

# Self-Organizing Geometric Certainty Maps: A Compact and Multifunctional Approach to Map Building, Place Recognition and Motion Planning

J.A. Janét, S.M. Scoggins, M.W. White, J. C. Sutton, III, E. Grant and W. E. Snyder

Department of Electrical and Computer Engineering

North Carolina State University

Raleigh, NC 27695-7911, USA

E-MAIL: jajanet@eos.ncsu.edu

## Abstract

*In this paper we show how a self-organizing Kohonen neural network can use hyperellipsoid clustering (HEC) to build maps from actual sonar data. Because the HEC algorithm uses the Mahalanobis distance, the elongated shapes (typical of sonar data) can be learned. The Mahalanobis distance metric also gives a stochastic measurement of a data point's association with a node. Hence, the HEC Kohonen can be used to build topographical maps and to recognize its own topographical cues for self-localization. The number of nodes can also be regulated in a self-organizing manner by using the Kolmogorov-Smirnov (KS) test for cluster compactness. The KS test determines whether a node should be divided (mitosis) or pruned completely. Because fewer nodes are needed for an HEC Kohonen than for a Kohonen that uses only Euclidean distance, the data size is smaller for the HEC Kohonen. Relative to grid-based approaches, the savings in data size is even more profound. By incorporating principal component analysis (PCA), the HEC Kohonen can handle problems with several dimensions (3-D, time-series, etc.). The HEC Kohonen is also multifunctional in that it accommodates compact geometric motion planning and self-referencing algorithms. It can also be used to solve a host of other pattern recognition problems.*

## 1 Introduction

It is proposed that an ideal map building algorithm has the following characteristics: 1) Self-organizing; 2) Multifunctional (within, and outside of, the environment representation problem); 3) Small data requirements; 4) Low computational complexity; 5) Updatability; and 6) Unlimited dimensions ( $R^2$ ,  $R^3$ , time-series, etc.). Few, if any, current models satisfy all of these requirements simultaneously. In this paper we explore a model that incorporates the self-organizing characteristics of the Kohonen neural network [10, 11], the stochastic metric

of hyper-ellipsoid clustering (HEC) [2, 12, 14, 15] and the compactness of geometric environment representation [6].

### 1.1 Geometric vs Cellular Representation

Grid-based certainty maps are widely used to store and maintain occupancy information because they are easy to build and maintain [17]. However, certainty grids are known to have large data requirements. In fact, data requirements for certainty grids of global maps and/or  $R^3$  environments can still be prohibitive [4, 16]. Certainty grids are also subjective since resolution and maximum region dimensions must be determined *a priori*. Finally, it is difficult to assess object surface information from a cellular model.

Geometric representations, on the other hand, have been difficult to build, particularly on the basis of a stochastic occupancy. But geometric representations are significantly more compact, less complex and fully compatible with high- and low-level motion planning and self-referencing approaches [4, 6]. With higher dimensions, the geometric model data requirements become exponentially smaller than the cellular model data requirements.

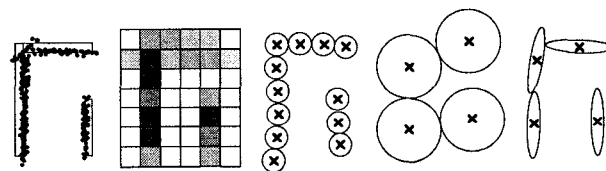


Figure 1: From left to right: (a) Sonar data, (b) certainty grid, (c) high-resolution Kohonen node configuration using Euclidean cost function, (d) low-resolution Kohonen node configuration using Euclidean cost function, and (e) Kohonen node configuration using Mahalanobis distance cost function.

## 1.2 Significance of the Stochastic Metric

Stochastic models are important utilities in decision theoretic problems like motion planning, topographical mapping, place recognition and sensor estimation [9, 17]. They provide a probabilistic value that can depend on the frequency of occurrence, position, orientation, *a priori* and *a posteriori* predictions, etc. The HEC algorithm recently introduced by Mao and Jain [12] utilizes competitive learning and the Mahalanobis distance metric to stochastically model shape information of learned clusters (like mapped sonar data). Unlike the Euclidean distance metric, the Mahalanobis distance can estimate shapes of elongated data. The Mahalanobis contours can also be used to represent varying certainties of object presence and/or shape (for motion planning, sensor prediction and place recognition).

## 1.3 Self-Organization and Updatability

To be “autonomous”, a mobile robot must be capable of organizing its sensor- and time-based data. It must be capable of deciding where it is inside the environment it modeled, as well as, when and how an update should occur. Perhaps the best example of a vector-based self-organizing map is the Kohonen neural network [11]. The Kohonen neural network allows local updating. It also permits nodes (which can represent topographical cues) to be freely added and/or removed. The HEC utilizes the Kolmogorov-Smirnov (KS) test to determine the *compactness* of a data cluster [13], and decide if a node should be divided (mitosis) to better model what might be two different clusters. The KS test can also be used to determine if a node should be pruned.

## 1.4 Unlimited Dimensions

An ideal model should accommodate an  $R^3$  mapping of objects without adversely impacting the data size or complexity. It should also accommodate the time-dimension since it can be used for both the time-series mapping and the exponential decay of an object’s position certainty [3]. Finally, it should be flexible enough to allow data from a host of sensors (each type having its own dimension(s)) so that it can be easily integrated. The HEC employs principal component analysis (PCA) to estimate the hyperellipsoidal shapes of clusters. The PCA learning algorithm is an iterative model that computes the cluster eigenvectors while ensuring orthonormality for architectures of any dimensionality. We found the PCA model in [2, 14] faster and more reliable than Mao and Jain’s PCA model [12].

## 1.5 Data Size & Computational Complexity

When queried about the obvious voracious consumption of RAM, the common response from researchers that

use grid-based approaches is, “Memory is cheap, CPU’s are fast, smile.” But the truth is, mobile robots can have much more to do than map-building, motion planning and self-localization. There is still ample room for improvement in these tasks alone. So, in general, there is never an excuse for *not* looking for smaller data requirements and lower algorithmic- and/or time-complexities, provided functionality is preserved.

## 1.6 Multifunctionality

A general solution is typically the better solution. That is, a model that can be used to solve a variety of problems is typically worth more than dedicated models. This is especially true for platforms like mobile robots since power, memory and computational capabilities are limited. Part of the reason neural networks have become so popular is that they use a single math model to solve a wide variety of problems. For the environment representation problem, we want a compact model that allows for efficient sensor-based map-building, updatability, motion planning, self-referencing, etc. Outside the environment representation problem, we want a similar model for other pattern recognition or control problems, reasoning and maybe even creative processes.

It was shown in [6] that traversability vectors (t-vectors) are a simple, multifunctional math model that detect collisions between paths and polygons, collect geometric beacons, identify Euclidean optimal paths around polygons and identify which features of a geometric beacon are visible to sensors. T-vectors work in static and dynamic environments, and they have been shown to significantly reduce the data size and complexity of roadmap networks. It will be shown that the HEC Kohonen neural network complements the t-vector model (and any other geometric- or topographical-based motion planner or self-referencer) model by generating polygonal obstacles that conform to probability contours manifested by the Mahalanobis distance.

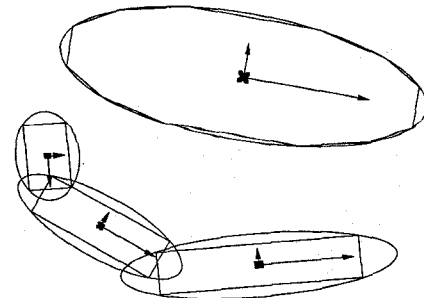


Figure 2: HEC Kohonen nodes with path polygons fit to 80% confidence level contour.

## 2 The HEC Kohonen Neural Net

A HEC Kohonen neural network has been implemented on actual sonar data. This section highlights the HEC learning algorithm as we implemented it in a Kohonen neural network. Some of our notation is taken from the work by [2, 12, 15, 14]. Given a set of  $n$  sonar data points in a  $d$ -dimensional space,  $\{\bar{x}_i | \bar{x}_i \in R^d, i = 1, 2, \dots, n\}$ , we assign  $\bar{x}_i$  to one of the  $K$  network nodes,  $\{M_{ik} | M_{ik} \in \{0, 1\}; i = 1, 2, \dots, n; k = 1, 2, \dots, K\}$ , where  $M_{ik} = 1$  if  $\bar{x}_i$  is assigned to the  $k^{th}$  node, and  $M_{ik} = 0$  otherwise.

### 2.1 The Regularized Mahalanobis Distance

The metric used for vector quantization is referred to as the regularized Mahalanobis distance [12].

$$D_{RM}(\bar{x}_i, \bar{m}_k) = (\bar{x}_i - \bar{m}_k)^T \bar{C}_k^{-1*} (\bar{x}_i - \bar{m}_k) \quad (1)$$

where  $\bar{m}_k$  is the cluster centroid and  $\bar{C}_k^{-1*}$  is referred to as the regularized inverse of the covariance matrix  $\bar{C}_k$  for the  $k^{th}$  node cluster.

$$\bar{C}_k^{-1*} = (1 - \rho) (\bar{C}_k + \varepsilon I)^{-1} + \rho I \quad (2)$$

The identity matrix  $I$  and covariance matrix  $\bar{C}_k$  are both  $d \times d$ . The parameters,  $\rho$  and  $\varepsilon$  are used to stabilize the training process:  $\rho$  is used to weight the Euclidean distance and Mahalanobis distance in  $D_{RM}$ ;  $\varepsilon$  is used to prevent singularities in  $\bar{C}_k$ . When  $\rho = 0$  and  $\varepsilon = 0$ ,  $D_{RM}$  becomes the squared Mahalanobis distance. When  $\rho = 1$ ,  $D_{RM}$  becomes the squared Euclidean distance. When  $0 < \rho < 1$   $D_{RM}$  is a linear combination of the squared Mahalanobis distance and Euclidean distance.

The Euclidean distance,  $D_E$ , is included in (1) because it is a reliable metric for early stages of self-organized training. It can remain a significant part of the recall metric if one so desires. But in the early stages of training,  $D_E$  is used to seat nodes over areas of most activation. Regardless of the value of  $\rho$ , the Mahalanobis distance,  $D_{M\varepsilon}$ , is computed each epoch. Hence, even if vector quantization relied solely on  $D_E$ , the stochastic information contained in  $D_{M\varepsilon}$  is still available.

Ideally, though,  $D_{RM}$  becomes more dependent on  $D_{M\varepsilon}$  as training continues. This is because the purely Euclidean distance metric tends to split elongated clusters. On the other hand, immediately relying on the Mahalanobis metric can cause one node to absorb several smaller clusters. This leads to unusually large (or unusually small) clusters. The regularized Mahalanobis distance attempts to strike a balance between these two properties and, thus, thwart the production of unusually large or unusually small clusters.

### 2.2 The Whitening Transform

To overcome the computational issues involved with computing  $\bar{C}_k^{-1*}$ , the Mahalanobis distance between  $\bar{x}_i$  and  $\bar{m}_k$  is "whitened" with the transformation,

$$D_{M\varepsilon}(\bar{x}_i, \bar{m}_k) = D_E(\bar{y}_{ik}, \bar{v}_k) \quad (3)$$

where

$$\bar{y}_{ik} = \Lambda_k^{-1/2} \Phi_k \bar{x}_i \quad (4)$$

$$\bar{v}_k = \Lambda_k^{-1/2} \Phi_k \bar{m}_k \quad (5)$$

Eigenvalues  $\lambda_{kj}, j = 1, 2, \dots, d$  of the buffered covariance matrix ( $\bar{C}_k$ ) are stored in  $\Lambda_k = \text{diag}\{\lambda_{k1}, \lambda_{k2}, \dots, \lambda_{kd}\} + \varepsilon I$ . The eigenvectors  $\bar{w}_{jk}$  are stored in the eigenvector matrix  $\Phi_k = \{\bar{w}_{1k}, \bar{w}_{2k}, \dots, \bar{w}_{dk}\}^T$ . The  $d$  rows of  $\Phi_k$  are eigenvectors corresponding to the  $d$  eigenvalues in  $\Lambda_k$ .

The whitening transform simply projects  $\bar{x}_i$  onto the axes of the ellipse and scales the projection by  $\Lambda_k$  to make  $D_{M\varepsilon}$  common to the purely Euclidean distance  $D_E$ . By projecting  $\bar{x}_i$  onto the whitened space and computing the Euclidean distance in the whitened space, the regularized distance becomes a weighted sum of two Euclidean distances: one in the original input space (weighted by  $\rho$ ) and the other in the whitened space (weighted by  $1 - \rho$ ). Each node cluster has its own whitening transform that depends on the points  $\bar{x}_i$  belonging to it.

### 2.3 Competitive Learning

So now the total signal intensity for the  $k^{th}$  node is measured by

$$D_k = (1 - \rho) \sum_{j=1}^d (v_{jk} - y_{jk})^2 + \rho \sum_{j=1}^d (m_{jk} - x_{jk})^2 \quad (6)$$

The node with the smallest  $D_{k_{min}}$  is considered the *winning* node  $z_k$  for the data point  $\bar{x}_i$ . In our HEC Kohonen software, the total error for the winning node,  $z_k$  is increased by  $D_{k_{min}}$ . Also, the data point  $\bar{x}_i$  is added to a list of data points  $\bar{X}_k$  belonging to  $z_k$ . The point list  $\bar{X}_k$  has four functions: 1) It is used in the PCA learning mode to compute  $\Phi_k$  and  $\Lambda_k$ ; 2) It is used in the Kolmogorov-Smirnov test for compactness; 3) It is used to locally train nodes that are split to better fit non-compact clusters; and 4) A subset of  $\bar{X}_k$  can be saved with the node weights so that subsequent update learning will be biased by the previously learned points.

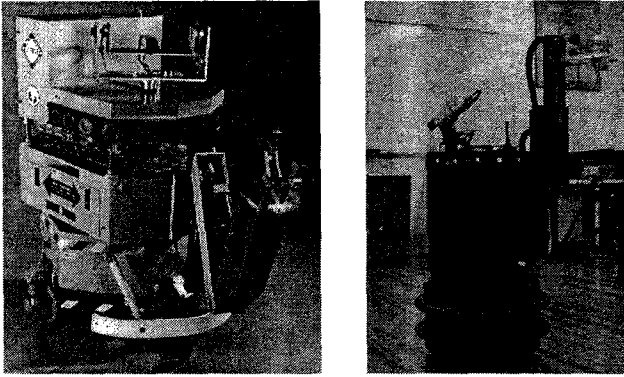


Figure 3: Cybermotion K2A and Nomad 200.

### 3 Map Building with the HEC Kohonen

Mentioned in [5, 8] we used two differently configured mobile robots (shown in Figure 3) to autonomously explore and learn ten regions of our building (using a different kind of neural network). Each robot explored the ten rooms six different times, collecting and mapping sonar data. Figure 4 shows an example data set.

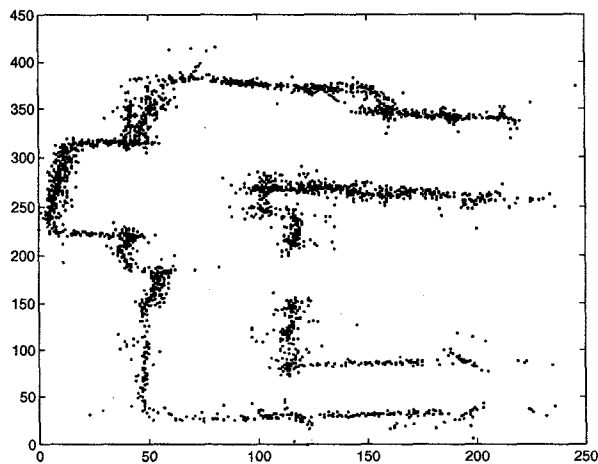


Figure 4: Actual sonar data collected by Lola.

#### 3.1 Architecture and Parameters

As an example application, we use the HEC Kohonen neural network to learn the mapped sonar data shown in Figure 4. There are 3000 data points in Figure 4. In previous work using a Kohonen neural network with the Euclidean distance metric, we needed 440 nodes to accurately model the shape of a room [7]. With the HEC Kohonen, we were able to reduce the number of nodes to less than 26. Before training, the nodes were initialized to coordinates uniformly distributed over the area bounded by the minimum and maximum  $x$ - and  $y$ -locations of the collected sonar data. Our regularization

parameter was initialized to  $\rho(0) = 1$  and set to decrease at a rate of  $\Delta\rho = 0.05$  until it reached a minimum value of  $\rho_{min} = 0.0$ .

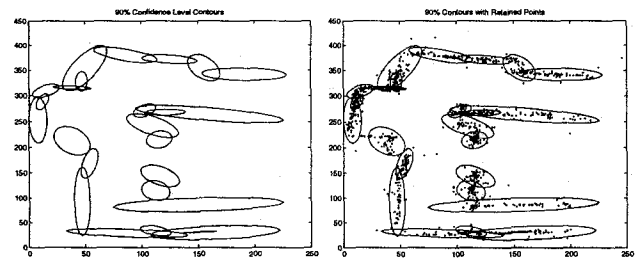


Figure 5: 25 HEC clusters and retained data points after 100 epochs of training.

To reduce the number of dormant nodes (nodes that are never selected as a winner), each node was initially given four neighbors to update at a fraction of the distance of the winning node. Neighborhood models are excellent for pulling dormant nodes towards areas of activity. The neighborhood sizes were linearly reduced to zero neighbors after the 16<sup>th</sup> epoch.

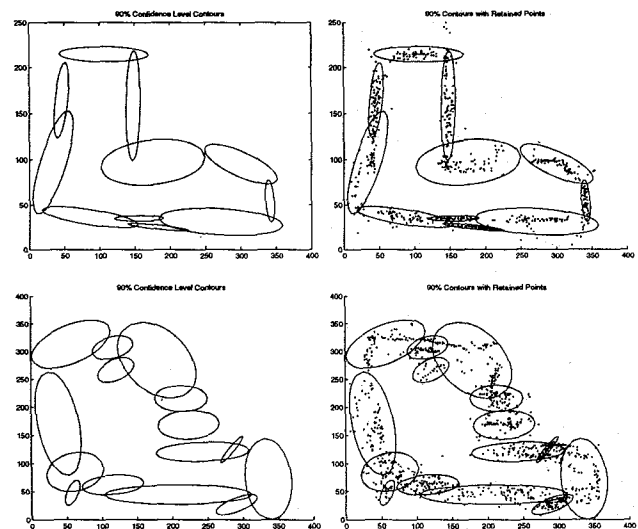


Figure 6: HEC cluster sets for two other topographical locations.

After training for 100 epochs (3 minutes, 25 seconds), the HEC Kohonen modeled the data from Figure 4 as shown in Figure 5. The data points in Figure 5 are subsamples that, as mentioned in Section 2.3, are saved to the weights file to retard any radical node changes in subsequent updating. Figure 6 shows some other topographical locations mapped by the HEC Kohonen neural network.

### 3.2 Mitosis: Node Division

In the context of this research, mitosis is the division of one node  $k$  into two daughter nodes  $k_1$  and  $k_2$ . Mitosis is essential because, after some training, there can be nodes that try to single-handedly model two distinctly different clusters. The criterion used to detect this phenomena comes from the Kolmogorov-Smirnov (KS) test [13]. For a Gaussian distribution, the significance level of the KS test on transformed patterns relative to their centroids can be used to measure the compactness of a cluster with respect to its theoretical  $\chi^2$  distribution.

If the compactness of a node's cluster is found to be less than a predefined threshold, the node is divided into two daughter nodes  $k_1$  and  $k_2$ . The daughter node centroids are initialized to the locations of the major axis foci from the parent node. Using the associated data list stored in  $\{\bar{x}_i | \bar{x}_i \in \bar{X}_k\}$ , the daughter nodes begin the training process all over again until their maturation levels match older nodes in the HEC Kohonen network.

### 3.3 Pruning Dormant Nodes

In [7] we explored the impact of dormant node pruning for Kohonen neural networks using the Euclidean distance metric. Simply put, Kohonen nodes are expected to represent part or all of a unique feature in the data space. If a node exists that does not reflect a feature, it is *pruned*. Pruning a Kohonen node renders that node unavailable as a winning node. This penalizes the Kohonen since  $\bar{x}_i$  must then resort to a node that is farther away than the truly closer pruned node. The result is a larger  $z_k$  for the improperly matched Kohonen-data set pair. Since errors are emphasized more the probability of misclassification was shown to decrease. But it was also shown in [7] that a more aggressive pruning approach (dormant *and* low-frequency nodes) resulted in an *increase* in misclassifications. So we decided to only prune those nodes that were not chosen as winning nodes after the neighborhood size was reduced to zero and during the later stages of training.

### 3.4 Place Recognition with Hill Climbing

Topographical maps have become very popular to the mobile robotics field because they can integrate stochastic, sensor and dead-reckoning information, are compact due to their roadmap configurations, and, most importantly, they are intuitive. Time and again, mobile robots have shown that they can build topographical maps and later use these maps for decision-theoretic motion planning and self-localization [17]. But most topographical maps are based on RAM-hungry cellular environment representations. The HEC Kohonen, on the other hand, provides the same stochastic information at a fraction of the data size. It also allows for updatability.

In [7] we examined two different neural network approaches to solve the global self-localization (GSL) problem. The primary condition was that the mobile robot did not know which region of space it was in, nor how it arrived there. A more precise approach to estimating position and orientation was demonstrated by Yamauchi with ELDEN (Exploration and Learning in Dynamic ENvironments) [17]. Specifically, ELDEN used a hill-climbing algorithm to match evidence grids by searching the space of possible translations and rotations. With great success, ELDEN was able to correct a robot's position estimate and cope with transient and lasting changes.

With the HEC Kohonen, we can use the same hill-climbing algorithm to perform the same task of place recognition. (Initial step sizes of 1.0 foot and 1.0° rotation, halved when a local maximum is reached, until a local maximum is found at a step size of 0.125 feet and 0.125°.) Future work will compare computational complexities and general success of both models. We speculate, however, that the computational complexity is less for the HEC Kohonen because there are fewer elements to examine than ELDEN's cell-based model. A comparison is forthcoming.

### 3.5 Motion Planning and Self-Referencing

Using the two conventions of polygon estimation shown in Figure 2, we start at a 90% confidence level contour and approximate a polygon centered at node  $k$  in Euclidean space, and oriented along the major and minor axes of the cluster's eigenvector matrix  $\Phi_k$ . These polygons can represent the C-space *path polygons*. We then use t-vectors (another compact and multi-functional model) to construct the Essential Visibility Graph (EVG) for planning motion on a global level [6]. If a complete plan of motion is not realized, we gradually decrease the confidence level contours, thereby shrinking the path polygon sizes [1]. This process is continued until a complete plan of motion is realized. T-vectors can also collect geometric beacons for the sensor prediction phase of self-localization. Of course, since the environment is geometrically represented, *any* other geometric motion planner or self-referencer can also be used.

## 4 Concluding Remarks and Future Work

In this paper we showed how a self-organizing HEC Kohonen neural network holds great promise for building topographical maps from actual sensor data. The HEC Kohonen neural network is significantly smaller than a purely Euclidean distance-based Kohonen neural network because the HEC algorithm uses the Mahalanobis distance to learn the elongated shapes typical of

sonar data. Inherently, the Mahalanobis distance metric yields a stochastic measurement of a data point's association with a node. Hence, the HEC Kohonen can be used to build topographical maps and to recognize its own topographical cues for self-localization. The number of nodes can also be regulated by the KS test for cluster compactness. The KS test determines whether a node should be divided (mitosis). Pruning can also be based on the KS test. However, in this paper we opted to only prune dormant nodes.

Relative to grid-based approaches, the savings in data size is quite profound. And the size difference increases with higher dimensional problems. By incorporating principal component analysis (PCA), the HEC Kohonen can handle problems with several dimensions (3-D, time-series, etc.). The HEC Kohonen is also multifunctional in that it accommodates compact geometric motion planning and self-referencing algorithms like t-vectors. It can also be used to solve a host of other pattern recognition problems, making it a truly multifunctional tool.

We are currently analyzing the computational complexities of various algorithms including the training time for the HEC Kohonen, place recognition using hill-climbing, etc. to determine if the benefits to using the HEC Kohonen extend beyond self-organization, smaller data size, greater multifunctionality and flexibility.

## References

- [1] B. Baginski, "Local Motion Planning for Manipulators Based on Shrinking and Growing Models", *IEEE Int'l Conf on Robotics and Automation*, Minneapolis, MN, April, 1996.
- [2] N. Cliff, *Analyzing Multivariate Data*, Harcourt Brace Jovanovich, Publishers, San Diego, CA, pp. 293-318, 1987.
- [3] A. Elfes, "Using Occupancy Grids for Mobile Robot Perception and Navigation", *IEEE Computer*, June, 1989, pp. 46-57.
- [4] J. A. Horst, "Maintaining Multi-level Planar Maps in Intelligent Systems", *IEEE Int'l Conf on Robotics and Automation*, Minneapolis, MN, April, 1996.
- [5] J. A. Janét, D. S. Schudel, M. White, A. G. England, R. C. Luo, W. E. Snyder, "Global Self-Localization for Actual Mobile Robots: Generating and Sharing Topographical Knowledge Using the Region-Feature Neural Network", *IEEE Int'l Conf on Multisensor Fusion and Integration*, December, 1996.
- [6] J. A. Janét, R. C. Luo and M. G. Kay, "Autonomous Mobile Robot Global Motion Planning and Geometric Beacon Collection Using Traversability Vectors." *IEEE Trans. on Robotics and Automation*, Vol. 13(1), February, 1997, pp.132-140.
- [7] J. A. Janét, R. Gutierrez, T. A. Chase, M. White, J. C. Sutton, III, "Autonomous Mobile Robot Global Self-Localization Using Kohonen and Region-Feature Neural Networks", *Journal of Robotic Systems: Special Issue on Mobile Robots*. April, 1997.
- [8] J.A. Janét, D.S. Schudel, M.W. White, A.G. England, J.C. Sutton, III, E. Grant and W.E. Snyder, "Two Mobile Robots Sharing Topographical Knowledge Generated by the Region-Feature Neural Network", *IEEE Int'l Conf on Robotics and Automation*, Albuquerque, NM, April, 1997.
- [9] L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe, "Analysis of Probabilistic Roadmaps for Path Planning", *IEEE Int'l Conf on Robotics and Automation*, Minneapolis, MN, April, 1996.
- [10] T. Kohonen, "Self-Organized Formation of Topologically Correct Feature Maps." *Biol. Cybernetics*, vol 43, pp. 59-69, 1982.
- [11] T. Kohonen, "Self-Organizing Maps: Optimization Approach." *Artificial Neural Networks*, T. Kohonen, K. Makisara, O. Simula and J. Kangas, Eds. New York: Elsevier Science, 1991, pp. 183-186.
- [12] J. Mao and A. K. Jain, "A Self-Organizing Network for Hyperellipsoidal Clustering (HEC)." *IEEE Trans. on Neural Networks*, vol 7, no 1, pp 18-29, January, 1996.
- [13] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. Cambridge University Press, Cambridge, UK, pp. 620-648, 1988.
- [14] J. G. Reich, *C Curve Fitting and Modeling for Scientists and Engineers*, McGraw-Hill, Inc, New York, NY, pp. 27-156, 1992.
- [15] J. Rubner and K. Schulten, "Development of Feature Detectors by Self-Organization." *Biol. Cybernetics*, vol 62, pp. 193-199, 1990.
- [16] J. Vandorpe, H. Van Brussel, and H. Xu, "Exact Dynamic Map Building for a Mobile Robot using Geometrical Primitives Produced by a 2D Range Finder", *IEEE Int'l Conf on Robotics and Automation*, Minneapolis, MN, April, 1996.
- [17] B. Yamauchi, "Mobile Robot Localization in Dynamic Environments Using Dead Reckoning and Evidence Grids", *IEEE Int'l Conf on Robotics and Automation*, Minneapolis, MN, April, 1996.