Sensor Map Discovery for Developing Robots

Jeremy Stober and **Lewis Fishgold**
Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, TX 78712-0233
{stober,lewfish}@cs.utexas.edu

Benjamin Kuipers
Computer Science and Engineering
University of Michigan
2600 Hayward Street
Ann Arbor, Michigan 48109
kuipers@umich.edu

Abstract

Modern mobile robots navigate uncertain environments using complex compositions of camera, laser, and sonar sensor data. Manual calibration of these sensors is a tedious process that involves determining sensor behavior, geometry and location through model specification and system identification. Instead, we seek to automate the construction of sensor model geometry by mining uninterpreted sensor streams for regularities.

Manifold learning methods are powerful techniques for deriving sensor structure from streams of sensor data. In recent years, the proliferation of manifold learning algorithms has led to a variety of choices for autonomously generating models of sensor geometry. We present a series of comparisons between different manifold learning methods for discovering sensor geometry for the specific case of a mobile robot with a variety of sensors. We also explore the effect of control laws and sensor boundary size on the efficacy of manifold learning approaches.

We find that ”motor babbling” control laws generate better geometric sensor maps than mid-line or wall following control laws and identify a novel method for distinguishing boundary sensor elements. We also present a new learning method, *sensorimotor embedding*, that takes advantage of the controllable nature of robots to build sensor maps.

Introduction

Accurate sensor, motor, and world models are a crucial component in the design of effective control laws for autonomous robots. For robots undergoing autonomous development, these models must be generated from data since no external observer is available. We will focus on the subproblem of learning the position of each sensor element within a sensor array in an unsupervised fashion.

We can view the problem of a robot situated in the world as a dynamical system

$$x_t = G(x_{t-1}, \mu_{t-1}) \tag{1}$$
$$s_t = H(x_t) \tag{2}$$
$$\nu_t = C(s_t) \tag{3}$$
$$\mu_t = M(\nu_t) \tag{4}$$

where G is a function representing the effect robot motion μ has on the state of the world x, H is a sensor model that determines the observable state s, C is the motor control law adopted by the robot, and M is the (possibly noisy) realization of motor commands ν as actual motions μ. We denote the composition $M \circ C \circ H$ by ψ. Then the state evolution of the complete robot-world system is given by:

$$\{G(x_t, \psi(x_t))\}_{t \in T} \tag{5}$$

A roboticist faces many challenges in trying to define control laws that result in desirable trajectories of system evolution. We seek to reduce this burden through mechanisms that allow robots to learn about themselves and the environment from experience.

This leads to a related question: What can an agent discover from behind an interface of an unknown body (ψ) in an unknown world (G) (Philipona et al. 2004)? Even for the limited case of autonomously learning sensor geometry, progress has many potential applications to the present and future science of robotics. For instance, as self-reconfigurable robotic systems scale (and component parts become smaller) so will the need for algorithms to robustly and autonomously handle failure modes (in our case sensor configuration failure modes) (Yim et al. 2007). In the limit of this technology, we suspect that the self organizing processes will only provide loose constraints on the geometry of sensor arrays for resulting large scale robot systems. Knowledge of exact geometry that we take for granted today will have to be learned in the future.

Even with the current state of the art components, where intrinsic sensor parameters are known, many robots are bespoke combinations of these components. We would like to automate the process of identifying the relative locations of sensor components on robots. Finally, sensor geometry discovery is a unique problem for testing the properties of manifold learning methods. Some of the typical properties of problem instances include

- large numbers of sense elements of varying modality and geometry
- many possible low dimensional target spaces (e.g. sensor location, robot pose)
- a (physically constrained) controllable system.

The heterogeneous and high dimensional nature of sensor traces from autonomous robots makes for a plentiful source of data for which to perform comparative analysis of existing manifold learning methods. From the developmental robotics perspective, where the robot is expected to learn as much as possible about the world from data alone, manifold learning provides one valuable method for gleaning structure from the firehouse of data available to nascent robot agents.

In addition, the controllable nature of autonomous robots presents some interesting possibilities beyond typical applications of manifold learning, where the data collection is normally a passive process. We will examine the effect of robot policies on sensor reconstruction as well as introduce a novel method of sensor model discovery that depends directly on robot controls.

Discovering Sensor Geometry

A crucial step in developing a sensor model from data is discovering the sensor geometry. For a given sensor s_i, denote the time-series of sensor readings for s_i by S_i. Given a metric δ for comparing sensor time series, we can construct a matrix Δ whose entries are sensor time-series differences $\delta(S_i, S_j)$ for each pair of sensors i and j.

If we view the sensor time-series S_i as points in a high dimensional space, we can apply a manifold learning technique to find a low dimensional embedding that preserves the inter-point distances of Δ. Since sensor traces S_i are inexact proxies for sensor element positions, we want our embedding method to be robust to noise. For this problem domain, we can vary the metric used to compare sensor time-series S_i, or vary the embedding technique. We adopt the standard Euclidean metric in the results below, and concentrate our effort on comparing the results of different embedding methods.

Comparing Manifold Learning Methods

We begin by comparing the reconstructions using Procrustes analysis (Seber 1984). This method of comparison first performs a linear transformation of each reconstruction consisting of any of a translation, orthogonal rotation, reflection, and scaling to best match the ground truth data. We then measure the remaining sum of squared errors between the transformed reconstruction and our ground truth data. Intuitively, this method of analysis allows us to limit our evaluation to differences in shape between reconstructed sensor maps and ground truth information (Figure 3). Though used here only for comparison with low-dimensional ground truth data, Procrustes analysis can be used for manifold alignment of two differing high dimensional datasets, as in (Wang and Mahadevan 2008).

We performed a series of experiments to compare the performance of six manifold learning algorithms on data generated by six different sensors. We evaluated

- Classical Multidimensional Scaling (MDS) (Kruskal and Wish 1978)
- Isomap (Tenenbaum, Silva, and Langford 2000)
- Locally Linear Embedding (LLE) (Roweis and Saul 2000)
- Hessian LLE (HLLE) (Donoho and Grimes 2003)
- Laplacian Eigenmaps (Laplacian) (Belkin and Niyogi 2003)
- FastMVU (Weinberger et al. 2007)

For all methods, we used the implementations of these algorithms provided in a freely available MATLAB Dimensionality Reduction Toolkit (van der Maaten 2007). Unless noted, the parameters for each algorithm were the default ones provided by the toolkit. In particular, we used a default neighborhood size of 12. In the datasets, each point represents a sensor element, and the number of dimensions is equal to the number of measurements made by each sensor element. We tested six datasets as follows.

- (Real Lasers) - A planar laser rangefinder mounted on a real mobile robot was used to collect range measurements at 180 one degree intervals along a semi-circle.
- (Sim. Lasers) - A planar laser rangefinder mounted on a simulated mobile robot was used to collect range measurements at 360 one degree intervals.
- (Sim. Lasers w/ Noise) - The simulated laser dataset was corrupted with Gaussian noise with mean=0 and variance=5. The values of each sensor ranged between 0 and 10, so this noise is quite substantial.
- (Half-Lasers) - A contiguous set of 180 sensors was deleted from the laser dataset to simulate a bank of sensors arranged in a semicircle.
- (Roving Eye) - A 10x10 “roving eye” of sensors arranged in a grid was scanned across an image containing a natural scene.
- (Sim. Sonar) - A set of 16 simulated sonar sensors arranged uniformly in a ring around a mobile robot were used to collect range measurements. Note that the semantics of sonar range sensors differ from laser range sensors in that sonar responds to obstructions in a cone region, whereas lasers are point estimates.

Figure 1 shows the environments and platforms used to collect each dataset.

The results of our comparison (Table 2) do not indicate that a single approach generates the best reconstruction across all data sets. Classical MDS performed best in both scenarios involving 180° laser rangefinders. As we see in Figure 3 with a Laplacian eigenmap reconstruction, the non-linear methods produce reconstructions that have more pronounced bowing than the ground truth data, resulting in a high residual error after Procrustes analysis. For full 360° laser rangefinders, both with and without noise, non-linear methods produced better reconstructions than classical MDS. Some of these methods, HLLE in particular, took longer to process, which may be an issue given the resource and time constraints of a developing robot (See Table 1).

FastMVU produced reconstructions with differing errors on different runs, indicating some variability in the search for local reconstruction refinements. Maximum variance

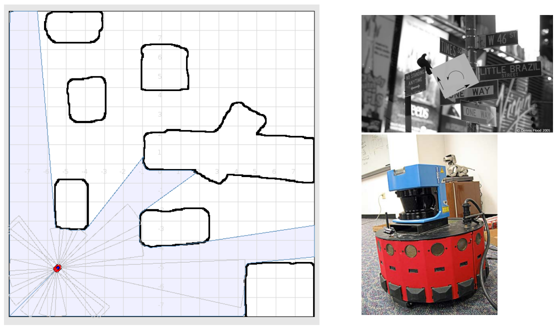


Figure 1: The environments used to collect data for reconstruction comparison. We used Player/Stage to collect our simulated laser and sonar data in the environment shown on the left. The image used to collect roving eye data is shown in the top right. The Pioneer robot used to collect physical sensor data is shown in the bottom right. In each case the data was collected using a random exploration policy.

unfolding methods alone produced unfeasible semi-definite program sizes.

In cases where sensors have a large boundary, such as the roving eye sensor, HLLE generates a better reconstruction. In particular, HLLE reconstructions seem to handle sensor configurations with large boundaries. Given the trade offs involved, this suggests that a method of identifying the relative size of a sensor array's boundary prior to applying HLLE may be useful. A method for identifying sensor boundaries may have other uses as well. For example, sensor boundaries are likely places for *sensor augmentation*. We will discuss a method for identifying sensor boundary elements in the next section.

Applying Isomap to half-laser datasets did not result in accurate reconstructions. For simply connected manifolds like the half laser arrays, using geodesic distances results in a correct one dimensional reconstruction. Unfortunately, using geodesic distances does not preserve enough information about the semi-circular shape of the laser arrays for correct two dimensional reconstructions.

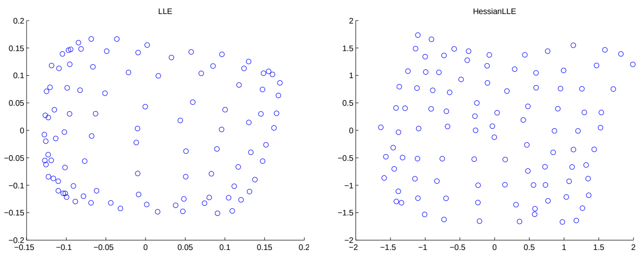


Figure 2: A comparison of LLE (left) and HLLE (right) reconstructions for the roving eye dataset. HLLE requires more running time, but produces a more accurate reconstruction, and does not show warping around the boundary of the sensor array.

Method	Time (s)
Classical MDS	1.21
Isomap	3.33
LLE	1.26
HLLE	131.46
Laplacian	1.04
FastMVU	4.47

Table 1: Running time comparison between methods on the simulated 360° laser rangefinder dataset.

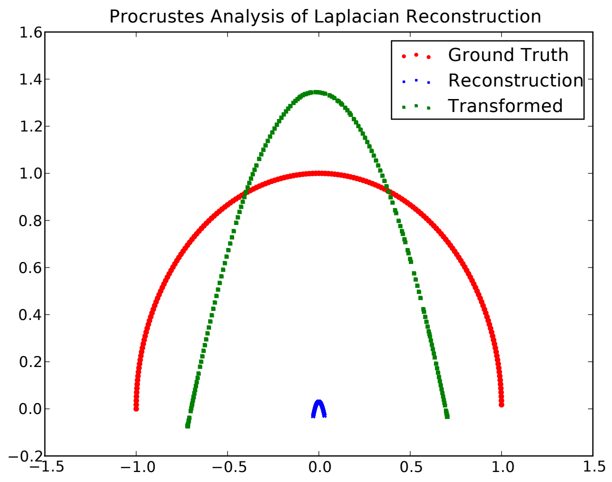


Figure 3: We use Procrustes analysis to correct for scale, rotation, and translation errors between the reconstruction and the ground truth data. This allows us to measure the shape discrepancy between reconstructions and ground truth as the remaining mean squared error after correcting for scale, rotation, and translation. In this example, the transformed laplacian reconstruction is considerably more bowed than the ground truth data, and points near the boundary are compressed, leading to high remaining error after Procrustes analysis.

Identifying Boundary Sensor Elements

We adopt a method originally used for detecting outliers presented in (Choi and Choi 2007) to detect sense elements near sensor boundaries. First, we compute all shortest geodesic paths as in the initial steps of Isomap. Next, we compute the network flow for each edge ϵ_k as

$$\eta(\epsilon_k) = \sum_i \sum_j \theta(\epsilon_k, i, j) \tag{6}$$

where

$$\theta(\epsilon_k, i, j) = \begin{cases} 1 & \text{if } \epsilon_k \in path(i,j) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Here each $path(i, j)$ is the shortest geodesic path between nodes i and j. Intuitively, we are counting the number of times each edge is included in a shortest path. The network

Method	Real Lasers	Sim. Lasers	Sim. Half-Lasers	Sim. Lasers w/ Noise	Roving Eye	Sim. Sonar
Classical MDS	0.0086	0.0599	0.0612	0.0871	0.1591	0.0714
Isomap	0.1322	0.0094	0.174	0.0454	0.0362	0.27
LLE	0.1409	0.0027	0.1433	0.0611	0.0373	0.0688
HLLE	0.1454	0.0035	0.1427	0.0343	0.0144	0.8221
Laplacian	0.1479	0.0006	0.1464	0.0146	0.0666	0.0777
FastMVU	0.0309	0.4766	0.2235	0.1216	0.1114	0.3242

Table 2: A comparison of manifold learning methods. All errors are the Procrustes distances between sensor geometry reconstructions and ground truth data.

flow of a node v_l is defined as the sum of the network flows of all the node's incident edges, which we denote by $\mathcal{E}_{v_l}$

$$f(v_i) = \sum_{\epsilon_k \in \mathcal{E}_{v_l}} \eta(\epsilon_k) \tag{8}$$

In (Choi and Choi 2007), the authors use the network flow of a node to identify "short circuit" nodes that connect two disparate parts of a manifold surface. These short circuit nodes are a bottle neck for shortest paths, and so would have higher than normal network flows. Removing these nodes, characterized as network flow outliers, results in better Isomap embeddings if the outliers are a result of noise, and not part of the underlying manifold structure.

We hypothesize that boundary nodes should have lower network flows than interior manifold points. In practice however, many interior manifold points also have low network flows. Being on the interior of the manifold does not mean that a node is necessarily connected to an edge that is included in many shortest paths. We observe, however, that interior points are almost always within some neighborhood of a node with high network flow, so instead of using network flow of a node as our boundary criteria, we use network neighborhood flow by summing the network flows for each nodes' original neighborhood $\mathcal{N}_{v_l}$

$$n(v_l) = \sum_{v_i \in \mathcal{N}_{v_l}} f(v_i) \tag{9}$$

In Figure 4 we show the results of this computation for 180° laser rangefinder and 360° sonar rangefinder traces taken from a physical robot. The laser rangefinder arrangement has two boundary points. We note that neighborhood networks flows reach a minimum in neighborhoods of these two boundary regions. The sonar arrangement has no boundary points, and the resulting neighborhood flows show no discernible pattern indicating a boundary.

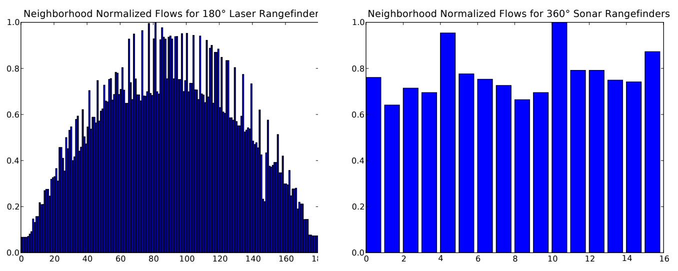


Figure 4: For the sonar sensors (right) whose placement approximates a manifold without boundary, the normalized neighborhood flows show no discernible pattern. For the laser rangefinders (left), whose arrangement contains two boundaries, the normalized neighborhood flows are smaller for boundary regions. All sensor data is from a physical Pioneer robot trace recorded using a random exploration policy in a square environment.

Policies and Reconstruction Error

Unlike passive sensor networks, autonomous robots are controlled systems. We need to consider the effect of the controlling policy on sensor reconstruction. In autonomous mental development, motor babbling is often used as the initial behavior for the robot. This is often justified as a plausible initial policy by explicitly comparing motor babbling to theories of infant development (Olsson, Nehaniv, and Polani 2006).

However, we would like to verify that random exploration as a policy choice leads to reconstructions that are at least as good as any other choice of controlling policy. Doing so would indicate that the apparent randomness of motor (Demiris and Dearden 2005) or body (Meltzoff and Moore 1997) babbling is also justified as a behavior that generates good reconstructions, not just as a behavior that mimics biological development.

To test the hypothesis that motor babbling leads to good reconstructions, we collected sensor traces from a simulated robot controlled by three different policies. We then compared the reconstructed sensor geometries using Procrustes analysis as above. We generated each reconstruction using classical MDS. We used two deterministic policies of varying complexity, and a random policy as in the previously examined sensor traces. The robot paths that resulted from each policy, along with the reconstructions, are shown in Figure 5.

Even in the sparse environment we simulated, a random policy generated a sensor trace which resulted in a more accurate sensor map reconstruction than either deterministic policy. Even for the deterministic policies, the policy that generated the more complex path resulted in a better reconstruction.

From the results, we conclude that controlling policies on a mobile robot need to induce a certain amount of variation in the sensor stream in order for manifold learning methods to find good sensor reconstructions. This runs counter to the goal of many controlling policies, which often explicitly try to maintain invariance among certain sensor properties (e.g. a mid line controller attempts to maintain equal left and right sensor readings). This suggests that for robots undergoing autonomous mental development, once a robot achieves a level of development beyond motor babbling, manifold learning methods are less effective. In particular, without explicit changes in control, manifold learning methods may not serve as a tool for identifying or diagnosing changes to sensor geometry after initial sensor map discovery.

Sensorimotor Embedding

Though the control policy of an autonomous robot effects the manifold learning methods described above, the controls themselves provide considerable additional information that we would like autonomous agents to exploit in the construction of sensor maps. We consider one method of doing so, *sensorimotor embedding*, which learns to associate with each sense element in a sensor map, a distinguished control signal that brings about a desirable change in sensory experience. We then interpret the learned control signals as locations for each sense element.

To apply this method, however, requires that we specify a reward signal that distinguishes desirable changes in sensory experience. Deciding what constitutes a desirable sensory change is difficult in general (particularly in the absence of sensor geometry) but we consider a particular sensor geometry where such a reward signal is easy to specify, a *foveated retina*. In a foveated retina, the density of sense elements is low around the periphery of the sensor and increases towards a maximum at the center, or *fovea*, of the sensor. In addition, the foveated retina can move, or *saccade*, around a scene. For this sensor geometry, we can specify a reward signal that tries to maximize the total activity of the sensor elements. Such a maximum is achievable by moving any visible activation to the fovea, where the sensor density is greatest. Our implementation of a "roving eye" foveated sensor is shown in Figure 6.

Formally, we require that each sense element implements an activation function $\delta : \mathcal{I} \times \mathcal{S} \to [0, 1]$. In our experiments using a "roving eye" fovea, each sense element observes an image patch I_k. $\delta(I_k, s)$ is the total brightness of the pixels in the image patch I_k observed by the k^{th}-sensor element given the current retina position s, normalized to $[0, 1]$ as a fraction of the maximum possible activation.

The activation over the entire retina is the sum of the activations for each sensor element for the current retina state,

$$R_{\mathcal{I}}(s) = \sum_{I_k \in \mathcal{I}} \delta(I_k, s) \tag{10}$$

In our computational model, saccades result in 2D displacements of the image on the retina or pan/tilt changes for a physical camera. Each action or saccade $a : \mathcal{S} \to \mathcal{S}$ is described by two-element vector denoting horizontal and vertical motion and results in a single globally rigid transformation of the image or scene.

If the sensor elements in the retina are of uniform size and distribution, and they are exposed to input consisting of a small spot of light against a uniform background, then $R_{\mathcal{I}}(s)$ would be approximately constant for all retinal states s, regardless of where the spot of light falls. However, with a *foveated* retina,$R_{\mathcal{I}}(s)$ will have a dramatic maximum for retina states that cause the spot of light to fall on the fovea, due to the larger density of sensor elements there.

Using the total activation of all the sensor elements for the current retina state, $R_{\mathcal{I}}(s)$ in Equation 10 as the reward, combined with saccade actions, we can define a simple reinforcement learning problem, the goal of which is to find a policy, or choice of action, that maximizes retinal activation.

We factor the global learning problem into an individual learning problem for each sensor element. The goal of each sensor element is to learn a policy that greedily maximizes the total retinal activation $R_{\mathcal{I}}(s)$,

$$\pi_k(s) = \arg_a \max R_{\mathcal{I}}(a(s)) \tag{11}$$

The problem is episodic and spans a pre- and post-saccadic state. The collective policy π^* for the entire retina is the weighted average of the actions preferred by the individual receptive fields,

$$\pi^*(s) = \frac{1}{\mathcal{R}_{\mathcal{I}}(s)} \sum_{I_k \in \mathcal{I}} \delta(I_k, s) \cdot \pi_k(s) \tag{12}$$

In this factored learning problem, the only information a sensor element has about the state of the retina is the intensity level for that sensor element's visible patch I_k. If the intensity is high ($\delta(I_k, s)$ is close to 1), then the policy $\pi_k(s)$ will have a large impact on the global policy calculated in Equation 12. In this case, we want the policy to suggest an action $\pi_k(s) = a$ that maximizes the reward $R_{\mathcal{I}}(a(s))$. The action that accomplishes this takes the activation that the current sensor element sees and shifts it to the fovea, where the density of sensor elements is higher.

If the intensity is low, then the policy for that sensor element will have little impact on the policy for the entire retina since $\delta(I_k, s)$ is close to zero. As a consequence, we can treat $\pi_k(s)$ as a constant. So in the factored problem, each sensor element only needs to estimate the optimal action and observe its own intensity level.

After sufficient training, the action specified by π_k will approximate the saccade that moves an image-point from sensor element k directly to the fovea. Consider the inverse $-\pi_k$ of the policy estimate for each sensor element. This is the action that would move an image-point from the fovea to the sensor element k. In other words, the inverse of the policy is a position for the sensor element relative to the fovea. Physically proximate sensor elements should have similar saccade policies, and hence similar learned positions. Note that we have not used any knowledge of the location of sensor elements within the fovea. In fact, that knowledge has been learned by the training process, and is encoded in the policy π_k. Spatial knowledge that was implicit in the anatomical structure of the retina becomes explicit in the policy.

The reinforcement learning problem described above has two unusual properties that constrain the choice of learning algorithm. First, the action space is continuous (as opposed to small and discrete). Second, the problem is episodic, and each episode spans only one choice of action.

During learning, each sensor element maintains an estimate for π_k, the current best action, and R_k, the current maximum estimated reward after performing the current best action. Initially, each π_k is set to a random action, and the reward estimate is initialized to zero.

At the beginning of each iteration or training, we randomly reposition the retina. For exploration, some noise ϵ is added to the current greedy policy. The retina agent executes $\pi^*(s) + \epsilon$, and measures the reward (R). Each individual sensor element's reward estimate and current policy are updated proportional to its state activation prior to the saccade ($\delta_k = \delta(I_k, s)$) since the optimal policy π^* is weighted according to those activations. We use a moving average learning rule to update both the reward estimate and current policy. For each sensor element k, we update the reward as follows

$$R_k^{new} = R_k^{old} + \delta_k \cdot \alpha \cdot (R - R_k^{old}) \quad (13)$$

If the reward received, R, is greater than our current reward estimate, we move the current policy π_k for that sensor element closer to the global policy responsible for the increased reward

$$\pi_k^{new} = \pi_k^{old} + \delta_k \cdot \alpha \cdot (\pi^* - \pi_k^{old}) \quad (14)$$

By varying the learning rate α, we can change how much recent experience affects both the estimate of reward (R_k) and the estimate of the optimal saccade (π_k) itself. We evaluated this approach to learning sensor maps using a physical pan/tilt camera and a simulated foveated retina. The camera was exposed to a single bright point of light in an otherwise dark room. The learned sensor element locations are shown in Figure 7.

We note that unlike in our previous analysis of manifold learning methods, the scale of the sensorimotor embedding has meaning. Sensorimotor embedding does not just produce sensor array shapes, but a geometry that is grounded in the properties of the controls. Though limited to situations where reward is easily specified, this method of producing sensor maps yields a more detailed, semantically meaningful picture of sense element organization, since it is able to exploit the controllable nature of the robot system.

Moreover, the found geometry is plastic, in the sense that physical changes that impact the reward (such as some kind of lesioning), will result in policy adjustments, and by extension adjustments to sensor geometry (Figure 8).

Related Work

In (Pierce and Kuipers 1997), the authors explored a method for reconstructing sensor geometry and primary motor effects for a circular array of sonar sensors on a mobile robot with a differential drive and a roving eye sensor traversing a fixed image. The authors used classical MDS to generate the desired sensor maps. In (Olsson, Nehaniv, and Polani 2006), the authors adapt the work of (Pierce and Kuipers 1997) to generate low dimensional sensor and motor models for the AIBO robot, using an information theoretic distance metric.

In other work (Philipona, O'Regan, and Nadal 2003; Stronger and Stone 2006), the learning focus is on the interaction between sensor and motion models. In (Stronger and Stone 2006), the sensor signal under consideration (and by extension H) is already a low-dimensional (albeit uncalibrated) representation of salient world state features. The autonomous robot's task is to develop an internally consistent model of the effect of actions on sensors.

Sensor geometry reconstruction is a deeply studied problem in distributed sensor networks. For the reconstruction problem on a mobile robot, all interpoint distances between sensor streams, which serve as proxies for sensor element positions, are known, though proxy distances may only be effective over small neighborhoods of sense elements. In distributed sensor networks, only a sparse subset of sensor element distances are typically known.

Sparsity in distance information necessitates the use of additional constraints for accurate low dimensional embeddings. Rigidity is one such constraint that has been explored in several papers (Eren et al. 2004; Priyantha et al. 2003) on distributed sensor networks, as well as in the manifold learning literature (Singer 2008).

In addition, robust behavior with noise in the sensor distances, which is also a concern for our mobile robot scenario, has been explored using various regularization techniques, with many recent methods employing semi-definite programming (Weinberger and Saul 2006).

In (Bowling et al. 2007), the authors adapt a method of semi-definite programming along with constraints inferred from motion properties of a mobile robot to perform subjective localization. This method, called action respecting embedding, uses minimally interpreted action labels to identify low dimensional descriptions of the path of a roving eye robot.

In general, efforts to employ semi-definite programming methods to our datasets resulted in unfeasible computations, though we were able to use more recent approaches that address the scalability issues associated with the high cost of performing semi-definite programming (Weinberger et al. 2007). We hope to extend *sensorimotor embedding* to localization tasks, replacing the discrete action labels used by action respecting embedding with policies over continuous action spaces.

Conclusions

We presented the results of comparing several manifold learning methods for the problem of learning sensor maps from robot sensor traces. These results show that no single algorithm dominates all others across all of our test sets. This complicates the choice of manifold learning method for inclusion in a developmental architecture, since any choice of method may result in poor sensor maps on some platforms.

Any developmental system that is expected to learn to control a variety of mobile robot platforms will have to utilize a wide variety of manifold learning methods. A heterogeneous developmental system, with access to many different algorithms, may be able to select the appropriate construction method based on certain properties of the sensors, such as boundary size. We gave one such method for computing boundary size based on neighborhood network flow.

Moreover, we examined the effect of policy on sensor reconstruction, and found that motor babbling, in addition to being a developmentally feasible policy, actually results in better reconstructions than deterministic policies.

Finally, we considered a new approach to learning sensor maps, *sensorimotor embedding*, that uses the geometry of controls to infer the geometry of sensor elements. The general approach of utilizing action spaces to better understand sensor spaces follows the "seeing is acting" paradigm of O'Regan and Noë (2001) . This method requires the specification of a reward signal, but unlike other approaches, produces sensor maps with meaningful scales that are robust to physical sensor changes over time.

Acknowledgments

This work has taken place in the Intelligent Robotics Lab at the Artificial Intelligence Laboratory, The University of Texas at Austin. Research of the Intelligent Robotics lab is supported in part by grants from the Texas Advanced Research Program (3658-0170-2007), and from the National Science Foundation (IIS-0413257, IIS-0713150, and IIS-0750011).

References

Belkin, M., and Niyogi, P. 2003. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation* 15(6):1373–1396.

Bowling, M.; Wilkinson, D.; Ghodsi, A.; and Milstein, A. 2007. Subjective localization with action respecting embedding. *Robotics Research* 190–202.

Choi, H., and Choi, S. 2007. Robust kernel isomap. *Pattern Recognition* 40(3):853–862.

Demiris, Y., and Dearden, A. 2005. From motor babbling to hierarchical learning by imitation: a robot developmental pathway. In *Proceedings of the 5th International Workshop on Epigenetic Robotics*, 31–37.

Donoho, D. L., and Grimes, C. 2003. Hessian eigenmaps: Locally linear embedding, techniques for high-dimensional data. *Proceedings of the National Acadademy of Sciences* 100(10):5591–5596.

Eren, T.; Goldenberg, D.; Whiteley, W.; Yang, Y.; Morse, A.; Anderson, B.; and Belhumeur, P. 2004. Rigidity, computation, and randomization in network localization. *Twenty-third Annual Joint Conference of the IEEE Computer and Communication Societies (INFOCOM)* 4(7–11):2673–2684.

Kruskal, J., and Wish, M. 1978. *Multidimensional Scaling*. Sage Publications.

Meltzoff, A. N., and Moore, K. M. 1997. Explaining facial imitation: A theoretical model. *Early Development and Parenting* 6(3–4):179–192.

Olsson, L.; Nehaniv, C.; and Polani, D. 2006. From unknown sensors and actuators to actions grounded in sensorimotor perceptions. *Connection Science* 18(2):121–144.

O'Regan, J., and Noë, A. 2001. A sensorimotor account of vision and visual consciousness. *Behavioral and Brain Sciences* 24(05):939–973.

Philipona, D.; O'Regan, J.; Nadal, J.; and Coenen, O. 2004. Perception of the structure of the physical world using unknown multimodal sensors and effectors. *Advances in Neural Information Processing Systems* 16:945–952.

Philipona, D.; O'Regan, J.; and Nadal, J. 2003. Is There Something Out There? Inferring Space from Sensorimotor Dependencies. *Neural Computation* 15(9):2029–2049.

Pierce, D., and Kuipers, B. 1997. Map learning with uninterpreted sensors and effectors. *Artificial Intelligence* 92(1-2):169–227.

Priyantha, N.; Balakrishnan, H.; Demaine, E.; and Teller, S. 2003. Anchor-free distributed localization in sensor networks. *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems* 340–341.

Roweis, S., and Saul, L. 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290(5500):2323–2326.

Seber, G. 1984. *Multivariate observations*. Wiley.

Singer, A. 2008. A remark on global positioning from local distances. *Proceedings of the National Academy of Sciences* 105(28):9507.

Stronger, D., and Stone, P. 2006. Towards autonomous sensor and actuator model induction on a mobile robot. *Connection Science* 18(2):97–119.

Tenenbaum, J.; Silva, V.; and Langford, J. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290(5500):2319–2323.

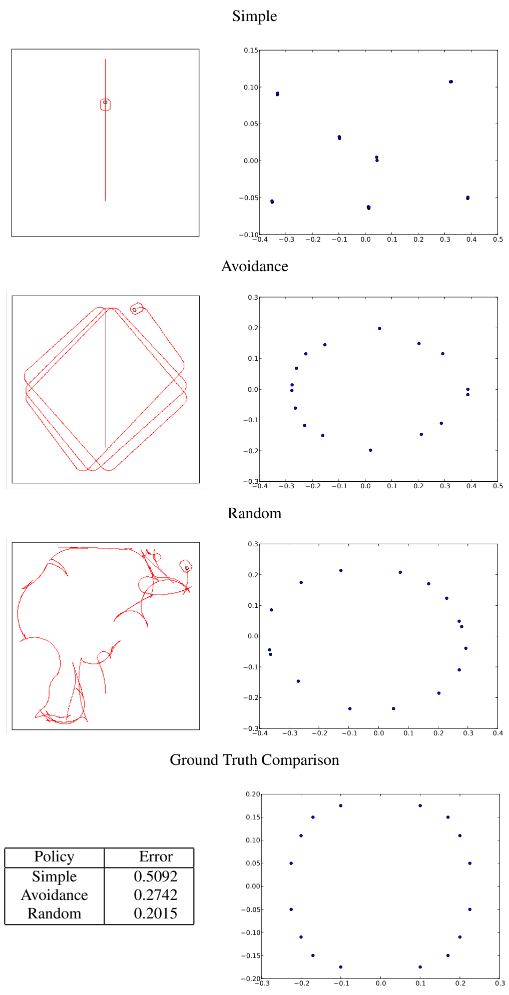
van der Maaten, L. 2007. An Introduction to Dimensionality Reduction Using Matlab. Technical Report MICC 07-07, Maastricht University.

Wang, C., and Mahadevan, S. 2008. Manifold alignment using Procrustes analysis. In *Proceedings of the 25th International Conference on Machine Learning*, 1120–1127.

Weinberger, K., and Saul, L. 2006. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *Proceedings of the Twenty First National Conference on Artificial Intelligence*.

Weinberger, K.; Sha, F.; Zhu, Q.; and Saul, L. 2007. Graph laplacian regularization for largescale semidefinite programming. *Advances in Neural Information Processing Systems* 19:1489.

Yim, M.; Shen, W.-M.; Salemi, B.; Rus, D.; Moll, M.; Lipson, H.; Klavins, E.; and Chirikjian, G. S. 2007. Modular self-reconfigurable robot systems. *Robotics & Automation Magazine, IEEE* 14(1):43–52.



Policy	Error
Simple	0.5092
Avoidance	0.2742
Random	0.2015

Figure 5: As we vary the policy from complex to simple, the quality of the reconstruction of 16 simulated sonar sensors decreases. The robot path during each trace is shown on the left. The reconstruction is shown on the right. The last row shows the Procrustes distances for each policy as compared to the ground truth shown on the bottom right. We used classical MDS to generate all the reconstructions.

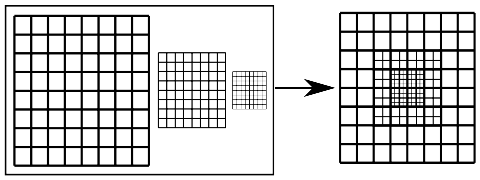


Figure 6: Our implementation of the fovea consists of overlapping layers of receptive fields. As the layer resolution increases, the extent of each receptive field decreases, and the number of bits necessary to describe the layer state remains constant.

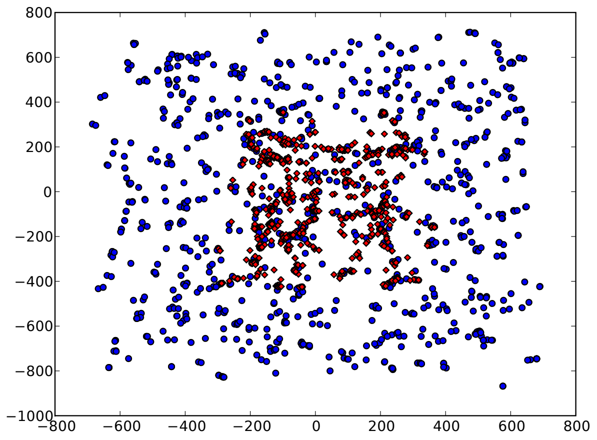


Figure 7: With sensorimotor embedding each sensor element learns a policy that centers local activation at the fovea resulting in greater post-saccade reward. The plot shows the corresponding action space coordinates of each sensor element in a foveated retina. Though the retina was simulated, the underlying images and saccades were generated using a pan/tilt camera. Note that the density of sensor elements is higher near the fovea.

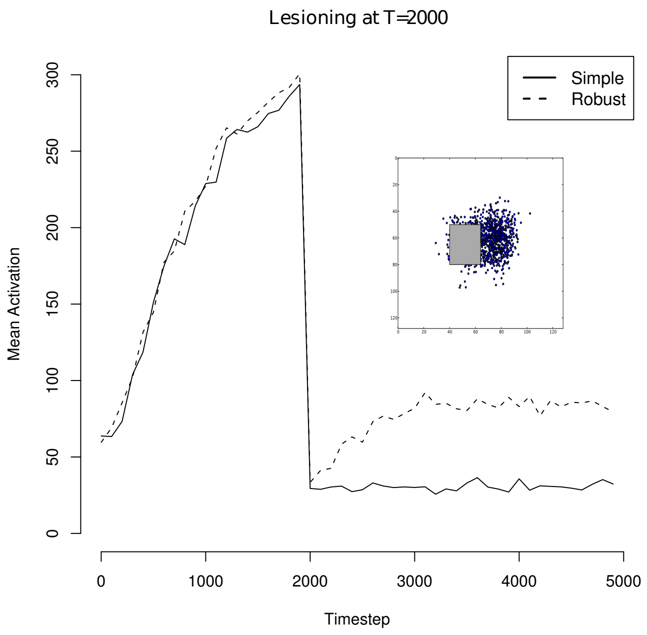


Figure 8: As a result of lesioning, a retina adapts its policy to favor saccades to regions just outside the damaged region (see subfigure), providing higher post-saccadic activation in the case of lesioning than the previous optimal saccades directly to the fovea. We note that this increases the position error relative to the ground truth, but provides a coordinate system consistent with the sensorimotor properties of the damaged retina.