

# Topological Mapping from Image Sequences

Jane Mulligan and Greg Grudic\*  
Department of Computer Science  
University of Colorado at Boulder  
Boulder, Colorado, 80309

## Abstract

*An autonomous agent should be able to traverse a new environment and construct a topological representation of what it has seen. We present two new semi-supervised learning techniques which allow us to segment extended sensor (image) sequences into a topological map by clustering on low-dimensional manifolds in sensor space. The general approach is based on outlier detection in manifold space, closely related to spectral clustering. The first technique fixes the  $\sigma$  parameter of the affinity matrix, the second allows the each cluster to optimize for a different  $\sigma$ . In both cases manifold clusters can be associated with the user's conceptual map, by labelling one image per cluster. We demonstrate these techniques for indoor and outdoor sequences.*

## 1. Introduction

Acquiring and exploiting environment maps is a key function for autonomous agents. Turning internal and external sensor readings into a map suitable for navigation, has become a topic of intense interest [13]. Mapping techniques can be loosely divided into topological and geometric approaches. Topological approaches can be thought of as robot-centric, or representations in sensor space rather than in some world coordinate frame [10]. Essentially certain distinguished places and the transitions between them are identified.

Ideally an agent would pass through the environment recording sequential sensor data, and use this to automatically generate the places and transitions which provide a topological description of the space. In this paper we will describe our approach to clustering regions of the sensor sequence (in our case images) using proximity on low dimensional manifolds in sensor space. Our goal is to develop techniques to navigate on the manifold of clustered sensor

sequences. We will not address mapping of images or landmarks to metric pose [12].

Sensor traces such as video sequences for any useful traversal of the environment may involve hundreds or thousands of samples. Supervised learning approaches entail the expense of labelling such large quantities of training data, or subsampling and hence discarding some of the available information. We present a Semi-supervised Learning approach, which requires only limited interactive labelling of images in the sequence and thus can exploit all the available information.

We propose the following two scenarios. The robot is moved (either via teleoperation or using some autonomous search strategy) throughout the environment it is intended to work in. The sensor data that it observes is recorded and projected into manifold space. In the first scenario the data is then clustered in manifold space with a fixed affinity matrix parameter  $\sigma$ . In the second scenario, we propose automatically constructing a low dimensional manifold structure for each of a set of image sequences, allowing a different affinity matrix parameter  $\sigma$  for each cluster. The algorithm uses a framework based on spectral clustering techniques [11, 9, 15]. We present an optimization procedure to automatically compute the clustering parameters for each sequence.

If we want the map to correspond to human concepts of rooms etc, an operator is asked to label a single sensor example in each cluster. This labelled data is then combined with the unlabelled data and every sensor timesample is labelled. These labels are then checked by the operator, and if any inconsistencies are found, more of the data is labelled and semi-supervised learning is once more applied to infer labels for the remaining examples. This process continues until the labels are correct. Each cluster manifold in sensor space corresponds to a uniquely labelled room or part of a room.

Leonardis et al. [6] proposed multiple low dimensional eigenspaces to represent image sets. They grow eigenspaces simultaneously from a set of seed images, then choose the optimal model using a Minimum Description Length For-

---

\*This work has been supported in part by NSF CNS-0430593

mulation. They demonstrate image clusters for a navigation sequence, but do not offer a method for navigating among these eigenspaces.

Image-based approaches to Topological Mapping have been described before [4, 14, 5]. Often these systems have used omnidirectional cameras, which have the advantage of capturing a large field of view and thus reducing ambiguity. However they can also be sensitive to limited dynamic range of the camera in scenes containing light sources. Maeda et al. [7] propose precomputing active vision strategies for regions of the manifold which are close in eigenspace. By acquiring additional images they can disambiguate where the current pose falls. Similarly Kröse and Bunschoten [4] acquire additional images by rotating in place, in order distinguish positions with similar appearance. Kelly [3] composes extended image sequences into globally consistent mosaics and then tracks motion over the mosaic to estimate the current pose. Matsumoto et al. [8] navigate in a sequence of images, assuming that the robot makes transitions in order from one memorized image to the next. Ulrich and Nourbakhsh [14] use Nearest Neighbour learning on colour histograms for labelled images.

In Section 2 we develop the method for semi-supervised clustering on the manifold. Section 3 describes its application to robot navigation. We present results for indoor and outdoor image sequences in Section 4 and sum up in Section 5.

## 2. Semi-Supervised Learning

Let  $X$  be a set  $\{x_1, \dots, x_n\} \subset \mathbb{R}^m$  of images (data points), and let  $L$  be a set  $\{1, \dots, c\}$  of labels. Let the first  $l < n$  points in  $X$  be labelled, the rest unlabelled. We wish to predict the labels of the points  $\{x_{l+1}, \dots, x_n\}$ . To this end, consider a nonnegative  $n \times c$  cost matrix  $F$ . We assign a point  $x_i$  to class  $y_i$  when  $y_i = \arg \max_{j \leq c} F_{ij}$ , to obtain a classification on  $X$ . The original algorithm of Zhou *et al* [16] makes use of  $X$  and knowledge of the labels of  $x_1, \dots, x_l$ , with the aim of finding an  $F$  that gives us a good prediction of the labels of the unlabelled points in  $X$ .

The algorithm works as follows:

- Construct an affinity matrix  $W$  such that  $W_{ij} = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$  if  $i \neq j$  and  $W_{ii} = 0$ .
- Let  $D$  be the diagonal matrix such that  $D_{ii}$  is the sum of the  $i$ -th row of  $W$ : then  $S = D^{-1/2} W D^{-1/2}$  normalizes the rows of  $W$ .
- Finally, let  $Y$  be the  $n \times c$  matrix such that  $Y_{ij} = 1$  if  $x_i$  has label  $j$ , and  $Y_{ij} = 0$  otherwise.

We can now define the cost function:

$$Q(F) = \frac{1}{2} \left( \sum_{i,j} W_{ij} \left\| \frac{F_i}{\sqrt{D_{ii}}} - \frac{F_j}{\sqrt{D_{jj}}} \right\|^2 + \mu \sum_{i=1}^n \|F_i - Y_i\|^2 \right) \quad (1)$$

where  $F_i$  is the  $i$ -th row of  $F$  and  $\mu > 0$ . Then for variable candidate matrices  $F$ , our classifying function is:

$$F^* = \arg \min_F Q(F). \quad (2)$$

The first term term in  $Q(F)$  maintains local consistency by constraining classification of nearby points to not change too much (recall that  $W_{ij}$  encodes nearness of  $x_i$  and  $x_j$ ). The second term constrains classification to not stray too far from the initial assignment of labels. The algorithm tries to maintain global consistency by balancing between the two terms, as  $Q(F)$  penalizes points differing in class from nearby initially labelled ones: the initially labelled points serve as reference anchors for classification. The weighting parameter  $\mu$  respects the fact that these constraints may compete.

It can be shown by differentiating  $Q(F)$  with respect to  $F$  and doing some algebraic manipulation that

$$F^* = \beta (I - \alpha S)^{-1} Y, \quad (3)$$

where  $\alpha = 1/(1 + \mu)$  and  $\beta = \mu/(1 + \mu)$ :  $F^*$ , then, is the matrix that allows for a good classification on  $X$ . Several experiments in [16] show that this algorithm yields good classifications on data sets.

In [17], this algorithm is applied to the task of determining a weighted relevance metric that respects both local and global consistency. In particular, it is applied to show that it yields the same ranking list as Google's PageRank algorithm. Notably, queries and pages are represented as vectors, and a query plays the role of an initially labelled point. What makes the analogy between queries and initially labelled points notable is that, in  $Q(F)$ , labellings constrain the final assignment of labels to points. Looking at labellings in this way suggests a natural use of the algorithm to detect outliers: if labelled points are regarded as paradigmatic instances of a class, and we construct an empirical cdf of the values in  $F^*$  that lead to classifications, those points that lie below a given threshold can be considered to be outliers.

Thus, points distant from labelled points are more prone to being identified as outliers. A significant disadvantage of this approach is that it makes outlieriness relative to our choice of images to label. If we are to use this algorithm to identify outliers objectively, we need to somehow separate the main work of the algorithm from the initial assignment of labels. In the next section, we see how this is done.

## 2.1. Clustering with Local and Global Consistency

From equation (3), it is evident that the solution to the semi-supervised learning problem only depends on the labels after the the matrix  $(I - \alpha S)$  has been inverted. This matrix only contains the training data inputs,  $\{x_1, \dots, x_n\}$ , and it is this property that we will exploit to derive our clustering algorithm. We define a matrix  $U$  as:

$$U = \beta(I - \alpha S)^{-1} = [u_1^T, \dots, u_n^T] \quad (4)$$

and note that  $U$  defines a graph or diffusion kernel (as described in [1, 2]). In addition, the columns of  $U$ , denoted by  $u_i^T$ , define distances between training points on these graphs, which can be interpreted as distances along a manifold [17]. The ordering of these distances along each manifold is maintained independent of scaling. From  $U$ , we create a new matrix  $V$ , by scaling the columns of  $U$  to have unit length. We define this  $V$  matrix as:

$$V = \left[ \frac{u_1^T}{\|u_1^T\|}, \dots, \frac{u_n^T}{\|u_n^T\|} \right] = [v_1^T, \dots, v_n^T] \quad (5)$$

Note that, by definition,  $\|v_i\| = 1$ .

Based on this column normalized matrix  $V$ , we define the proposed version of semi-supervised clustering:

$$F_V^* = VY = [f_{V1}^T, \dots, f_{Vn}^T]$$

Finally, we further normalize the columns of  $F_V^*$  to give:

$$G^* = \left[ \frac{f_{V1}^T}{\|f_{V1}^T\|}, \dots, \frac{f_{Vn}^T}{\|f_{Vn}^T\|} \right]$$

A class label  $l_i$  is assigned to point  $x_i$  as:

$$l_i = j = \arg \max_{j \leq c} G_{ij}^*$$

where  $G_{ij}^*$  is the  $(i, j)$  element of  $G^*$ .

## 2.2. Semi-Supervised Model Selection via Optimization

We define a distance (along a manifold specified by  $U$ ) between points  $x_i$  and  $x_j$  to be:

$$d_M(x_i, x_j) = 1 - v_i v_j^T \quad (6)$$

The intuition behind this distance measure is that two points on a manifold are identical if they have identical ordering of distances to all other points in the manifold. If this is the case for points  $x_i$  and  $x_j$ , then  $d_M(x_i, x_j) = 0$ . Conversely, if  $x_i$  and  $x_j$  have different distances along  $U$  to other points

in the training data, then  $d_M(x_i, x_j)$  will approach 1. This leads to our definition of a distance matrix:

$$\begin{aligned} D_M &= 1 - \begin{pmatrix} v_1 v_1^T & \dots & v_1 v_n^T \\ \vdots & \ddots & \vdots \\ v_n v_1^T & \dots & v_n v_n^T \end{pmatrix} \\ &= \begin{pmatrix} d_M(x_1, x_1) & \dots & d_M(x_1, x_n) \\ \vdots & \ddots & \vdots \\ d_M(x_n, x_1) & \dots & d_M(x_n, x_n) \end{pmatrix} \end{aligned} \quad (7)$$

Next, let  $\mathbf{p}_j$  be the set of points that belong to class  $j$ . Using matrix  $D_M$  we can define the mean distance between points in class  $j$  as:

$$\overline{D}_M^{jj} = E[D_M(\mathbf{p}_j, \mathbf{p}_j)] \quad (8)$$

where  $D_M(\mathbf{p}_j, \mathbf{p}_j)$  denotes all entries of  $D_M$  corresponding to columns and rows of points  $\mathbf{p}_j$  and  $E[\cdot]$  is the average value of these. Similarly, The mean distance between points in class  $j$  and points in class  $k$  is give by:

$$\overline{D}_M^{jk} = E[D_M(\mathbf{p}_j, \mathbf{p}_k)]$$

Given that our goal is to find semi-supervised models that maximize the distances between points in different classes, while minimizing the distances between points in the same class, we can now state the optimization problem we are solving. Specifically,

$$\Omega = \max_{\alpha, \sigma} \left[ E \left[ \overline{D}_M^{jk} \right]_{\substack{k=1, \dots, c \\ j=1, \dots, c \\ i \neq j}} - E \left[ \overline{D}_M^{jj} \right]_{\{j=1, \dots, c\}} \right] \quad (9)$$

## 2.3. Clusters with Independent $\sigma$

There are two problems with the preceding approach. First, the optimizing over all  $\sigma$  and  $\alpha$  can be computationally expensive. Second, every cluster will use the same  $\sigma$ . In our experiments in Section 4, we show this can lead to significant degradation in clustering performance when points in different clusters are not equally spaced, since  $\sigma$  controls the spread of the Gaussians in the affinity matrix  $W$ . To avoid these difficulties, we take a recursive clustering approach, where the dataset is split into two partitions at each step of the algorithm. The criteria used to determine this split is

$$\Omega(2) = \max_{\alpha, \sigma} \left[ \frac{\min(|\mathbf{p}_1|, |\mathbf{p}_2|)}{|\mathbf{p}_1| + |\mathbf{p}_2|} \left( E \left[ \overline{D}_M^{jj} \right]_{\{j=1\}} - E \left[ \overline{D}_M^{jk} \right]_{\substack{k=1 \\ j=2}} \right) \right] \quad (10)$$

where  $|\mathbf{p}_j|$  is the number of points in cluster  $j$ . This optimization encourages points to be split into two partitions of approximately equal size. In addition,  $\sigma$  and  $\alpha$  are chosen such that the first cluster of points consists of points that are tightly clustered on one manifold, while the second cluster of points is far away from the first, but need not be on a single manifold. Thus  $\sigma$  and  $\alpha$  are always optimized with respect to the first manifold.

This type of recursive partitioning continues until clusters have a minimum number of points  $M$ , which is user specified. This produces a set of  $C$  clusters. The user must also specify the desired number of clusters  $K$ . The constraint on  $K$  and  $M$  is that  $K < C$ . The  $C$  clusters are finally combined to give the best  $K$  clusters that each have maximal values of  $\Omega(2)$  in equation (10).

In the next two sections, the description of the proposed algorithm is completed by first defining the concept of cluster outliers in manifold space, and then showing how this is used to define point clusters.

## 2.4. Outlier Detection

We define a cluster independent outlier point to be one that has greatest average distance to all other points. This can be directly calculated from equation (7) by taking the average of the columns of  $D_M$ , and defining an outlier cluster independent vector  $O_d$  as follows:

$$O_d = \frac{1}{n} \left[ \sum D_{M1}^T, \dots, \sum D_{Mn}^T \right] = [O_{d1}, \dots, O_{dn}]$$

where the element  $O_{di}$  is the average distance (in manifold space) between point  $x_i$  and all the other points and  $D_M = [D_{M1}^T, \dots, D_{Mn}^T]$ . Thus by ordering the values of  $O_{di}$  in decreasing order, we order the points from furthest to closest, and the points appearing first in the list constitute the outliers.

Similarly, we can find outliers within a cluster  $j$  by looking at the  $D_M^{jj} = D_M(\mathbf{p}_j, \mathbf{p}_j)$  matrix defined above. Specifically, we obtain an outlier  $O_d^j$  vector for cluster  $j$  as follows:  $O_d^j = \frac{1}{n} \left[ \sum D_{M1}^{jjT}, \dots, \sum D_{Mn}^{jjT} \right] = [O_{d1}^j, \dots, O_{dn}^j]$  where  $O_{di}^j$  is the mean distance of  $x_j$  to all other points in its cluster. Thus the point which has minimum  $O_{di}^j$  is the one which is *most inside* the cluster, while the point that has maximum  $O_{di}^j$  is *most outside* of the cluster.

## 2.5. Finding Points That Define a Cluster

As outlined above, the points  $x_{l_1}, x_{l_2}$  are used to specify clusters  $\Omega(2)$  in equation (10). These points can be identified by looking at the cluster independent outlier vector  $O_d$  defined above. In this paper we use the following greedy

heuristic to identify  $x_{l_1}, \dots, x_{l_c}$ . First we assign  $x_{l_1}$  to the point that is closest to all other points, which is defined by the point that has the smallest value  $O_{di}$ . To find  $x_{l_2}$ , we multiply each element of  $O_d$  by the corresponding element in the column vector  $D_{Ml_1}^T$ , to obtain a new, re-weighted vector of  $O_d^2$  as follows:

$$O_d^2 = [O_{d1}^1 D_{Ml_1}^T(1), \dots, O_{dn}^1 D_{Ml_1}^T(n)] = [O_{d1}^2, \dots, O_{dn}^2] \quad (11)$$

where  $O_{di}^1 = O_{di}$ . The point  $x_{l_2}$  then corresponds to the point which has minimum  $O_{di}^2$ .

## 2.6. Classifying New Points

In order to cluster a new point without adding it to  $S$  and re-inverting the matrix  $(I - \alpha S)$ , we once more use the property that two points are similar if they have similar distances to all other points. However, this time we measure similarity using the  $S$  matrix as follows. Given a point  $x_k$ , we calculate  $W_{kj} = \exp(-\|x_k - x_j\|^2 / (2\sigma^2))$ , for  $j = 1, \dots, n$  and obtain a vector  $W_k$ . We then calculate the  $D_k = \sum_{j=1}^n W_{kj}(j)$  and compute the vector in the  $S$  matrix that is associated with  $x_k$ , as  $S_k = D_k^{-1/2} W D^{-1/2}$ . Finally we normalize  $S_k$  to have length 1 and call it  $S_k^1$  and similarly normalize the rows of  $S$  to also have length 1, denoting this matrix by  $S^1$ . We then obtain a set of coefficients  $\Theta = (\theta_1, \dots, \theta_n)^T = S^1 (S_k^1)^T$ . This vector has the property that if  $x_k = x_i$ , then  $\theta_i = 1$ , but if  $x_k$  is very far away from  $x_i$  then  $\theta_i$  will approach zero. Therefore,  $\theta_i$  measures the closeness of  $x_k$  to  $x_i$  in  $S$  matrix space (with  $\theta_i = 1$  being close and  $\theta_i = 0$  distant). We use this property to assign  $x_k$  to a cluster by creating an  $F_k = [v_1 \Theta^T, \dots, v_n \Theta^T]$  and assigning  $l_k = \arg \max_{j \leq c} F_k$ .

## 2.7. Visualizing Manifolds

In order to visualize which points belong to which manifold, we propose the following one dimensional graph. Starting from equation (7), let  $\mathbf{p}_j = (j_1, \dots, j_K)$  be the set of  $K$  points that belong to class  $j$ . Then, the mean distance, symbolized by  $D_K^j$ , of point  $j_k \in \mathbf{p}_j$  to all other points in the class can be defined as:

$$D_k^j = \frac{1}{K} \sum_{i=1}^K \lambda_{i1}^j \quad (12)$$

where

$$\lambda_k^j = (\lambda_{k1}^j, \dots, \lambda_{kK}^j)$$

is a column vector of the matrix  $D_M(\mathbf{p}_j, \mathbf{p}_j)$  (see equation (8)) as defined by:

$$\Gamma^j = [\lambda_1^j, \dots, \lambda_K^j] = D_M(\mathbf{p}_j, \mathbf{p}_j)$$

Within our framework, points in a class that have similar values of  $D_K^j$  belong to the same manifold, which allows a plot of these values as a function to identify manifolds. An example of this is given in Section 4.

Examples of these visualizations are given in Figure 2 for the indoor images in Figure 1, and in Figure 4 for the outdoor images in Figure 3.

### 3. Semi-Supervised Learning For Robotics

We apply the above Semi-Supervised learning algorithm to robot navigation as follows:

**STEP 1: Robot Exploration:** The robot explores the environment (either autonomously or via teleoperation) and at each time step  $i$ , a set of  $m$  sensor readings is recorded, denoted by  $\mathbf{x}_i = (x_{i1}, \dots, x_{im}) \in \mathbb{R}^m$ . For  $n$  time steps, the sensor input values are denoted by  $\mathbf{x}_1, \dots, \mathbf{x}_n$ .

**STEP 2: Build Manifold Cluster Model:** The first sensor input  $\mathbf{x}_1$  is classified as belonging to one section of the sequence, and the last sensor input  $\mathbf{x}_n$  is classified as belonging to another section. This labelling is arbitrary and simply allows the algorithm to separate out the manifold clusters. This amounts to setting  $Y_{11} = 1, Y_{n2} = 1$  and all other values of the  $n$  by 2 matrix  $Y$  to zero. We then solve the optimization problem in equation (10), and plot  $D_M^{jj}$  from Equation 8 (see Figures 2 and 4).

**STEP 3: Label Representative Manifolds:** Equation (12) is then used to visualize the manifold structures in each class by plotting  $D_K^j$  values for each class. The human operator is then asked to label a single image from each manifold.

**STEP 4: Build Semi-Supervised Model:** These labelled points are then used to build the final model (once more by minimizing equation (10)) that maps sensor inputs to locations in the robot’s workspace.

## 4. Experimental Results

We have tested the constant and variable  $\sigma$  versions of our algorithm on indoor and outdoor image sequences.

In all of the experimental results reported here the optimization function used is the constrained optimization matlab function `fmincon()`, where  $0 \leq \alpha \leq 1$  and  $0 \leq \sigma \leq (1/m)$ . Other than the ranges for search in  $\alpha$  and  $\sigma$ , the learning algorithm presented in this paper has no parameters that need to be set.

We experiment with both indoor and outdoor environments.

### 4.1. Indoor and Outdoor Environments: Fixed $\sigma$

For sensor inputs we used 320x240 images, thus each sensor reading  $\{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subset \mathbb{R}^m$  has a dimension of  $m = 320 \times 240 = 76,800$  pixels. Images were captured using a UniBrain Fire-i 1394 camera. Examples can be seen in Figures 1 and 3.

After Step 3 of the constant  $\sigma$  algorithm given in Section 3, we obtain the plots in Figure 2. Approximately continuous regions in this plot refer to manifolds in visual input space. The arrows indicate the images that are labelled as belonging to a manifold. These labelled images are used to automatically label the remaining images in the manifold. Thus a total of 8 labelled images are sufficient to label all 500 images in our sequence. Therefore only 1.6% of the images require labelling.

A typical image sequence on a manifold is shown in Figure 1. Note that the images show a gradual sequence of motion and the first image is significantly different from the last.

For the outdoor scene, after Step 3 of the algorithm given in Section 3, we obtain the plots in Figure 4. As with the indoor scenes, approximately continuous regions in this plot refer to manifolds in visual input space. The arrows indicate the images that are labelled as belonging to a manifold. These labelled images are used to automatically label the remaining images in the manifold. Thus a total of 6 labelled images are sufficient to label all 400 images in our sequence. Therefore only 1.5% of the images require labelling.

A typical image sequence on a manifold is shown in figure 3. Note that the images show a gradual sequence of motion and the first image is significantly different from the last.

### 4.2. Indoor and Outdoor Environments: Variable $\sigma$

For the variable  $\sigma$  approach we used a dataset consisting of 6 sets of 320 by 240 RGB images (therefore each image is defined by  $3 \times 320 \times 240$  pixels). Each sequence contains 150 images, for a total of 900 images. These images are summarized in Figure 6. Each row in Figure 6 shows one of 8 clusters. The first cluster (in the first row) shows images down an indoor hallway. The second cluster (in the second row) shows images obtained after exiting the hallway and turning left. The third cluster (in the third row) shows images obtained after exiting the hallway and continuing straight. The fourth cluster (in the fourth row) shows images obtained after exiting the hallway and turning right.

The last 4 rows show outdoor scenes. The fifth cluster (in the fifth row) shows images obtained while walking on a lawn towards a driveway. The sixth cluster (in the sixth



Figure 1. Images along a single manifold in an indoor environment.

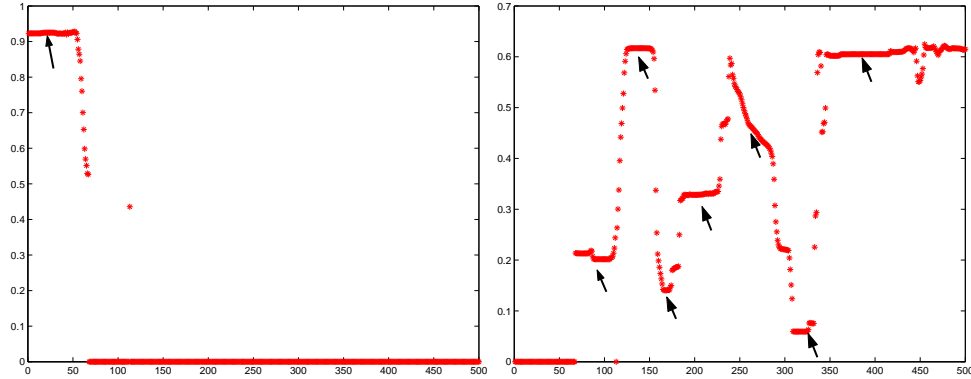


Figure 2. Images along a single manifold in an indoor environment. Arrows indicate the images that are labelled as belonging to a manifold.

row) shows images obtained after leaving the lawn and turning left. The seventh set of images (in the seventh row) shows images obtained after leaving the lawn and continuing straight. Finally, the eighth cluster (in the eighth row) shows images obtained after leaving the lawn and turning right.

The order of these images was randomized and the clustering errors obtained by the algorithm proposed in this paper were compared to those obtained by the Randomized Cuts algorithm. The results obtained using the proposed algorithm for this dataset are summarized in Figure 5a setting  $K = 8$  clusters and  $M = 75$  minimum points per cluster. The 8 clusters are defined on the Y axis. The red stars indicate the true labels and blue squares indicate the predicted labels. The clustering error of the proposed algorithm is 0.039.

We compare these results those obtained with the Normalized Cuts Algorithm [11]. This algorithm uses a single sigma set to 186.5 (experimentation indicated that this sigma resulted in the best clustering error). The clustering error for the Normalized Cuts algorithm was 0.453.

Thus the algorithm proposed in this paper gave a 91% improvement in clustering error over the Normalized Cuts algorithm by allowing each cluster to individually optimize for sigma.

The running time of the variable  $\sigma$  algorithm on this dataset was 32 minutes on a 1.8 GHz Pentium 4 running Windows XP.

## 5. Summary and Conclusions

We have demonstrated two Semi-supervised learning techniques for transforming a free form video sequence into clusters on low dimensional manifolds in sensor space with minimal labelling by human experts.

The first approach uses fixed affinity matrix parameters ( $\sigma$ ). For the experiments in Section 4 the indoor sequence of 500 images required labelling of only 6 images and the 400 image outdoor scene, only 8 images. The resulting set of independent clusters forms a topological map of the space traversed by the sequence.

The second algorithm allows each cluster to independently optimize its own affinity matrix parameters. The algorithm results in a spectral clustering-like procedure that can significantly outperform algorithms which constrain all clusters to have the same affinity matrix parameters. The proposed algorithm requires the user to specify the desired number of clusters and the minimum number of points per cluster - the affinity matrix parameters are completely defined by these two parameters.

This paper suggests a number of open research questions. First, the optimization procedure posed was solved using an off the shelf optimization algorithm (matlab optimization in 2D). Reanalyzing the theory behind this optimization may allow a more direct solution to this optimization problem. Second, the user is required to specify the number of desired clusters. The framework developed here may allow the number of clusters to be directly optimized for. Finally, the initial results presented appear promising and further experimental investigation appears warranted.

Of course a map is only useful to a robot if it can be



Figure 3. Images along a single manifold in an outdoor environment.

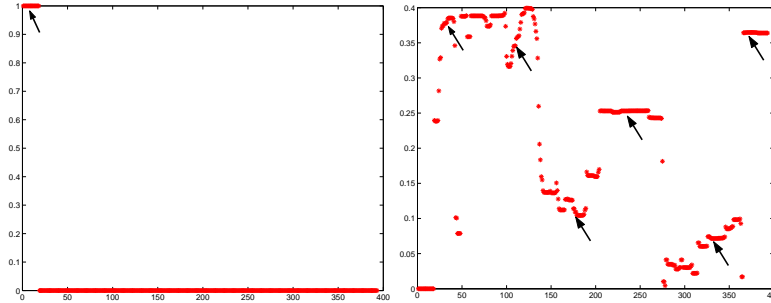
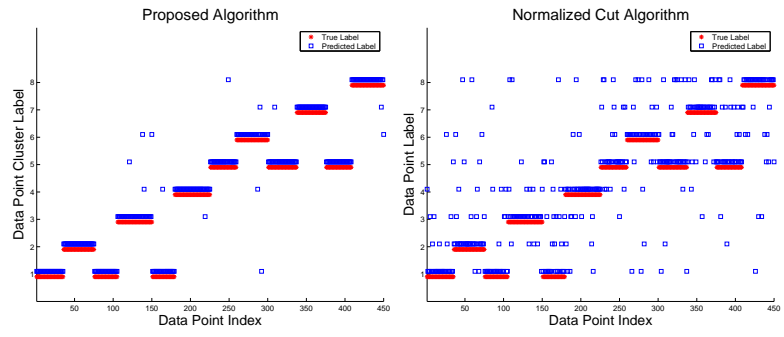


Figure 4. Images along a single manifold in an outdoor environment.

used to improve navigation. Our future work will address navigation on the manifold. Since the image sequences correspond to low dimensional structures our goal is to identify the dimensions of these structures and use them as a guide to servo along the manifolds.

## References

- [1] D. DeCoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46:161–90, 2002.
- [2] J. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In *Advances in Neural Information Processing Systems (NIPS)*, volume 15, 2003.
- [3] A. Kelly. Mobile robot localization from large scale appearance mosaics. *International Journal of Robotics Research*, 19(11):1104–1125, 2000.
- [4] B. J. A. Kröse and R. Bunschoten. Probabilistic localization by appearance models and active vision. In *Proc. of the 1999 IEEE International Conference on Robotics and Automation*, pages 2255–2260, Detroit, MI, May 1999.
- [5] B. Kuipers and P. Beeson. Bootstrap learning for place recognition. In *Proc of the 18th Natnl Conf on AI (AAAI02)*, 2002.
- [6] A. Leonardis, H. Bischof, and J. Maver. Multiple eigenspaces. *Pattern Recognition*, 35(11):2613–2627, 2002.
- [7] S. Maeda, Y. Kuno, and Y. Shirai. Active navigation vision based on eigenspace analysis. In *Proc. 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1018–1023, 1997.
- [8] Y. Matsumoto, M. Inaba, and H. Inoue. View-based approach to robot navigation. In *Proceedings of 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'2000)*, pages 1702–1708, Takamatsu, Japan, Nov 2000.
- [9] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)*, volume 14, 2002.
- [10] E. Remolina and B. Kuipers. Towards a general theory of topological maps. *Artificial Intelligence*, 152:47–104, 2004.
- [11] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2002.
- [12] R. Sim and G. Dudek. Comparing image-based localization methods. In *Proc Int. Joint Conf. on AI (IJCAI03)*, 2003.
- [13] S. Thrun. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.
- [14] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proc of the 2000 IEEE Int. Conf. on Robotics and Automation*, pages 1023–1029, San Francisco, CA, April 2000.
- [15] D. Verma and M. Meila. A comparison of spectral clustering algorithms. Technical Report 03-05-01, Department of Computer Science and Engineering, University of Washington, 2003.
- [16] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems (NIPS)*, volume 16, 2004.
- [17] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *Advances in Neural Information Processing Systems (NIPS)*, volume 16, 2004.



a) Proposed variable  $\sigma$  Algorithm      b) Normalized Cuts Algorithm

Figure 5. Results on Image Data.

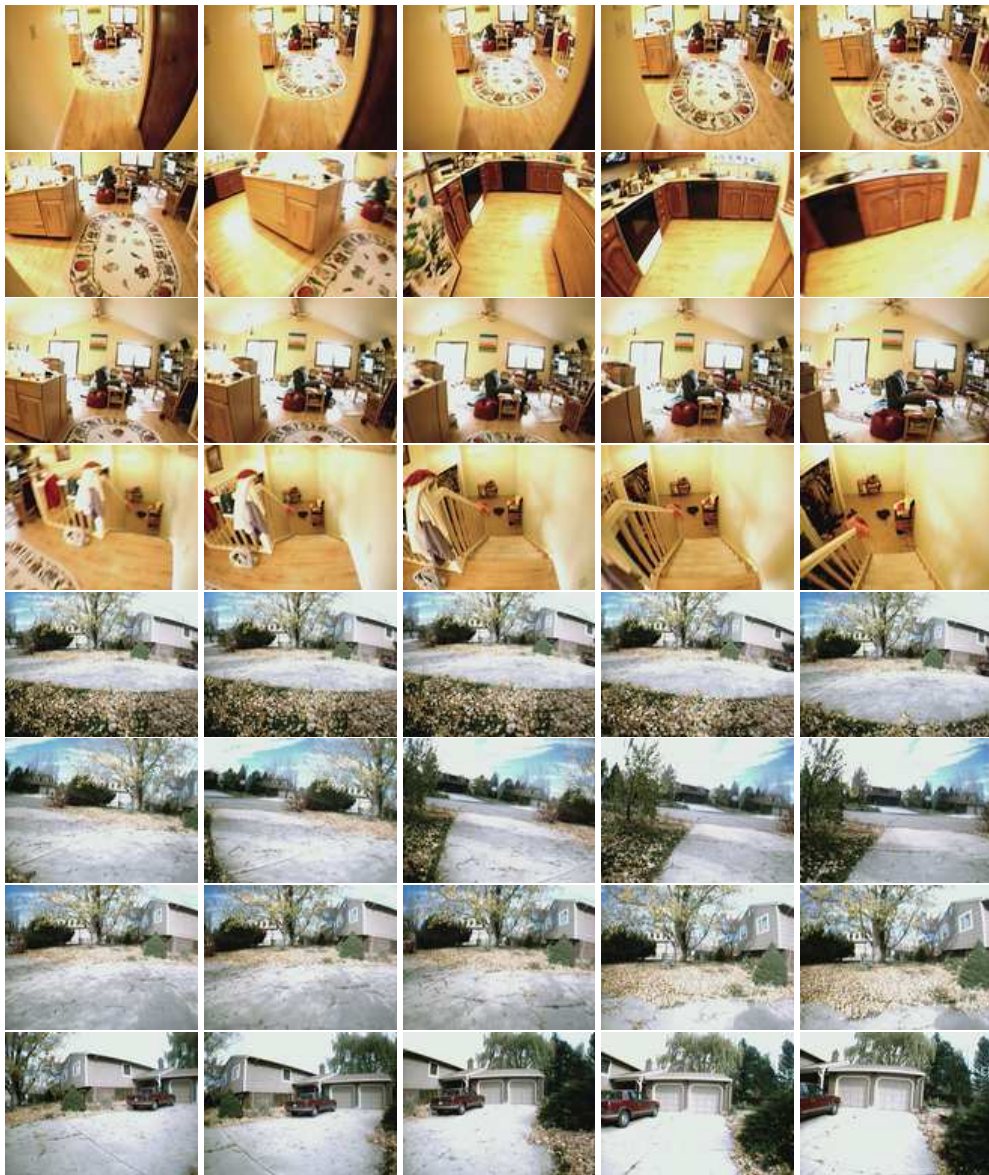


Figure 6. Images along a 8 manifolds in indoor and outdoor environments.