter spaces of its controllers.

## 4 Future Work

Future work will remove a number of limitations of the current method. In particular, rather than representing outcomes of actions as fixed-length prototypes, we will apply the algorithm described in [8] to identify and represent outcomes of variable duration. Also, having identified discrete actions and their outcomes, it becomes possible to go back to the time series data and search for features of the environment that condition the outcome probabilities. In terms of classical planning operators, we will identify preconditions. Another limitation of the current method is that sensor groups are pre-specified. Ideally, the robot would determine which sets of sensors should be grouped together because patterns in those sensors capture outcomes of invoking actions. We plan to explore the utility of a simple generate and test paradigm to this problem, with the test phase involving statistical hypothesis tests of the form previously described. Another extension to this method would be to apply active learning techniques by letting the robot choose its parameters rather than selecting them at random. Finally, the current algorithm runs only in batch. We intend to move towards a completely online implementation whereby the robot will continuously find, test and improve its prototypical actions. Further work also needs to be done to investigate the scalability of our approach and to deal with non-stationary time series. Indeed, as the robot's learning will modify its interactions, we are prima facie in a non-stationary environment. We have hopes that the continuous on-line version will help to shed light on this developmental problem.

## 5 Acknowledgments

We would like to thank our anonymous reviewers for their helpful comments and ideas. This research is supported by DARPA contract DASG60-99-C-0074 and DARPA/AFOSR contract F49620-97-1-0485. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of the DARPA or the U.S. Government.

## References

1. R. Agrawal, K. Lin, H. S. Sawhney, and K. Shim. 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*, 1995.

2. Scott Benson. Inductive learning of reactive action models. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 47–54, 1995.
3. Paul R. Cohen, Marco Ramoni, Paola Sebastiani, and John Warwick. Unsupervised clustering of robot activities: A bayesian approach. To appear in *Proceedings of the Fourth International Conference on Autonomous Agents*, 1999.
4. Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press, 1990.
5. Yolanda Gil. *Acquiring Domain Knowledge for Planning by Experimentation*. PhD thesis, Carnegie Mellon University, 1992.
6. Eamonn Keogh and Michael J. Pazzani. 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*, pages 44–51, 1998.
7. Sridhar Mahadevan and Jonathan Connell. Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55(2–3):189–208, 1992.
8. Tim Oates. Identifying distinctive subsequences in multivariate time series by clustering. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pages 322–326, 1999.
9. Tim Oates and Paul R. Cohen. Searching for planning operators with context-dependent and probabilistic effects. 1996.
10. Tim Oates, Zachary Eyler-Walker, and Paul R. Cohen. Toward natural language interfaces for robotic agents: Grounding linguistic meaning in sensors. In *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 227–228, 2000. Extended abstract.
11. David M. Pierce. *Map Learning with Uninterpreted Sensors and Effector*. PhD thesis, University of Texas, Austin, 1995.
12. David Sankoff and Joseph B. Kruskal, editors. *Time Warps, String Edits, and Macromolecules: Theory and Practice of Sequence Comparisons*. Addison-Wesley Publishing Company, Reading, MA, 1983.
13. J. C. Santamaria, R. S. Sutton, and A. Ram. Experiments with reinforcement learning in problems with continuous state and action spaces. *Adaptive behavior*, 6(2):163–218, 1998.
14. Patrick Suppes. *A Probabilistic Theory of Causality*. North Holland, Amsterdam, 1970.
15. Xuemei Wang. Learning by observation and practice: An incremental approach for planning operator acquisition. In *Proceedings of the Twelfth International Conference on Machine Learning*, 1995.
16. Andreas S. Weigend, Morgan Mangeas, and Ashok N. Srivastava. Nonlinear gated experts for time series: discovering regimes and avoiding overfitting. *Int J Neural Syst*, 6:373–399, 1995.