Discovering Natural Kinds of Robot Sensory Experiences in Unstructured Environments

Daniel H. Grollman,
Odest Chadwicke Jenkins,
and Frank Wood
Department of Computer Science
Brown University
Providence, Rhode Island 02912-1910
e-mail: dang@cs.brown.edu,
cjenkins@cs.brown.edu, fwood@cs.brown.edu

We address the symbol grounding problem for robot perception through a data-driven approach to deriving categories from robot sensor data. 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 apply Bayesian clustering (Gaussian mixture models) with model identification techniques to discover categories (or 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 then used to classify new sensor data (out-of-sample readings). We present results indicating greater consistency in classifying sensor data employing mixture models in nonlinear low-dimensional embeddings. © 2007 Wiley Periodicals, Inc.

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 infor-

mation 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 these observations can be uncovered using recent techniques from manifold learning. Once uncovered, this structure can provide a solid foundation for autonomous sensory under-

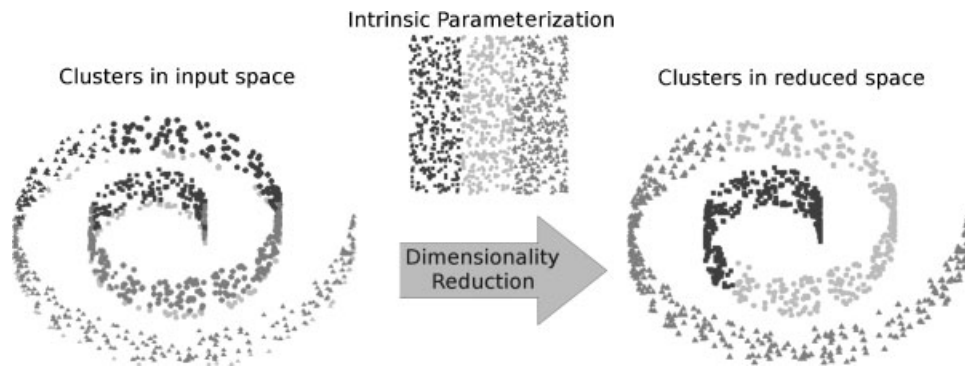WILEY InterScience®
DISCOVER SOMETHING GREAT

Figure 1. An example of clustering three-dimensional "Swiss roll" data generated by a contorted two-dimensional manifold. Clusters (colors) learned directly in the input space do not necessarily reflect intrinsic structure of the data (here represented by shape). That is, proximal datapoints along the underlying manifold may be classified differently. This problem can be alleviated by applying manifold dimensionality reduction prior to clustering.

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

We present such a data-driven method for classifying robot sensor input via unsupervised dimensionality 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 nonlinearly 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 naive approach might be to do simple linear dimensionality reduction by finding the dimensions of highest variance and ignoring others. This approach, however, would not capture nonlinear structure latent in the data.

Once classes are learned, we show that new sensor readings can be quickly classified with an out-of-sample classification procedure. This procedure

projects new samples into the embedding space where they can be classified into a kind. 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 and often subjective. 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 important, however, 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 on a symbol set, or which region types exist in the robot's world and which features are important (Tomatis, Nourbakhsh & Siegwart, 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

---

[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 term "kind," "class," and "category," interchangeably.

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, Ko, Fox & Konolige, 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, Matarić & Sukhatme, 2001). All of these approaches require a robust way of identifying the kind of space that the robot currently occupies.

There has been much work in the area of symbol grounding, particularly as it applies to region identification. Usually in this scenario, an example of a region is provided by a human and the algorithms learn to classify new stimuli. Because the regions (and therefore the symbols) are selected by humans, their biases can have adverse effects on the efficacy of the system.

A semisupervised approach to discovering these regions in vision data is introduced in Grudic & 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, exemplar photographs of each cluster are required by the algorithm. In contrast, our approach is completely unsupervised and allows for the discovery of space classes and outliers that are potentially nonobvious to humans.

One of the obstacles that has to be surmounted when learning region types is that a region's type has to be identifiable from many different viewpoints if the classification system is to be robust. In addition, every place within a region should be classified the same. When the sensor modality is vision, this means that any image of a space has to be recognized as coming from that space, even if the image was previously unseen. In Kosecká & Li (2004), features were extracted from multiple hand-segmented and labeled camera images of a space to come up with a representation of images of the space itself. This allows for

new images to be correctly classified as being of that space. Additionally, Weng & Chen (2000) use linear subspace methods with a partition tree on robot vision data. Because robot sensor data is potentially nonlinear, we consider a nonlinear alternative to subspace embedding.

In a more general case, Tapus, Tomatis & Siegwart (2004) learn a ring of features (a fingerprint) around the robot to identify a place. The features correspond to aspects of the environment, and function as grounded symbols. The fingerprints themselves involve data from both vision and laser sensors and by modeling the occlusion of features in their identification algorithm, they enable the robot to identify a location from multiple positions inside the location.

In order to tie sensing and action together, Klingspor, Morik & Rieger (1996) learn sensory and action concepts directly from the sonar data of a robot, after the data are 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.

In the field of dimension reduction (DR), principal components analysis (PCA) (Bishop, 1995) is the most widely utilized and accessible method for uncovering subspace embeddings. PCA, however is only suitable for uncovering linear subspaces. Recently, Isomap (Tenenbaum, de Silva & Langford, 2000) has emerged as a very good nonlinear DR algorithm. It has been successfully applied to 4096-dimensional pixel data to recover the three actual image dimensions embedded within. For spatio-temporal data, Jenkins & Matarić (2004) has extended Isomap to leverage temporal structure along with spatial nonlinearities. In order to apply a previously learned embedding to new points, Bengio et al. (2004) present a framework for extending discovered embeddings to out-of-sample data points. Recently, DR and manifold learning have been shown to have beneficial effects on reinforcement learning (Roy & Gordon, 2003; Mahadevan, 2005). We believe our approach can help apply these benefits towards autonomous robot understanding.

3. METHODOLOGY

Our method, outlined in Figure 2, views *d*-dimensional robot sensor data as lying on a mani-
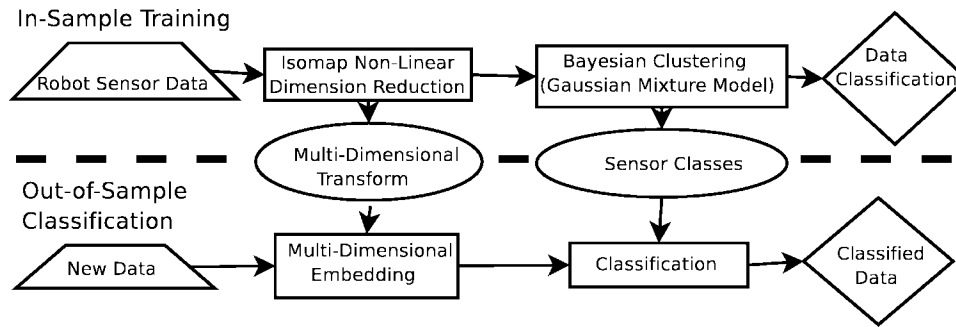
Figure 2. Our method in flowchart form. Data from robot sensors are analyzed with Isomap to obtain a low-dimensional embedding. The embedded data is then clustered with a Gaussian mixture model (GMM) to develop sensor classes. Out-of-sample data can be quickly projected and classified using models learned during the in-sample training.

fold 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 $D=\{\vec{x}_1, \dots, \vec{x}_N\}$ be the collection of readings from S sensors at N time instances. We discover a nonlinear manifold supporting this data using Isomap (Tenenbaum et al., 2000). We briefly review the procedure: First, we compute an affinity matrix M by approximating the geodesic distance between points on the sensor data manifold. Geodesic distance between points a and b is approximated by

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

where p is a sequence of points of length $l \geq 2$ with $p_1 = a$, $p_l = b$, and $p_i \in D \, \forall \, i \in \{2, \dots, l-1\}$ 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, Leiserson & Rivest, 1990) to the graph $V=D, E=\{p_i, p_{i+1}\}$ where edge length is the Euclidean distance between neighbors. It is possible to use a different distance metric, perhaps chosen based on prior knowledge about the sensor types that have generated the data. For generality and applicability, we use Euclidean distance here.

M is formed with elements $M_{ij} = \tilde{D}^2(x_i, x_j)$ and

then centered and converted to equivalent dot products using the "double-centering" formula to obtain $\tilde{M}$,

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

where $S_i = \Sigma_j M_{ij}$. Double-centering ensures that the embedding will be centered around the origin. In practice, $\tilde{M}$ 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 (landmarks) can be fully processed.

The k dimensional embedding $\vec{e}_i$ of each sensor output $\vec{x}_i$ on the sensor data manifold is obtained via multidimensional scaling (MDS). The embedding is approximated by the vector $\vec{e}_i = [\sqrt{\lambda_1}v_{1i}, \sqrt{\lambda_2}v_{2i}, \dots, \sqrt{\lambda_k}v_{ki}]$ where λ_k is the kth largest eigenvalue of $\tilde{M}$ and v_{ki} is the ith 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 error between distances in the input and reduced spaces for different dimensionalities. In particular, we look for an "elbow," a point after which increasing dimensionality does not lead to a significant decrease in residual variance. Practically, we take k to be a few dimensions higher than the elbow to avoid loss of signal. We then define $E=\vec{e}_1, \dots, \vec{e}_N$ to be the reduced dimensionality embedding of the training sensor data D in k dimensions, henceforth referred to as the "sensor embedding."

This concludes our review of Isomap. It is relevant to note that $\tilde{M}$ does not need to be formed from the Geodesic distances. In particular, if all pairs Euclidean distance is used instead, the MDS step would return a result equivalent to standard PCA (Williams, 2002).

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 $j, 1 \leq j \leq J$ given these assumptions is

$$P(\vec{e}_i|j) = \frac{1}{(2\pi)^{k/2}\sqrt{\det(\Sigma_j)}}$$
$$\times \exp\left[-\frac{1}{2}(\vec{e}_i - \mu_j)^T \Sigma_j^{-1}(\vec{e}_i - \mu_j)\right], \quad (1)$$

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

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

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

The Expectation Maximization (EM) algorithm (McLachlan & 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

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

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

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

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.

Algorithm 1 Training

Input:	data (D), neighborhood size (ns), dimensionality (k) number of clusters (J)
Output:	embedding and mixture parameters
1:	Create N, a neighborhood matrix where $N_{ij} = \text{dist}(i,j)$, the Euclidean distance between points i and j in D, if j is one of i's ns nearest neighbors, ∞ 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 λ 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 μ, Σ, the means and covariances of a Gaussian mixture model with J components fit to E
8:	return λ, v, μ, Σ

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 does not 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 overfit, the likelihood of the held-out data may decrease relative to simpler models. These two techniques guide us in manually selecting J.

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 of $\vec{p}$ is given by

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

$$- E_{\vec{x},\vec{x}'}[\tilde{D}^2(\vec{x},\vec{x}')] - \tilde{D}^2(\vec{x}_i,\vec{p})\}, \qquad (2)$$

where E is an average over the training data set. Assuming that the data are from one of the classes previously discovered, we use the GMM from the training stage and determine the probability of this newly embedded point belonging to each cluster. This classification process is shown algorithmically in Algorithm 2.

Algorithm 2 Out-of-sample classification

Input: data (D), neighborhood size (ns), Geodesic distance ($\tilde{D}$), embedding (λ, v) and mixture (μ, Σ) parameters, new datapoint ($\vec{p}$)

Output: soft cluster assignments

1:      Create N, where N_i=the Euclidean distance between $\vec{p}$ and the ith point in D if it is one of $\vec{p}$s ns nearest neighbors, else ∞

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 embedding the new point into the manifold according to Eq. (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 Eq. (1)

8:      **end for**

9:      return S

---

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 were 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 compared the results of our Isomap based algorithm to one based on PCA.

4.1. Data Collection

Data were collected from two robots in two different environments, demonstrating our approach's applicability to multiple platforms, environments, and sensor modalities. Indoors, in an office environment, we used **Crunch**, the small, cylindrical, inverted pendulum robot pictured in Figure 3(a). It has eight sonar and eight infrared (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 10 Hz.

For outdoor experiments, we used **Chew**, the large, six-wheeled all-terrain robot built by Nextek Mobility, pictured in Figure 3(b). Chew has been equipped with a SwissRanger time-of-flight distance camera that uses structured light to determine a 160×124 distance array. Running at $\sim$5 Hz, these 19 840 dimensional data are time stamped and logged onboard. Chew was driven around various locations on the campus of Brown University, which included open grassy areas, streets with car and pedestrian traffic, and collections of buildings for retail shopping.

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 16D training data, we examined the residual variances and retained eight of the resulting dimensions for future processing. Based on the BIC and holdout calculations, we judged that there were five 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 with the underlying floor plan and overlaid these classes on the path that the robot followed. This assignment is illustrated in Figure 4(a). Figures 4(c)–4(g) show expected readings from each of the five classes discovered by our method. These images were gener-
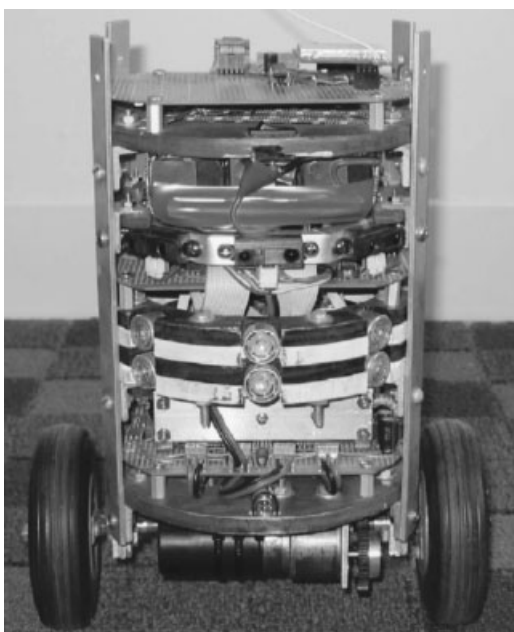
(a) Crunch
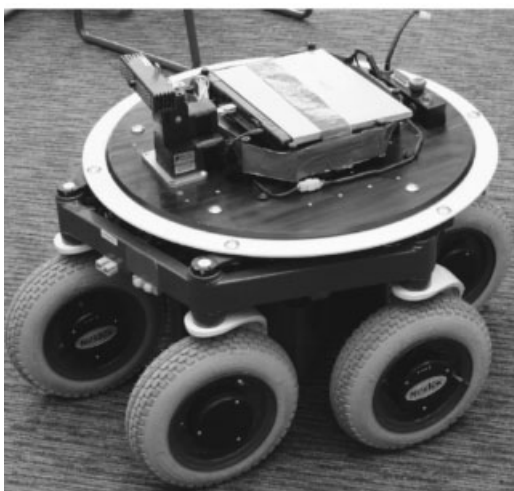
(b) Chew

Figure 3. The robots used in our experiments. (a) Crunch is a small inverted pendulum robot equipped with sonar and infrared sensors and (b) Chew is a large all-terrain vehicle with a time-of-flight distance camera.

ated 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 19 840 to 15 dimensions, we estimated eight clusters in the data. The odometry was registered, coded, and overlaid on a map of the environment in Figure 5(a). Figures 5(c)–5(j) show the mean-centered canonical distance measurements for each of the classes.

We compare the results from the Isomap-based technique described above with those of the PCA-based one, where Geodesic distance has been replaced with all pairs Euclidean distances. We also attempted to perform clustering in the native dimensionality of the robots. Unfortunately, the Chew data are so high dimensional that clustering is computationally intractable on our equipment. Further, while the Crunch data are processable, the results are degenerate in that almost all of the points are assigned to one cluster. Table I shows the dimensionalities and number of discovered clusters for the different techniques used herein.

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 4(b), and Chew's in Figure 5(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. We compare the consistency of classification performed by our manifold based approach and the PCA variant in Figure 6. As you can see, manifold learning greatly improves the consistency of classification for both Crunch and Chew. For example, if Crunch requires 50% of points within a 1 ft radius to be classified the same, our approach achieves 40% consistency, while PCA performs at less than 10%. Faulty, cheap, and idiosyncratic sensors and noisy environments combine to
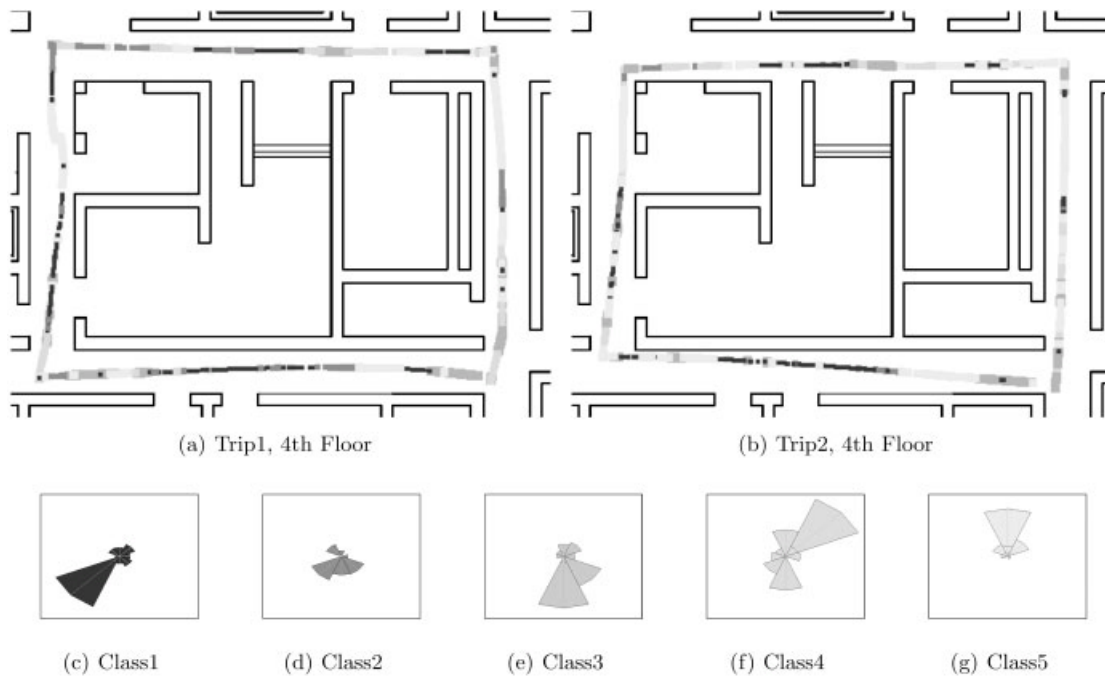
(a) Trip1, 4th Floor   (b) Trip2, 4th Floor

(c) Class1   (d) Class2   (e) Class3   (f) Class4   (g) Class5

Figure 4. Results from Crunch on the fourth floor of the Brown University CIT Building. (a) Sensor data from trip 1 has been clustered into five classes. A unique width and color value for each class is overlaid on registered odometry to show the classification of regions of space. (b) Data from a second trip was classified into the learned classes. (c)–(g) The mean-centered expected sensor readings for each class under the standard ray model.

bring these numbers down. For reference, the consistency between random assignments over these paths is ~0.5%.

In general, Chew has a lower consistency rating. This difference is to be expected as its environment is noisier and far more complicated. In addition to more kinds of objects, Chew has to deal with many more animate entities than Crunch does, such as cars and pedestrians. We hope to address these issues in future work by improving our manifold discovery techniques and incorporating temporal information.

4.4. Consistency in New Spaces

To evaluate the applicability of our approach to new, but similar, spaces, we ran Crunch on a different floor of our CIT building. Data from two trials in this new space were collected and separately classified using our out-of-sample technique. Results are shown in Figure 7.

If the learned classes were nonapplicable to the space, 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
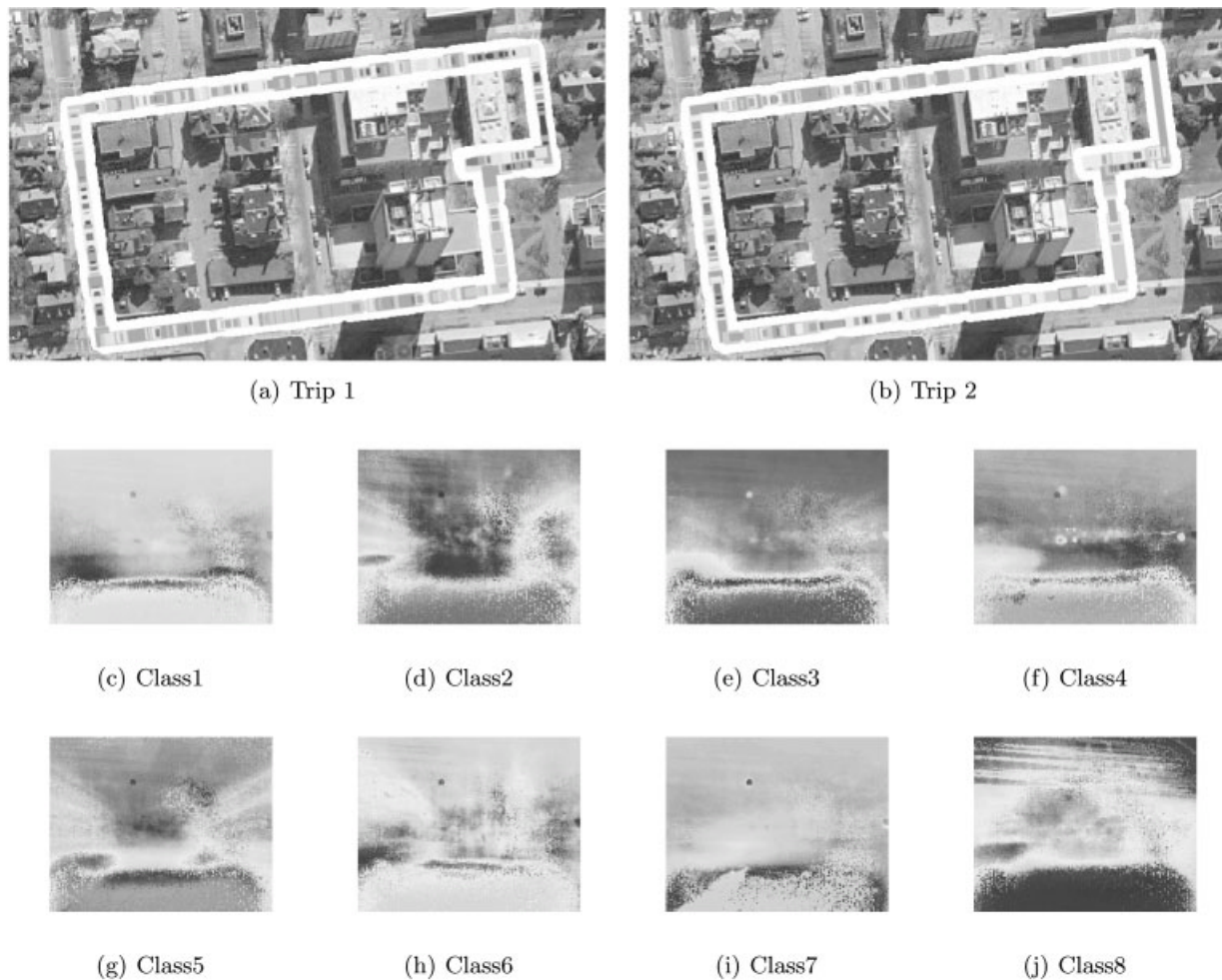
(a) Trip 1

(b) Trip 2

(c) Class1

(d) Class2

(e) Class3

(f) Class4

(g) Class5

(h) Class6

(i) Class7

(j) Class8

Figure 5. Sensory data from the first of two trips with Chew were analyzed to discover sensor classes. (a) and (b) Both trips were then categorized, colorcoded, and overlaid on a map (courtesy of Google Maps). (c)–(j) Mean-centered canonical depth views of each class discovered by Chew.

to alleviate this issue, such as infinite mixture models (Blei, Ng & Jordan, 2003) which allow for new classes to be developed during operation that can 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 & Matarić, 2004). Parametric embedding (Iwata et al., 2005) is an approach that preserves

associations between data objects and mixture components during embedding, which could be useful in this context.

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

Table I. Results of clustering in each of the spaces. Shown are the number of dimensions and the number of discovered clusters for the first data set from Crunch and Chew. Clustering in the raw input space of Chew was computationally infeasible, and is thus not shown here. Clustering in Crunch's raw input space produced a degenerate GMM, and thus the results are also not shown.

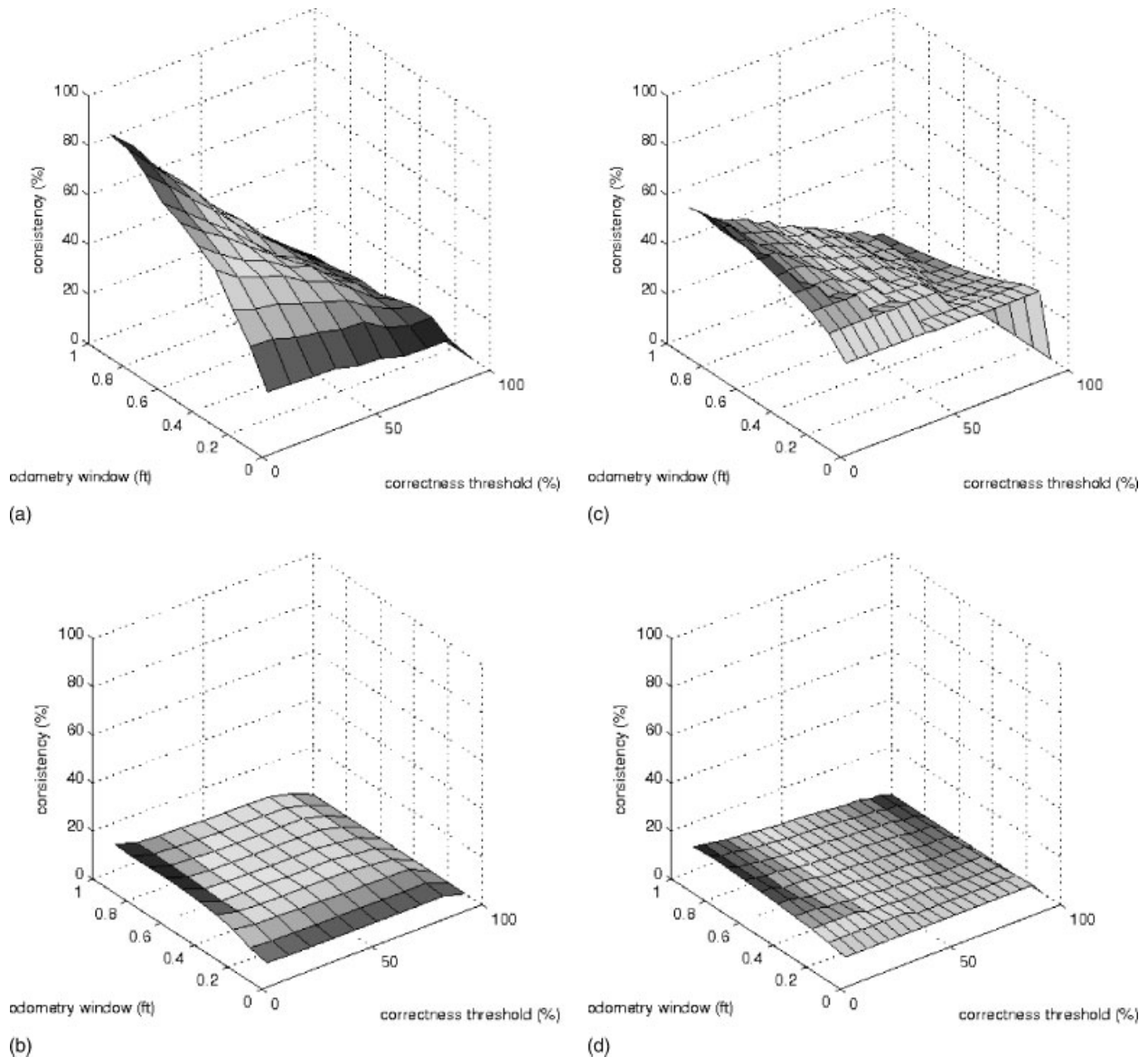|	Input dimensions	Isomap		PCA	
		Dimensions	Clusters	Dimensions	Clusters
Crunch	16	8	5	10	10
Chew	19840	15	8	15	11

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 [(a) and (c)], and compare it to a PCA-based version [(b) and (d)].

(a) Trip1, 5th Floor

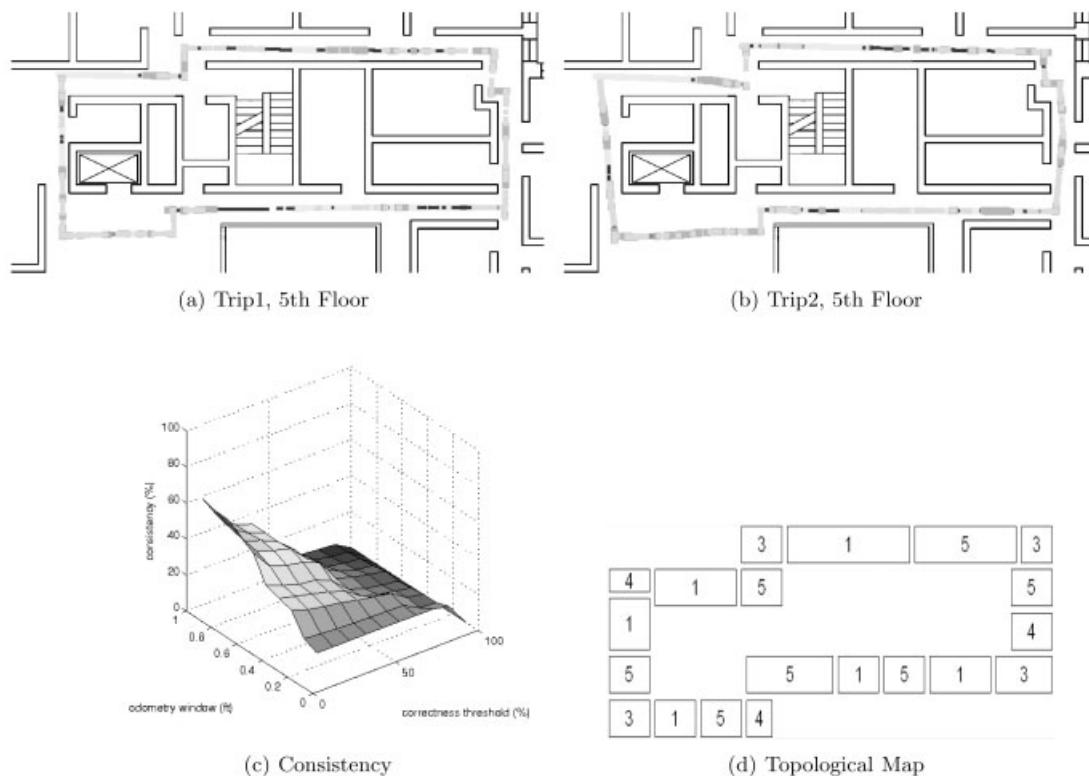(b) Trip2, 5th Floor

(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, (c) shows that the two trips are classified similarly. Thus the learned classes are applicable in other (although similar) locations. (d) A topological map derived from our method.

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 canonical 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 & 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 Figure 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. First, 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 & 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. By comparing human and robot segmentations of a space, disagreements and ambiguities in the environment and control policy can be discovered and dealt with. As an example, consider a control policy that requires a robot to turn left when faced with a dog, and right when encountering a wolf. Manifold-based analysis may show that the robot cannot distinguish between dogs and wolves, and thus the control policy or the environment should be changed. Once tasks are learned in this manner, they can be updated as the robot continues operation. As the task is repeated, more data become available and the clusters and actions may be fine tuned. 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.

REFERENCES

Bengio, Y., Paiement, J., Vincent, P., Delalleau, O., Roux, N.L., & Ouimet, M. (2004). Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. Paper presented at the Conference on Advances in Neural Information Processing Systems 16, Vancouver, BC (pp. 177–184).

Bishop, C.M. (1995). Neural networks for pattern recognition. Oxford: Oxford University Press.

Blei, D., Ng, A., & Jordan, M. (2003). Latent Dirichlet allocation. Journal of Machine Learning Research, 3, 993–1022.

Cormen, T.H., Leiserson, C.E., & Rivest, R.L. (1990). Introduction to algorithms. Cambridge, MA: MIT Press/ McGraw-Hill.

Grudic, G., & Mulligan, J. (2005). Topological mapping with multiple visual manifolds. In S. Thrun, G. Sukhatme, S. Schaal & O. Brock (Eds.), Robotics: Science and systems I (pp. 185–192). Cambridge, MA: MIT Press.

Howard, A. (2004). Multi-robot mapping using manifold representations. Paper presented at the IEEE International Conference on Robotics and Automation, New Orleans, LA (pp. 4198–4203).

Howard, A., Matarić, M.J., & Sukhatme, G.S. (2001). Relaxation on a mesh: A formalism for generalized localization. Paper presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems, Wailea, HI (pp. 1055–1060).

Iwata, T., Saito, K., Ueda, N., Stromsten, S., Griffiths, T.L., & Tenenbaum, J.B. (2005). Parametric embedding for class visualization. Paper presented at the Conference on Advances in Neural Information Processing Systems 17, Vancouver, BC (pp. 617–624).

Jenkins, O.C., & Matarić, M.J. (2004). A spatio-temporal extension to isomap nonlinear dimension reduction. Paper presented at the International Conference on Machine Learning 21, Banff, AB, Canada (pp. 441–448).

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

Kosecká, J., & Li, F. (2004). Vision based topological Markov localization. Paper presented at the International Conference on Robotics and Automation, New Orleans, LA (pp. 1481–1486).

Mahadevan, S. (2005). Proto-value functions: Developmental reinforcement learning. Paper presented at the International Conference on Machine Learning, Bonn, Germany (pp. 553–560).

McLachlan, G.J., & Basford, K.E. (1988). Mixture models: Inference and applications to clustering. New York: Marcel Dekker.

Nicolescu, M.N., & Matarić, M.J. (2003). Natural methods for robot task learning: Instructive demonstrations, generalization and practice. Paper presented at the Joint Conference on Autonomous Agents and Multi-Agent Systems, Melbourne, Australia (pp. 241–248).

Roy, N., & Gordon, G. (2003). Exponential family pca for belief compression in pomdps. Paper presented at the Conference on Advances in Neural Information Processing Systems 15, Vancouver, BC (pp. 701–716).

Shatkay, H. (1998). Learning models for robot navigation. Unpublished Ph.D. thesis, Brown University, Providence, RI.

Stewart, B., Ko, J., Fox, D., & Konolige, K. (2003). The revisiting problem in mobile robot map building: A hierarchical Bayesian approach. Paper presented at the 19th Conference on Uncertainty in Artificial Intelligence, San Francisco, CA (pp. 551–555).

Tapus, A., Tomatis, N., & Siegwart, R. (2004). Topological global localization and mapping with fingerprints and uncertainty. Paper presented at the International Symposium on Experimental Robotics, Singapore.

Tenenbaum, J.B., de Silva, V., & Langford, J.C. (2000). A global geometric framework for nonlinear dimensionality reduction. Science, 290(22), 2319–2323.

Thrun, S. (1998). Learning maps for indoor mobile robot navigation. Artificial Intelligence, 99(1), 21–71.

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

Weng, J., & Chen, S. (2000). Visual learning with navigation as an example. IEEE Intelligent Systems, 15(5), 63–71.

Williams, C.K.I. (2002). On a connection between kernel pca and metric multidimensional scaling. Machine Learning, 46(1–3), 11–19.