Discovering Natural Kinds of Robot Sensory Experiences in Unstructured Environments

Daniel H Grollman and **Odest Chadwicke Jenkins** and **Frank Wood**
Brown University Department of Computer Science
Providence, RI 02912
{*dang,cjenkins,fwood*}*@cs.brown.edu*

Abstract

We derive categories directly from robot sensor data to address the symbol grounding problem. Unlike model-based approaches where human intuitive correspondences are sought between sensor readings and features of an environment (corners, doors, etc.), our method learns intrinsic categories (or natural kinds) from the raw data itself. We approximate a manifold underlying sensor data using Isomap nonlinear dimension reduction and use Bayesian clustering (Gaussian mixture models) with model identification techniques to discover kinds. We demonstrate our method through the learning of sensory kinds from trials in various indoor and outdoor environments with different sensor modalities. Learned kinds are used to classify new sensor data (out-of-sample readings). We present results indicating greater consistency in classifying sensor data employing mixture models in low-dimensional embeddings.

1. Introduction

The symbol grounding problem in robotics deals with connecting arbitrary symbols with entities in the robot's world. Names such as 'door', 'hallway, and 'tree' must be associated with sensor readings so that an autonomous robot can reason about them at a higher level. Traditionally, a human programmer is relied upon to provide these connections by identifying areas in the world that correspond to preconceived labels and building *models* of how they would appear to the robot. However, actual sensory information is dictated by the robot's embodiment and may not accord with models of sensor function. Consequently, our understanding of a robot's perception of the world is often biased and heuristic.

A data-driven approach to sensor analysis could discover a more appropriate interpretation of sensor readings. Sensor data collected during robot operation are observations of the underlying sensory process and, if teleoperation is involved, the control policy of the operator. We posit that the intrinsic structure underlying robot sensor data can be uncovered using recent techniques from manifold learning. Once uncovered, sensory structures can provide a solid foundation for autonomous sensory

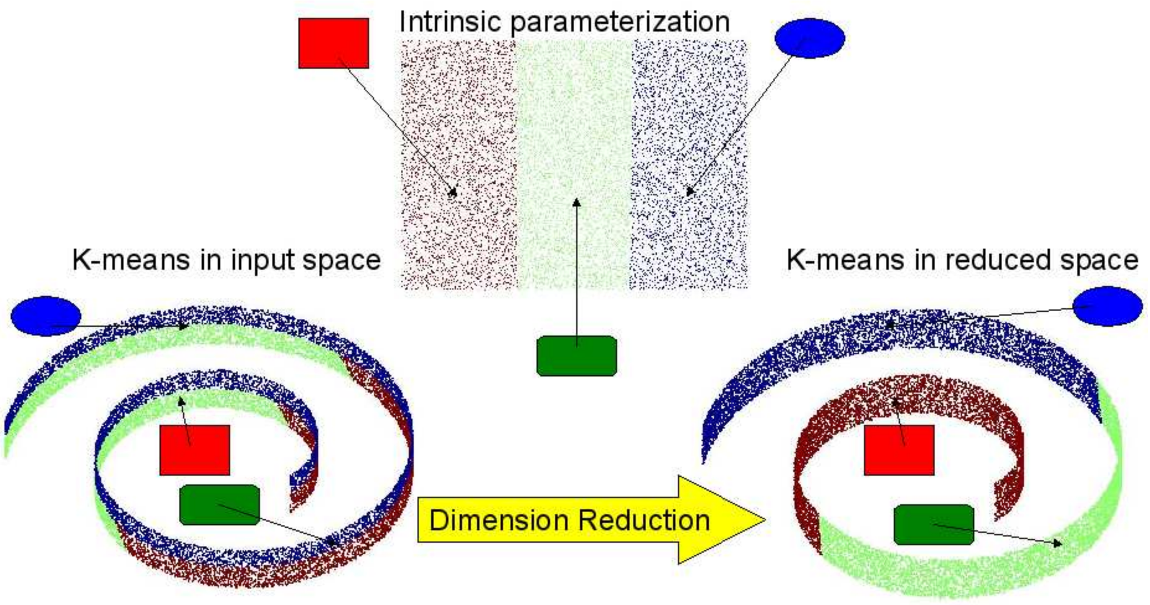


Figure 1: Here a 2D manifold is embedded in 3D space. Clustering directly in the input space leads to clusters that do not accurately reflect intrinsic structure of the data; similar datapoints may classified differently. By applying manifold learning techniques to reduce dimensionality before clustering, this problem can be alleviated.

understanding as a robot's perceptual system is allowed to develop classes of sensor data based on its own, unique, experiences.

We present a data-driven method for classifying robot sensor input via unsupervised dimension reduction and Bayesian clustering. We view the input of the system as a high dimensional space where each dimension corresponds to a reading from one of the robot's sensors. This sensory space is likely to be sparse and described by a lower dimensional subspace. Our approach is to embed sensor data into a lower-dimensional manifold that condenses this space and captures latent structure. By clustering in this embedded space we generate simpler probability densities while grouping together areas that appear similar to the robot. We take each cluster of sensor readings in the reduced-dimensional space as a kind[1] of entity as viewed by the robot. As seen in Figure 1, datapoints that are intrinsically similar may be placed into different clusters if clustering is performed without manifold learning. A different approach would be to do simple dimensionality reduction, by finding the dimensions of highest variance and ignoring others. However, this approach does not capture structure latent in the data.

1. Philosophically, a natural kind is a collection of objects that all share salient features. For instance, the 'Green Kind' includes all green objects. We use the terms 'kind', 'class' and 'category' interchangeably.

Once classes are learned, new sensor readings can be quickly classified with an out-of-sample (OOS) classification procedure. This procedure projects new samples into the embedding space where they are classified with a Gaussian mixture model (GMM). When a location is revisited, this procedure should embed the new readings near the old ones, allowing the space to be classified consistently.

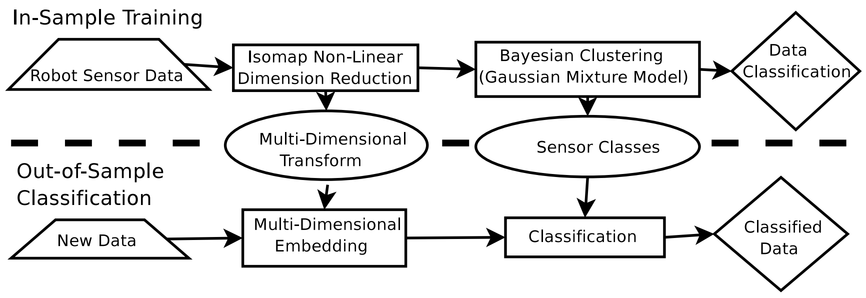
We use consistency as an evaluation metric because ground truth is unknown. The clusters developed by this technique reflect areas that are perceived similarly by the robot, and as previously stated, our models of robot perception are biased and heuristic. Therefore the discovered classes may not reflect any categories we would develop ourselves. It is however important that the found kinds be consistent, by which we mean that similar inputs should belong to the same kind and be classified similarly.

2. Related Work

Topological mapping depends on the ability to discover regions in an explored area (Thrun, 1998). This process is usually done by extracting features from sensor data that indicate the robot's current location. When a human decides which region types exist in the robot's world and which features are important (Tomatis et al., 2003), biases from models of sensor operation are introduced. We attempt to remove these biases by deriving classes directly from sensor data.

Localization techniques also depend on region identification. Landmarking, or the identification of unique places, is commonly used to let a robot know when it has returned to a previously visited location on a map (i.e., revisiting, loop closure). The revisiting problem is key when it comes to map-making because it allows a robot to discover loops in the world (Howard, 2004) or, in the case of multiple exploration robots, it allows one robot to discover when it has entered space explored by another (Stewart et al., 2003). Often, landmarking is accomplished by modifying the environment to disambiguate similar places. We hypothesize that with a data-driven classification technique, it will become clearer which areas of the world look similar to the robot and require disambiguation. Without landmarking, localization depends on estimating the location of the robot using, for example, a Hidden Markov Model (Shatkay, 1998) or the connections between regions already seen (Howard et al., 2001). All of these approaches require a robust way of identifying the kind of space that the robot currently occupies.

A semi-supervised approach to discovering clusters in vision data is introduced in (Grudic and Mulligan, 2005). By allowing each cluster to self-optimize its parameters, they are able to discover clusters that more accurately correspond to the predefined ones, as well as detect outlying points that do not belong to any cluster. However, the original clusters must be decided upon by human operators and exemplar photographs of each cluster are provided to the algorithm. In contrast, our approach


In-Sample Training
Robot Sensor Data
Isomap Non-Linear Dimension Reduction
Bayesian Clustering (Gaussian Mixture Model)
Data Classification
Multi-Dimensional Transform
Sensor Classes
Out-of-Sample Classification
New Data
Multi-Dimensional Embedding
Classification
Classified Data

(a) Methodology

(b) Crunch

Figure 2: Our method in flowchart form. Data from robot sensors are analyzed with Isomap to obtain a low-dimensional embedding. The embedded data is then clustered to develop sensor classes. Out-of-sample data can be quickly projected and classified using the information developed during the in-sample training.

is completely unsupervised and allows for the discovery of space classes and outliers that are potentially non-obvious to humans.

In order to tie sensing and action together, (Klingspor et al., 1996) learn sensory and action concepts directly from the sonar data of a robot, after the data is segmented and categorized by hand. By utilizing sensor information related to actions (such as teleoperation data), we can determine the usual action performed in each space class in an unsupervised way and use these actions as a first-attempt control policy.

Advancements in dimension reduction (DR) and manifold learning have been shown to have beneficial effects on reinforcement learning (Roy and Gordon, 2003; Mahadevan, 2005). We believe our approach can help bring these benefits to autonomous robot understanding.

3. Methodology

Our method, outlined in figure 2(a), views d-dimensional robot sensor data as lying on a manifold in $\mathbb{R}^d$. We model each sensor datum $\vec{x}$ as having been generated by a mixture model on this manifold, where each mixture density corresponds to a natural kind. Here we closely follow the methods and notation of (Bengio et al., 2004).

3.1 Training

For training, let $D = \{\vec{x}_1, \ldots, \vec{x}_N\}$ be the collection of readings from S sensors at N time instances. We compute an affinity matrix M by approximating the geodesic distance between points on the sensor data manifold. As in (Tenenbaum et al., 2000), we define the geodesic distance between points a and b to be:

$$\tilde{D}(a,b) = \min_p \sum_i d(p_i, p_{i+1})$$

where p is a sequence of points of length $l \geq 2$ with $p_1 = a$, $p_l = b$, and $p_i \in D\ \forall i \in \{2, \ldots, l-1\}$ and (p_i, p_{i+1}) are neighbors as determined by a k-nearest neighbors algorithm. We compute $\tilde{D}$ by applying Dijkstra's algorithm (Cormen et al., 1990) to the graph $V = D, E = \{p_i, p_{i+1}\}$ where edge length is the Euclidian distance between neighbors.

M is formed with elements $M_{ij} = \tilde{D}^2(x_i, x_j)$ and then converted to equivalent dot products using the "double-centering" formula to obtain $\tilde{M}$.

$$\tilde{M}_{ij} = -\frac{1}{2}(M_{ij} - \frac{1}{2}S_i - \frac{1}{N}S_j + \frac{1}{N^2}\sum_k S_k)$$

where $S_i = \sum_j M_{ij}$. In practice, this grows as N^2 and is thus currently infeasible to calculate for more than a few thousand points. For larger datasets, only a subset of the data can be fully processed (landmarks).

The k dimensional embedding $\vec{e}_i$ of each sensor output $\vec{x}_i$ on the sensor data manifold is obtained via Multi-Dimensional Scaling (MDS). Here the embedding is approximated by the vector $\vec{e}_i = [\sqrt{\lambda_1}v_{1i}, \sqrt{\lambda_2}v_{2i}, \ldots, \sqrt{\lambda_k}v_{ki}]$ where λ_k is the k^{th} largest eigenvalue of $\tilde{M}$ and v_{ki} is the i^{th} element of the corresponding eigenvector. We reduce the dimensionality of the sensor data by setting $k < d$, thus removing many of the low eigenvalue coordinates of the embedding. k is selected by comparing the variance between distances in the input and reduced spaces for different dimensionalites. We define $E = \vec{e}_1, \ldots, \vec{e}_N$ to be the reduced dimensionality embedding of the training sensor data D in k dimensions, henceforth referred to as the "sensor embedding."

Initially, we assume that the sensor embeddings were generated by exactly J statistically distinct intrinsic classes of sensor readings. We assume that the distribution of each of these classes is Gaussian and fit E with a mixture model with J components.

The probability that $\vec{e}_i$ was output by the robot's sensors while it was in a physical space corresponding to sensor class j, $1 < j < J$ given these assumptions is:

$$P(\vec{e}_i|j) = \frac{1}{(2\pi)^{\frac{k}{2}}\sqrt{det(\Sigma_j)}} exp(-\frac{1}{2}(\vec{e}_i - \mu_j)^T \Sigma_j^{-1}(\vec{e}_i - \mu_j)) \qquad (1)$$

where μ_j and Σ_j are the mean and covariance of the sensor output while in class j.

Assuming that each sensor datum is independent, then the probability of E according to the mixture model is:

$$P(E) = \prod_{i=1}^{N}\sum_{j=1}^{J}\alpha_j P(\vec{e}_i|j)$$

Algorithm 1 Training

```
Input: Data ($D$), NeighborhoodSize ($ns$), Dimensionality ($k$) , ClusterNumber ($J$)
Output: Embedding and Mixture Parameters
1: Create $N$, a neighborhood matrix where $N_{ij} = \text{dist}(i, j)$, the Euclidian distance
   between points $i$ and $j$ in $D$, if $j$ is one of $i$'s $ns$ nearest neighbors, $\infty$ otherwise
2: $\tilde{D} = \text{dijkstra}(N)$ (Geodesic Distance)
3: $\tilde{M} = \text{double center}(\tilde{D}^2)$
4: $[\lambda, v] = \text{eigendecomposition}(\tilde{M})$
5: Keep only the first $k$ elements of $\lambda$ and $v$
6: Create $E$, the embedded coordinates where $E_i = [\sqrt{\lambda_1}v_{1i}, \sqrt{\lambda_2}v_{2i}, \ldots, \sqrt{\lambda_k}v_{ki}]$
7: Get $\mu$, $\Sigma$, the means and covariances of a Gaussian Mixture model with $J$ com-
   ponents fit to $E$
8: return $\lambda, v, \mu, \Sigma$
```

where the $\alpha_j > 0$ are mixing coefficients and $\sum_{j=1}^{J} \alpha_j = 1$.

The EM algorithm (McLachlan and Basford, 1988) is used to maximize $P(E)$ by solving for optimal distribution parameters and membership weights. This maximization is accomplished by the iterative optimization of a log likelihood function:

$$\log(\mathcal{L}(\Theta|E, \mathcal{Y})) = \sum_{i=1}^{N} \log(\sum_{j=1}^{J} \alpha_{y_i} P(\vec{e}_i|\Sigma_j, \mu_j))$$

where $\Theta = \{\mu_1, \ldots, \mu_J, \Sigma_1, \ldots, \Sigma_J\}$ is a set of unknown parameters corresponding to the mean sensor data embeddings and covariance matrices for the J classes and

$$\mathcal{Y} = \{y_i\}_{i=1}^{N}, 1 < y_i < J, y_i \in \mathbb{Z}$$

is an array of unknown variables such that $y_i = j$ if $\vec{e}_i$ came from mixture component j. The training step is show algorithmically in Algorithm 1.

Model selection is a central issue in clustering and corresponds to determining the number of clusters (intrinsic classes) in the data. We employ two existing empirical criteria for model selection, Bayesian Information Criteria (BIC) and cross-validation. The BIC penalizes likelihood as a function of the complexity of the model. If κ is the number of free parameters in the model, then we calculate the BIC as:

$$-2\log(\mathcal{L}(\Theta|E, \mathcal{Y})) - \kappa(\log(N) + 1)$$

Since in practice the BIC often doesn't sufficiently penalize complex models, we additionally use cross-validation on held-out data to check for overfitting: We train our model on half the training data and then compute the unpenalized likelihood of the remainder. When too many classes are posited, i.e. the model may be over-fit, the likelihood of the held-out data may decrease relative to simpler models. These two techniques guide us in selecting J.

Algorithm 2 Out-of-sample Classification

```
Input: Data ($D$), NeighborhoodSize ($ns$), Geodesic Distance ($\tilde{D}$), Embedding ($\lambda, v$)
and Mixture ($\mu, \Sigma$) parameters, new datapoint ($\vec{p}$)
Output: Soft cluster assignments
1: Create $N$, where $N_i$ = the Euclidian distance between $\vec{p}$ and the $i$th point in $D$
   if it is one of $\vec{p}$'s $ns$ nearest neighbors, else $\infty$
2: for all $\vec{x} \in D$, indexed by $i$ do
3:     $\bar{D}_i = min(N_i, min_j(N_j + \tilde{D}_{ji}))$
4: end for
5: Get $\vec{e}$ by embeding the new point into the manifold according to Eqn. 2 where
   $\tilde{D}(\cdot, \vec{p})$ and $\tilde{D}(\vec{p}, \cdot)$ are given by $\bar{D}$
6: for each of the $J$ classes do
7:     Get $S_j$, the probability of $\vec{p}$ coming from class $j$ using Eqn. 1
8: end for
9: return $S$
```

3.2 Online Classification

Online classification of a new point $\vec{p}$ is simple and rapid. We refer the reader to (Bengio et al., 2004) for full details. The embedding is given by:

$$e_k(\vec{p}) = \frac{1}{2\sqrt{\lambda_k}} \sum_i v_{ki}(E_{\vec{x}}[\tilde{D}^2(\vec{x}, \vec{x}_i)] + E_{\vec{x}'}[\tilde{D}^2(\vec{p}, \vec{x}')] - E_{\vec{x},\vec{x}'}[\tilde{D}^2(\vec{x}, \vec{x}')] - \tilde{D}^2(\vec{x}_i, \vec{p})) \quad (2)$$

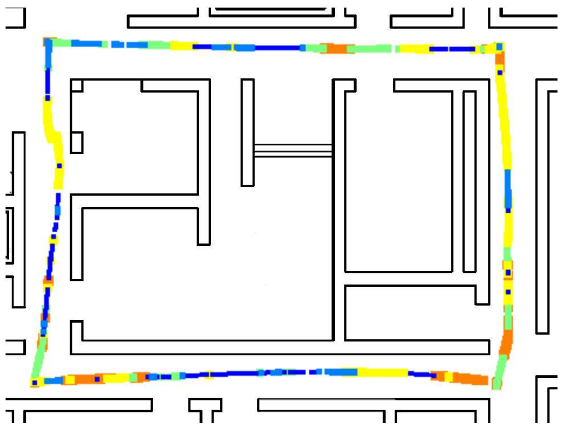
where E is an average over the training data set. Using the GMM from the training stage, we determine the probability of this newly embedded point belonging to each cluster. This classification process is show algorithmically in Algorithm 2.

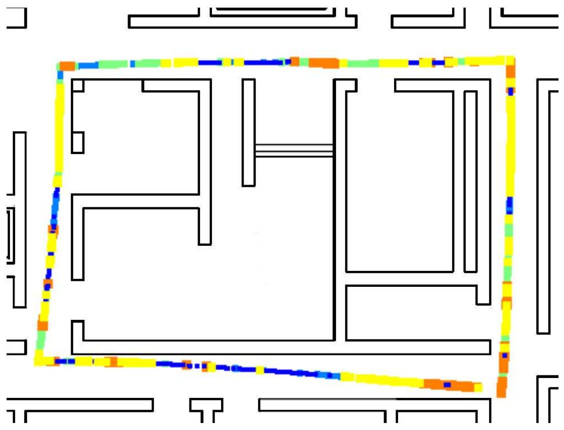
4. Experiments

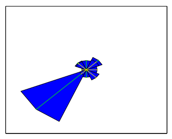
To test our algorithms, we collected robotic sensory data as a robot was teleoperated through an environment several times. Data from one trip was analyzed using algorithm 1 to learn embedding and clustering parameters. Then, data from other trips were run through algorithm 2. Categorizations from multiple trips in the same environment were then examined for consistency. We also compared the results of our Isomap based algorithm to one based on Principle Component Analysis (PCA). For that approach we used the first k principle components of the data as the embedding space.

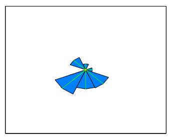
4.1 Data Collection

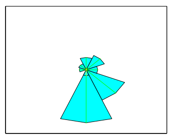
Data was collected from two robots in two different environments. Indoors, in an office environment, we used **Crunch**, the small, cylindrical, inverted pendulum robot

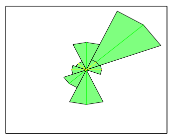

(a) Trip1, 4th Floor


(b) Trip2, 4th Floor


(c) Class1


(d) Class2


(e) Class3


(f) Class4

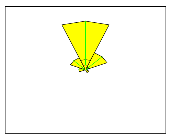

(g) Class5

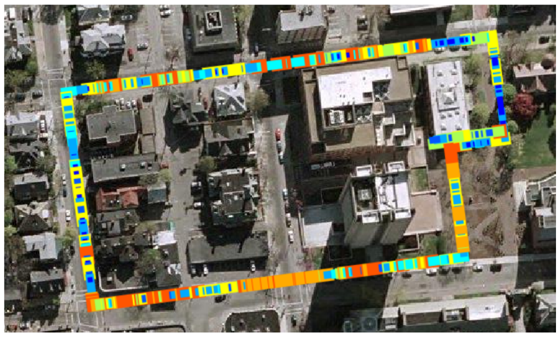
Figure 3: Results from Crunch on the fourth floor of the Brown University CIT. 3(a): Sensor data from trip 1 has been clustered into 5 classes. A unique width and color value for each class is overlaid on registered odometry to show the classification of regions of space. 3(b): The learned classes were used to classify data from a second trip. 3(c)-3(g): The mean-centered expected sensor readings for each class under the standard ray model.

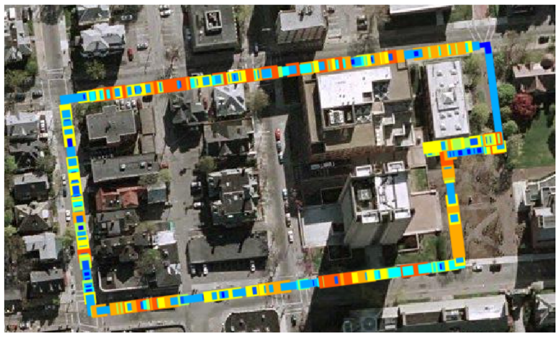
pictured in figure 2(b). It has eight sonar and eight IR sensors arranged in dual rings around its body as well as wheel encoders that record wheel rotation. During operation, these sensors are sampled and transmitted back to a base laptop where they are logged at around 10Hz.

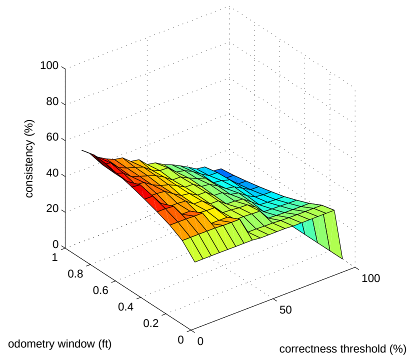
Outdoors, we used **Chew**, the large, six-wheeled all-terrain robot pictured in 4(d). It is equipped with a SwissRanger time-of-flight distance camera that uses structured light to determine a 160 x 124 distance array. Running at ~5hz, these 19840 distances are timestamped and logged onboard.

4.2 Learning Sensory Kinds

For the training phase, one set of data from each robot was used to discover embedding and clustering parameters. For Crunch, after computing the geodesic distance and MDS embedding of the training data, we retained 8 of the resulting dimensions for future processing. Based on the BIC and holdout calculations, we judged that there were 5 classes in the data.The resulting mixture model was used to assign each datapoint to a class. For display purposes, we manually registered the odometry


(a) Trip 1


(b) Trip 2



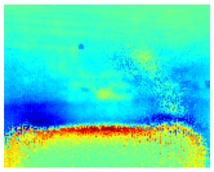
(c) Consistency

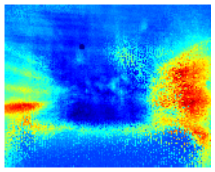
(d) Chew

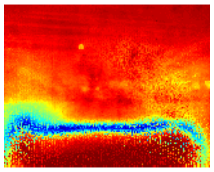
Figure 4: The first of two trips with Chew 4(d) was analyzed to discover sensor classes. Both trips were then categorized, colorcoded and overlaid on a map (courtesy of Google Maps): 4(a) and 4(b). Consistency across both trips as a function of odometric window and cutoff percentage is show in 4(c)

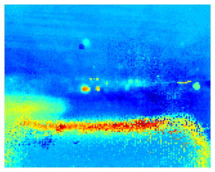
with the underlying floor plan and overlaid these classes on the path that the robot followed. This assignment is illustrated in figure 3(a). Figures 3(c)-3(g) show expected readings from each of the 5 classes discovered by our method. These images were generated using a "ray model" of Crunch's IR and sonar sensors and the values were computed from a weighted average of the mean-centered datapoints. Under this model, many of these shapes are hard to interpret as corresponding to a hallway, doorway, corner, etc, but these are the sensor readings that are most distinguishable to the robot.

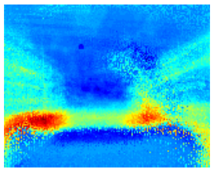
Similarly, we processed the first trip that Chew took. After reducing from 19840 to 15 dimensions, we estimated 8 clusters in the data. The odometry was registered, coded, and overlaid on a map of the environment in Figure 4(a). Figures 5(a)-5(h) show the cannonical distance measurements for each of the classes. These have been mean-centered for ease of viewing.

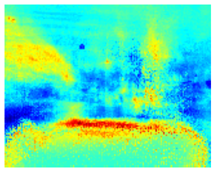

(a) Class1

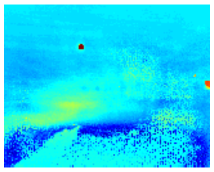

(b) Class2


(c) Class3


(d) Class4


(e) Class5


(f) Class6


(g) Class7

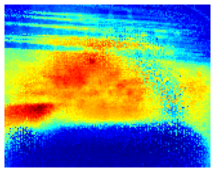

(h) Class8

Figure 5: Mean-centered cannonical depth views of each class discovered by Chew.

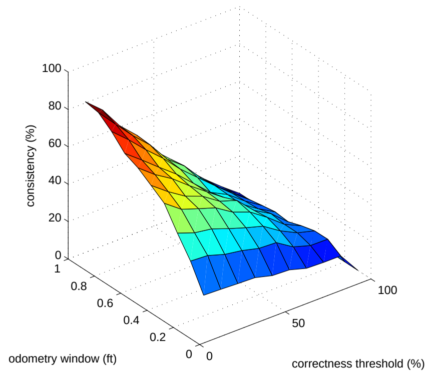
4.3 Consistent Sensory Classification

Our first experiment was designed to test the consistency of our classification when a location is revisited. We used the parameters learned from the training stage to classify data from a second trip in the same environment. As the robot followed the same general path as it did in the first trip, we expected the sensory readings along the path to be classified similarly across trips. The results from the out-of-sample classification of Crunch's second trip are shown in figure 3(b), and Chew's in figure 4(b).

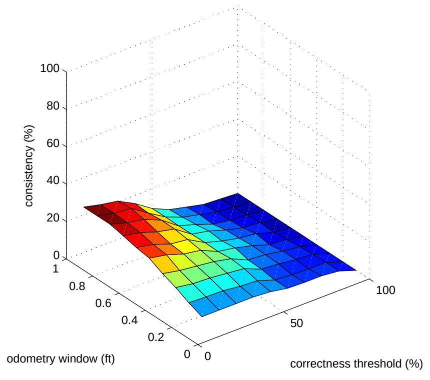
We used the registered odometry to compare classifications of the same physical space across trips. Given a position (x, y) in the first trip that has been classified as generating sensor readings of kind k, we compare it to all points from the second trip within a certain radius, r. If more than a given percentage z of these points have also been classified as kind k, we declare a match. Figure 6 shows a plot of how measured consistency varies with these two parameters for both our approach and the PCA based one applied to the Crunch data set. As you can see, manifold learning greatly improves the consistency of classification. For example, if we require 50% of points within a 1 foot radius to be classified the same, our approach achieves 40% consistency, while PCA performs at less than 10%. The consistency between random assignments over these paths is $\sim$0.5%.

4.4 Consistency in New Spaces

We wanted to see if classes learned in one space were applicable to new, although similar, spaces. To this end Crunch moved from the fourth to the fifth floor of our building and we collected and classified data using our out-of-sample technique. Results are shown in Figure 7. If the learned classes were non-applicable to the space,



(a) Isomap Based



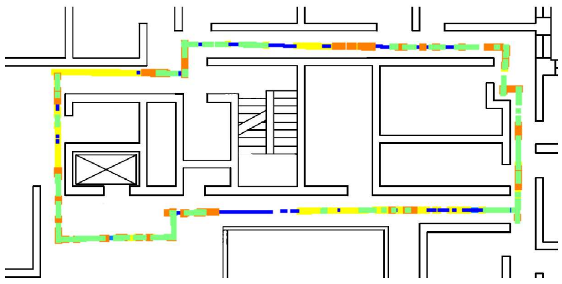
(b) PCA based

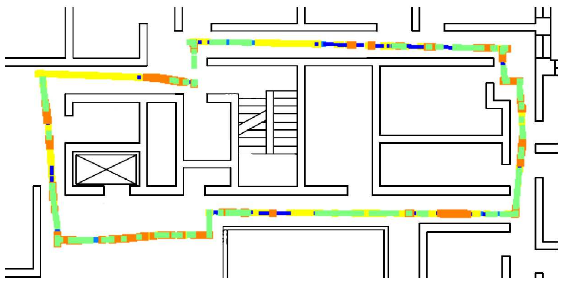
Figure 6: The consistency metric is highly sensitive to constant selection and registration errors. Here we show the measured consistency of our Isomap based technique as the constants (r, z) are varied, 6(a), and compare it to a PCA-based version 6(b).

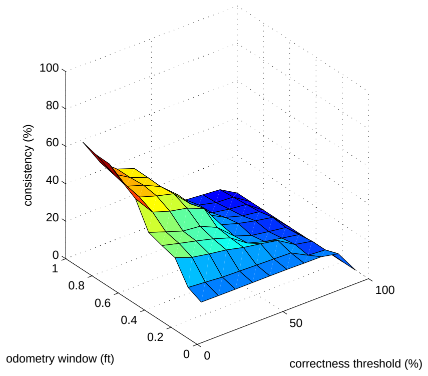
that is, if areas that looked similar to the robot were not assigned to the same cluster, we would expect to see successive data points assigned to different classes. Instead, there are several large contiguous sections of points that are all assigned to the same class. Furthermore, by repeating the consistency test from above, and classifying data from a second trip on the fifth floor using the same classes, we see that these classifications are usable in this area, even though they were not learned here.

5. Discussion and Conclusion

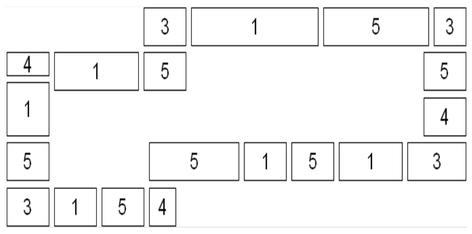
We attempt to remove human bias from the analysis of robotic sensor data by identifying latent structure in the sensor readings themselves. Currently, we empirically determine the neighborhood function and size, the number of embedding coordinates to retain, and the number of intrinsic sensor classes. In theory, each of these can be determined automatically, and perhaps even adaptively, from the data. In particular, model selection is very difficult. There are techniques to alleviate this issue, such as infinite mixture models that will be incorporated in future work. The main contribution of the work presented here is in demonstrating that intrinsic sensor classes may form a better foundation for applications that require classifying sensor data. In addition, we currently treat each sensor reading as independent. Better performance may result from modeling spatial and temporal correlations as in ST-Isomap (Jenkins and Matarić, 2004). Parametric Embedding (Iwata et al., 2004) is a an approach that preserves associations between data objects and mixture components during embedding, which could be useful in this context.


(a) Trip1, 5th Floor


(b) Trip2, 5th Floor



(c) Consistency



(d) Topological Map

Figure 7: Using the sensor classes discovered on the fourth floor, Crunch took two trips around the fifth floor of our building and classified each datapoint. As before, the consistency metric, 7(c) shows that the two trips are classified similarly. Thus the learned classes are applicable in other (although similar) locations. 7(d): a topological map derived from our method, see text.

Because our technique operates in a space defined by robot sensors, the results are sometimes difficult to reconcile with human intuition. In particular, when the "canonical" sensor reading for a Crunch class is examined, it does not correspond to any class that we, as humans, would have developed for the robot. In fact, even the *number* of classes in the space differs. However, as Crunch is a small wheeled robot equipped with sonar and IR and we are tall humans with eyes, it makes sense that our world views, and our divisions of that world into categories, would be different. Our intuition is further bolstered by noting that armed with the kinds discovered by our system, a human crawling on his hands and knees through the area explored by Crunch can see how they match up. In addition, while it is tempting to interpret the cannonical views from Chew as images, it must be remembered that they are actually distance measurements. Even this interpretation is incorrect, as reflections, refractions, and other sources of photons can influence the sensor and cause it to return something other than distance.

Alas, there is no “ground truth” we may use to evaluate our model. By design we cannot determine the “correct” classification of each point in robot sensor space. At most, we can use an ad-hoc metric to test classifications for consistency. The metric described herein is highly sensitive to registration errors and constant selection. It served only to help us intuit that our classification scheme is consistent and reapplicable.

5.1 Mapping and Control

One use of our system would be the creation of topological maps of the robot’s environment. Such “robot-centric” maps (Grudic and Mulligan, 2005) require that the robot accurately recognize when it is in certain types of space. By combining our classification with odometric data, rough topological maps can be derived. Figure 7(d) shows a topological map derived from 7(a) by dividing the space into regions based on classification. Further processing with loop-closure algorithms and landmark identification techniques (Howard, 2004) can refine these maps into useful tools for autonomous robot navigation.

In addition, control algorithms can be derived from the motor data associated with each class. Firstly, we can use the average movement of the robot in each space class as a first-pass control policy for what the robot should do if it finds itself in that class. Furthermore, we can include the teleoperation data in the training process, so areas that are clustered together not only look similar, but are areas where the robot should behave similarly as well (at least according to the teleoperator). We plan to use this ability to perform robotic learning by demonstration (Nicolescu and Matarić, 2003). After being led through a task by a human teleoperator, a robot can segment the task and associate actions with each segment in an unsupervised manner. As the task is repeated, more data become available to fine tune the robot’s actions. Standard reinforcement learning techniques can also be applied to allow a human trainer better control.

6. Conclusion

This paper presents an extensible method for data-driven discovery of intrinsic classes in robot sensor data. We demonstrate that classes discovered with manifold-learning techniques are more consistently recognizable than those found using PCA. We also show that these classes are reapplicable to new data using out-of-sample techniques. We believe that this technique can provide a basis for future work in autonomous robot operation.

Acknowledgements

This work was supported in part by the NSF. The authors would also like to thank E. Chris Kern and Brock Roberts for their support and assistance.

References

Y. Bengio, J. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press/McGraw-Hill, 1990.

G. Grudic and J. Mulligan. Topological mapping with multiple visual manifolds. In *Robotics: Science and Systems (R:SS 2005)*, 2005.

A. Howard. Multi-robot mapping using manifold representations. In *IEEE International Conference on Robotics and Automation*, pages 4198–4203, New Orleans, Louisiana, Apr 2004.

A. Howard, M. J. Matarić, and G. S. Sukhatme. Relaxation on a mesh: a formalism for generalized localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1055–1060, Wailea, Hawaii, Oct 2001.

T. Iwata, K. Saito, N. Ueda, S. Stromsten, T. L. Griffiths, and J. B. Tenenbaum. Parametric embedding for class visualization. In *Neural Information Processing Systems*, 2004.

O. C. Jenkins and M. J. Matarić. A spatio-temporal extension to isomap nonlinear dimension reduction. In *The International Conference on Machine Learning (ICML 2004)*, pages 441–448, Banff, Alberta, Canada, Jul 2004.

V. Klingspor, K. J. Morik, and A. D. Rieger. Learning concepts from sensor data of a mobile robot. *Machine Learning*, 23(2-3):305–332, 1996.

S. Mahadevan. Proto-value functions: Developmental reinforcement learning. In *International Conference on Machine Learning*, 2005.

G. J. McLachlan and K. E. Basford. *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, 1988.

M. N. Nicolescu and M. J. Matarić. Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In *Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2003.

N. Roy and G. Gordon. Exponential family pca for belief compression in pomdps. *Advances in Neural Information Processing Systems*, 15, 2003.

H. Shatkay. *Learning Models for Robot Navigation*. PhD thesis, Brown University, 1998.

B. Stewart, J. Ko, D. Fox, and K. Konolige. The revisiting problem in mobile robot map building: A hierarchical bayesian approach. In *The Conference on Uncertainty in Artificial Intelligence (UAI 2003)*, 2003.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290, 2000.

S. Thrun. Learning maps for indoor mobile robot navigation. *Artificial Intelligence*, 99:21–71, 1998.

N. Tomatis, I. Nourbakhsh, and R. Siegwart. Hybrid simultaneous localization and map building: a natural integration of topological and metric. *Robotics and Autonomous Systems*, 44:3–14, 2003.