Global Self-Localization for Autonomous Mobile Robots Using Self-Organizing Kohonen Neural Networks

Jason A. Janét, Ricardo Gutierrez-Osuna, Troy A. Chase, Mark White and Ren C. Luo
Center for Robotics and Intelligent Machines
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC 27695-7911

ABSTRACT - An approach to global self-localization for autonomous mobile robots has been developed using self-organizing Kohonen neural networks. This approach categorizes discrete regions of space using mapped sonar data corrupted by noise of varied sources and ranges. Our approach is similar to optical character recognition (OCR) in that the mapped sonar data can, over time, assume the form of a character unique to that room. Hence, it is believed that an autonomous vehicle can be capable of determining which room it is in based on mapped sensory data ascertained by wandering through and exploring that room. With some pre-processing and a robust explore routine, the solution becomes time-, translation- and rotation-invariant.

1 Introduction

Can a robot determine which region of space it is in without knowing how it got there? To date a significant amount of work has been devoted to developing *low-level* self-localization approaches for autonomous mobile robots. These approaches depend on prior dead reckoning estimates and discrete-time models that iteratively rationalize and correct robot position and orientation based on correlations between predicted and actual sensor data [4, 5, 7, 11, 18-22, 27-29]. But, without an *initial* reliable position estimate, even the most reliable techniques can become ineffective.

The objective of this research is to endow autonomous mobile robots with the ability to perform self-localization on a *global* level. That is, the robot should be able to use sensor data to determine which region of indoor space it is in. See figure 1. Since most indoor environments can be easily segmented into *rooms*, different room sizes and configurations will define discrete regions of space. Hence, the global self-localizer is expected to identify features unique to a room and classify any sensor-based data set according to its level of similarity to memorized rooms.

1.1 Time-, Translation- and Rotation-Invariance

To be truly robust, a global self-localization (GSL) technique should have the following characteristics. First, it should be time-invariant for the simple reason that no two robots will explore a room the same way. In fact, a single robot will likely not follow the same trajectory each time. Second, it should be translation-invariant because the robot does not know the actual global coordinates of the region of space it is in, much less the sensor data it collects. Third, it should be rotation-invariant because, through the course of becoming lost, a robot can also become disoriented.

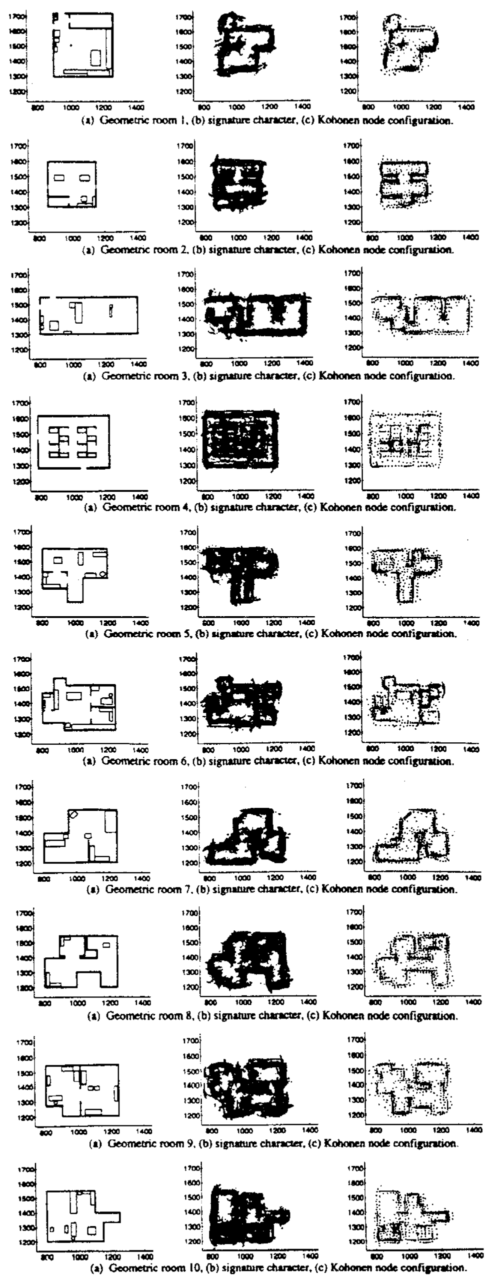

Figure 1. Geometric rooms, mapped sonar data and mature Kohonen node configurations.

0-8186-7108-4/95 $4.00 © 1995 IEEE

1.2 Self-Organizing Kohonen Neural Networks

Several applications of solving the optical character recognition (OCR) problem with neural networks have been investigated [10, 12, 13, 25, 26, 30, 33, 34]. In this application, a Kohonen neural network recognizes the *room character* associated with a particular room. The Kohonen neural net is known for its abilities to perform classification, recognition, data compression and association in an unsupervised manner [14, 30, 34]. That is, the Kohonen requires no *a priori* knowledge of a sensor data point's affiliation to a particular feature. All that must be estimated beforehand is the maximum expected number of features (neurons) per room. When a Kohonen is mature, the point density of the weight vectors approximate the probability distribution of the input space [17].

2 The Domain

We assume the environment to be in R^2 space. Specifically, as a robot wanders autonomously or is manually driven around a room, its sensor data is mapped to a two-dimensional plane. Since ultrasonic sensors seem to be the sensor of choice for autonomous and semi-autonomous vehicles, it is felt that the sensor used for extracting feature information should also be the ultrasonic sensor [1, 2, 6, 9, 23, 24, 31].

2.1 The Autonomous Mobile Robot Simulation

Given that the simulation described in [18-22] has proven accurate at modeling both sonar and robot behavior and that it provides a graphical display essential to understanding each step of solving this problem, it was considered a suitable platform for creating training sets and test sets. The simulated sonar models are based on work done by Kuc [23], Moravec [31] and Barshan [2] as well as information provided by Polaroid [32] and Cybermotion [8].

Also, the simulation can ensure that the 160 training sets and 180 test sets could be created in a timely fashion and without the risk of harming either the robot or the environment. Furthermore, the simulation can guarantee that no furniture is radically rearranged through the course of creating the training and test sets. Finally, the simulation can track both the robot's *dead reckoning* coordinates (i.e., where the robot *thinks* it is) and the robot's *actual* coordinates (i.e., where the robot *really* is) without needing someone to physically measure and/or estimate the true location of the real robot.

2.2 Room Character Generation

A room's unique character is created by clouds of sonar data points collected and mapped by the robot in its travels through the region. Typically, these clouds are clustered near the geometric beacon surfaces encountered by the sonar windows. With a single-transducer sonar it is difficult to know from a TOF reading the specific point that produced an echo because sonars sample a *region*. Hence, we assume that each sonar reading occurs along the axis normal to its transducer ($X_S = 0$, Y_S = TOF). To map the TOF reading, the point is transformed from the sonar frame to the global frame. TOF readings are mapped according to where the robot *thinks* it is (i.e., dead reckoning).

2.2.1 Sonar Corruption. Sonar readings are inherently corrupted by noise. So, "to represent the random errors in an adequate manner for a properly controlled experiment", a 6σ gaussian corruption function is applied to all TOF readings calculated by the simulation [16]. Specifically, for a *calculated* TOF distance, R_{calc}, C_s is a user-specified variable in the closed interval $C_s \in [0,1]$ that defines the range of noisy readings by R_{sens}. See figure 2.

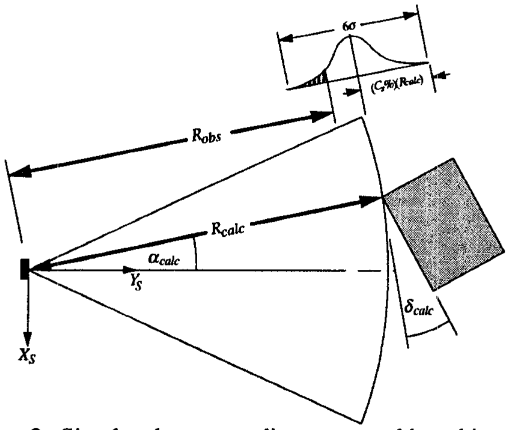

Figure 2. Simulated sonar reading corrupted by white Gaussian noise.

2.2.2 Trajectory Corruption. The very motivation for developing low-level sensor-based self-localization techniques stems from the fact that true mobile robot dead reckoning (from odometers, inertial navigation systems, etc.) grows more and more unreliable with each bump in the floor and turn of the robot. That is, where the robot *thinks* it is might not be where it *actually* is. To simulate this phenomenon, a user-defined random Gaussian trajectory corruption is induced as shown in figure 3.

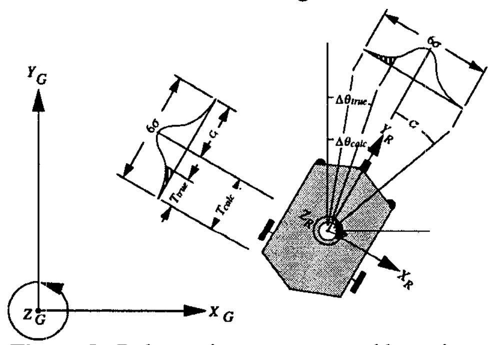

Figure 3. Robot trajectory corrupted by noise.

A 6σ gaussian corruption function is applied to all robot rotations and translations. Rotation corruption is limited to the closed interval $C_r^\circ \in [-\pi, \pi]$ and is specified by the user to define the range of potential noise added to each turn. Translation corruption is a variable on the closed interval $C_t \in [-1,1]$ and is also specified by the user to define the range of potential noise added to each translation. Figure 4 shows mapped sonar data from a simulated robot subjected to different degrees and types of corruption.

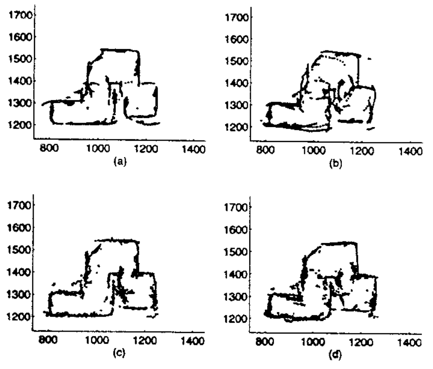

Figure 4. Room character a) $C_s = 0$, $\pm C_r^\circ = 0^\circ$, $C_t = 0$; b) $C_s = 0$, $\pm C_r^\circ = 18^\circ$, $C_t = 10\%$; c) $C_s = 15\%$, $\pm C_r^\circ = 0^\circ$, $C_t = 0$; d) $C_s = 15\%$, $\pm C_r^\circ = 18^\circ$, $C_t = 10\%$.

3 Strategy

A Kohonen neural network will organize a 21×21 mesh of cluster centers to represent the maximum potential 441 features each room is assumed to contain. While training, the Kohonen will make a record of the frequency with which each neuron fires to be used later in pruning redundant and/or deceptive cluster centers. To test the capabilities of the mature Kohonen, test data sets are collected by the simulated robot as it autonomously drives around the room. Generality will be tested by allowing test data corruption ranges to be larger than those used for training. Three pre-processing operations will be applied to the Kohonen to reduce the probability of misclassification.

3.1 Training and Testing Data

In a realistic setting, a robot can collect data while either being manually driven around the room (manual teaching) or autonomously driving through certain sections of the room while carrying out other tasks. Trajectories and explored features vary from data set to data set. However, one extremely important requirement of collecting training and testing data is that the robot explore (what it might perceive to be) the southwest extremes of each room. This aspect helps make the GSL problem translation-independent. That is, to transpose the mapped sonar data as close to the Kohonen origin as possible, we need mapped data that reflects minimum *x*- and minimum *y*-coordinate values (*west* and *south* respectively) a room might have.

3.2 Winner-Take-All Learning

The procedures followed in both training and recall of the Kohonen neural networks are fairly standard. Our Kohonen network consists of a 21×21 node network. The Euclidean distance from each data point $\overline{X}$ to all of the Kohonen nodes $\overline{W}_{ij}$ (for $i,j = 1,2,\ldots 21$) is calculated. A Kohonen node is determined to be the winner $\overline{W}_w$ if

$$\left\|\overline{X} - \overline{W}_w\right\| = \min_{i,j=1,2,\ldots 21}\left\{\left\|\overline{X} - \overline{W}_{ij}\right\|\right\}. \qquad (1)$$

$\overline{W}_w$ and its neighbors $\overline{W}_{N_w}$ are then updated based on the learning rate η_ε and the nieghborhood size N_ε for a given epoch ε. Our learning rate, initially η_o=0.5, is linearly reduced as the Kohonen matures during training to a value of η_f=0.01 at epoch $\varepsilon_{\max}$=200.

$$\eta_\varepsilon = \left(\frac{\eta_o - \eta_f}{\varepsilon_{\max}}\right)\varepsilon - \eta_o \qquad (2)$$

The update rule for the winning node is

$$\overline{W}_w^{k+1} = \overline{W}_w^k + \eta_\varepsilon\left[\overline{X} - \overline{W}_w^k\right] \qquad (3)$$

The neighbors of the winning node are also updated. Our learning rate for neighbors of the winning node is based on the index distance B from the winning node.

$$\eta_N = (\eta_\varepsilon)(0.7)^B \qquad (4)$$

The update rule for neighboring nodes is

$$\overline{W}_{N_w}^{k+1} = \overline{W}_{N_w}^k + \eta_N(\eta_\varepsilon, B)\left[\overline{X} - \overline{W}_{N_w}^k\right] \qquad (5)$$

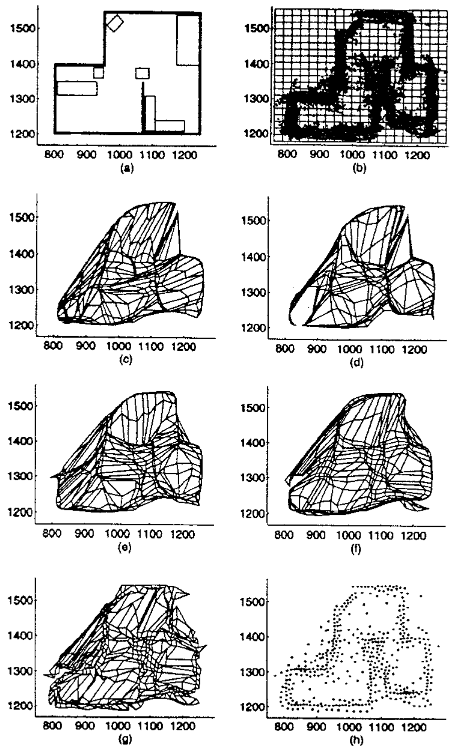

Figure 5. (a) Geometric map and (b) Node mesh at $\varepsilon = 0$, (c) $\varepsilon = 30$, (d) $\varepsilon = 50$, (e) $\varepsilon = 100$, (f) $\varepsilon = 150$ and (g) $\varepsilon_{\max}$. (h) mature Kohonen's cluster centers.

As the Kohonen trains, neighborhood size is gradually reduced. For the first 40% of our training the neighborhood size is a 5×5 window centered over $\overline{W}_w$. Then for the next 40% of training the neighborhood size is reduced to a 3×3 window During the final 20% of training only the winning node is updated. Stop training is based on convergence and $\varepsilon_{\max}=200$. See figure 5.

3.4 Recall

Testing the capabilities of this Kohonen-based global self-localization approach is a matter of presenting newly acquired room data to all the mature Kohonens. Recall is simply a summation Σ_{rk} of the Euclidean distances between each data point from the r^{th} data set and the nearest neuron from the k^{th} mature room Kohonen. The resulting Σ_{rk} is a measure of the match between the room data that has just been collected and the mature Kohonen that represents a particular room in memory. The Σ_{rk} measures for all the mature Kohonens are then compared to find the minimum. A Kohonen node mesh is determined to be the winner, Σ_{rw}, if it satisfies

$$\Sigma_{rw} = \min_{k=1,2,\ldots k_{\max}} \{\Sigma_{rk}\} \qquad (6)$$

The room w that corresponds to Σ_{rw} represents the room that the robot concludes it is currently in. Said differently, if the Kohonen belonging to room w yields the lowest summed Euclidean distance Σ_{rk} (for k = 1, 2,... $k_{\max}$) with test data set r, then the test data in question will be classified as belonging to room w.

3.5 Pre-Processing

Four pre-processing procedures are implemented in this research. First, *gross shifting* is used in all phases of testing to approximate a translationally-invariant problem space. Second, *fine shifting* (incrementally shifting the Kohonen and test data set about each other until a *best fit* is achieved) is used to reduce misclassifications. Third, we assess the value of pruning Kohonen neurons that are deemed redundant by not being activated in the latter stages of training. That is, we make unavailable to the test data points the neurons whose in-training firing frequency is zero. Fourth, we prune Kohonen nodes whose in-training firing frequency is less than 0.05% of number training data points.

3.5.1 Gross Shifting. The first step to translation-invariance involves gross shifting. This pre-processing procedure crudely attempts to fit a test data set arbitrarily placed in 2D space to a Kohonen node mesh, also arbitrarily placed in 2D space. As figure 6 shows, minimum x- and y-values of the data and Kohonen are subtracted from all data points, to transpose both to a common reference frame origin. While this translation takes care of the major portion of the translation-invariance problem, it can be seen that a single outlier in either the x- or y-direction can hinder a sound fit between Kohonen and data set. Hence, Σ_{rk} can unjustifiably increase and ultimately cause misclassification.

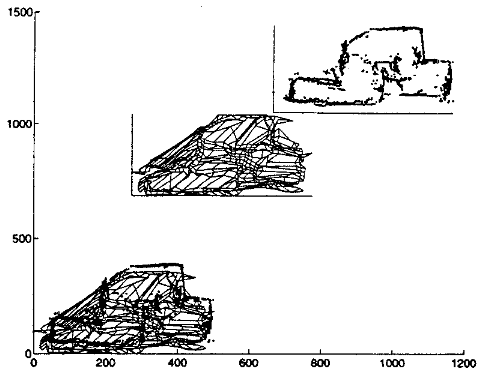

Figure 6. Gross shifting a Kohonen mesh (arbitrarily placed in 2D space) to a room character data set (also arbitrarily placed in 2D space).

3.5.2 Fine Shifting. We now need to resolve the less pronounced offset between the Kohonen and the room character test data set that resulted from gross shifting. With the minimum x- and y-values of the room data translated to the global origin, the Kohonen is then finely shifted about the data set to find the best fit. That is, we incrementally move each mature Kohonen mesh about the room data and search for the best fit with respect to Σ_{rk}. This systematic shifting is done by independently offsetting the Kohonen nodes' x- and y-values a distance in the range of $[-\rho,\rho]$. Figure 7 shows fine shifting until the best fit (d).

Time is the basic reason for 1) bounding the fine-shifting range, 2) moving the Kohonen about the data set in increments of 10 distance units (inches) and 3) sampling only 1% of the data points (out of ~2000 to ~4000 per test data set). Perhaps a genetic algorithm might do a better job of finding the optimal fit. But, *without* these constraints, recall is simply too time consuming to be practical.

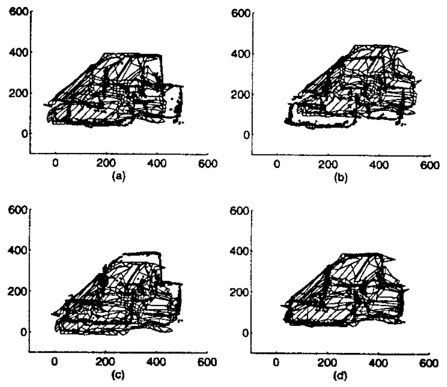

Figure 7 Fine-shifting a Kohonen mesh about a room character data set, $\rho = 1\,\mathrm{m}$.

Fine shifting attempts to refine the translation discrepancy caused by outliers in the data set or Kohonen. It also attempts to minimize the impact of room character skewedness and cloudiness. The range of potential offsets, $\pm\rho$, is sized on the basis of the following issues: the robot's confidence in its ability to aggressively and intelligently seek out and find the southwest extreme of a room; the anticipated maximum trajectory deviation; the anticipated maximum sensor corruption; and the expected worst-case room character alteration caused by rearranged furniture.

3.5.3 Pruning. Pruning a node of a Kohonen renders that node unavailable to a data point for use as a *nearest neuron.* Simply put, Kohonen nodes are assumed to represent part or all of unique feature in a particular room. If a node exists that does *not* reflect a feature, it is pruned. In recall, the Kohonen is penalized since the data point must then resort to a node that is farther away (in a Euclidean sense) than the truly closer pruned node. The result is a larger summed Euclidean distance Σ_{rk} for the *improperly* matched Kohonen-data set pair. *Properly* matched Kohonen-data set pairs, on the other hand, are believed to result in slight decreases in Σ_{rk}. Given that Σ_{rk} is the metric with which a data set is associated with a particular room and that errors are emphasized more than matched features by larger Σ_{rk}, the probability of misclassification should decrease.

Two levels of pruning mature Kohonen networks were tested to see if it reduces misclassification errors. First the mature Kohonen room networks are pruned of all nodes whose frequency of being chosen the *winning neuron*, $f_{w_{ij}}$ (for i,j = 1, 2,...21), is zero in the last epoch of training. Figure 8b marks with an 'x' all nodes that satisfy the condition $f_{w_{ij}} = 0$ for the first stage of pruning. The second pruning stage is more aggressive in that we prune all nodes whose frequency of being chosen the *nearest* or *winning neuron*, $f_{w_{ij}}$ (for i,j = 1, 2,...21) is less than 0.05% of the number of training data points N_p in the last epoch of training. Figure 8c marks with an 'x' all nodes that satisfy the condition $f_{w_{ij}} < 0.0005N_p$.

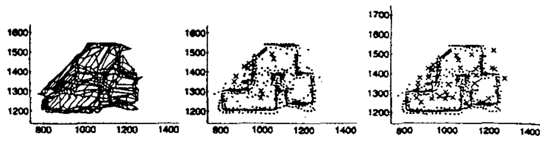

Figure 8 Final Kohonen node mesh and pruned nodes for (b) $f_{w_{ij}} = 0$ and (c) $f_{w_{ij}} < 0.0005N_p$.

4 Experimental Results/Concluding Remarks

Figure 1 shows all ten rooms and corresponding training characters and mature Kohonen cluster centers. With the ten Kohonen room networks trained, four pre-processing systems were tested and compared. The results of these systems were used to determine the most reliable configuration for room classification. The first system used only gross shifting. The second used both gross- and fine-shifting. The third system used gross- and fine-shifting and and pruned $f_{w_{ij}} = 0$ nodes. System four used gross- and fine shifting and pruned $f_{w_{ij}} < 0.0005N_p$ nodes. For each room, 18 test sets, each with varying levels and combinations of data corruption were created and used to test each of the four system configurations.

4.1 Gross Shifting Only

Using only Σ_{rk} and gross shifting produced fair results. Figure 9 shows that the misclassification rate for the *gross shifting only* configuration was 20%.

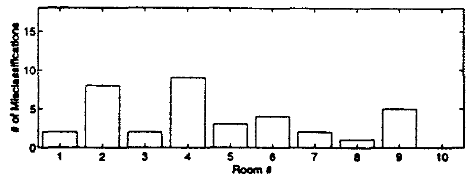

Figure 9. Misclassifications per room for *gross shifting only* system configuration.

Specific rooms were misclassified far more often than others. Room 10 was not misclassified at all while room 4 was misclassified half of the time. This shows that some rooms are distinct enough to be identified just using crude methods like gross shifting. Other rooms, though, are too closely related in shape and design, making recognition a much more difficult task. Although these results were promising, a lower misclassification rate is required.

4.2 Gross- and Fine-Shifting

From figure 10, incorporating fine shifting proved quite successful as the misclassification rate dropped from 20% to 10.56%. Half of the ten rooms had no misclassifications at all. Room number 4, however, suffered from more misclassifications. Shown in figure 1, room number 4 contains data and cluster centers dispersed somewhat evenly throughout the entire room. It is believed that the high misclassification rate associated with room 4 can be attributed to room symmetry and the poor spatial separation of unique features. Regardless, it is clear that fine shifting is an effective pre-processing tool for GSL.

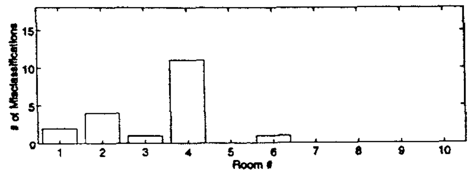

Figure 10. Misclassifications per room for *gross* and *fine shifting* configuration.

4.3 Pruning Dormant Nodes

Kohonen nodes that were placed, through self-organized training, in regions where no training data points actually existed ($f_{w_{ij}} = 0$) were apparently troubling from the start. Figure 11 shows the misclassification results for a system that underwent gross shifting, fine shifting and a pruning of dormant nodes. The performance of this method is a clear improvement of the previous two. There were only two misclassifications (out of 180 test sets) yielding an overall misclassification rate of 1.11%. The fact that this system configuration results in a 98.89% classification rate says a great deal.

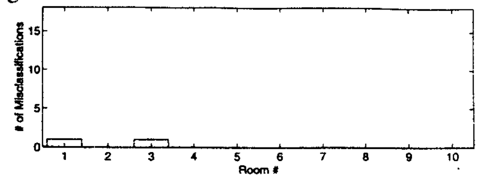

Figure 11. Misclassifications per room for configuration of *gross shifting, fine shifting* and *pruned nodes* with $f_{w_{ij}} = 0$.

4.4 Pruning Low-Frequency Nodes

To see if a more aggressive pruning approach would improve the classifier performance, we tested a system that was subjected to gross shifting, fine shifting and pruning nodes corresponding to $f_{w_{ij}} < 0.0005N_p$. From figure 12, it can be seen that, in fact, with such a minor tendency toward aggressive pruning the classifier performance deteriorated to a misclassification rate of 10%. Hence, it seems the Kohonen is as responsible as the data set to provide feature information. Any loss of true feature information, even from what seems to be outlier data points, can degrade the classifier.

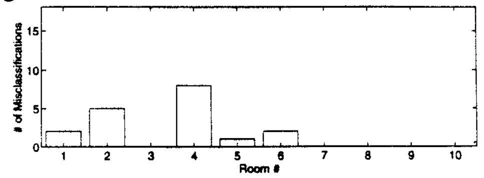

Figure 12. Misclassifications per room for configuration of *gross shifting, fine shifting* and *pruned nodes* with $f_{w_{ij}} < 0.0005N_p$.

4.5 General Comments

Of the four systems, the best classifier utilized gross shifting, fine shifting and pruning of $f_{w_{ij}} = 0$ nodes. This is for a static environment; the effects of other unknown objects on the recognition have not yet been tested. A radical redesign of a room would require any system to relearn the room. The addition of new furniture or the rearrangement of existing furniture will change the signature character of the room. How much the character changes and the effect it has on recognition is situational. This suggests the need for an update learning scheme that notices small physical changes in a previously trained room.

References

[1] B. Barshan and R. Kuc, "Differentiating Sonar Reflections from Corners and Planes by Employing an Intelligent Sensor." *IEEE Trans. on Pattern Analysis and Machine Intelligence* Vol. 12 no. 6, June 1990, pp. 560-569.

[2] B. Barshan and R. Kuc, "Active Sonar for Obstacle Localization Using Envelope Shape Information." *Proc. 1991 Int'l Conf. on Acoustics, Speech and Signal Processing*, 1991, pp. 1273-1276.

[3] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots." *IEEE Trans. on Rob. and Aut.* Vol. 7 no. 3, June 1991, pp. 278-288.

[4] C. Brown, H. Durrant-Whyte, et al., "Centralized and Decentralized Kalman Filter Techniques for Tracking, Navigation and Control." in *Proc. DARPA Image Understanding Workshop*, 1989.

[5] C. Brown, H. Durrant-Whyte, et al. (1989). Kalman filter algorithms, applications and utilities. Oxford University Robotics Research Group.

[6] O. Bozma and R. Kuc, "Building a Sonar Map in a Specular Environment Using a Single Mobile Sensor." *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 13 no. 12, pp. 1260-1269, Dec 1991.

[7] I. J. Cox and J. J. Leonard, "Probabilistic Data Association for Dynamic World Modeling: A Multiple Hypothesis Approach." in *Fifth International Conference on Advanced Robotics*, 1991, pp. 1287-1294.

[8] Cybermotion, User's Manual. 5457 Aerospace Road; Roanoke, VA.

[9] A. Elfes, "Sonar-Based Real-World Mapping and Navigation." *Journal of Robotics and Automation* Vol. 3 no. 3, June 1987, pp. 249-265.

[10] E. G. Elliman, R. N. Banks, "Shift Invariant Neural Net for Machine Vision." *IEE Proceedings*, Vol. 137, Pt. I, No. 3, June 1990, pp. 183-187.

[11] H. R. Everett, G. A. Gilbreath, et al. (1990). Modeling the Environment of a Mobile Security Robot. *Tech. Doc. 1835*. San Diego, CA, Naval Ocean Systems Center.

[12] K. Fukushima, S. Miyake, T. Ito, "Neocognitron: A Neural Network Model for a Mechanism of Visual Pattern Recognition." *IEEE Trans.on Systems, Man, and Cybernetics*, Vol SMC-13, No. 5, Sept 1983, pp 826-834.

[13] K. Fukushima, "A Neural Network for Visual Pattern Recognition." *IEEE Computer*, March 1988, pp 65-75.

[14] S. I. Gallant, *Neural Network Learning and Expert Systems*, MIT Press, Cambridge, MA, 1993, pp. 136-143.

[15] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan College Publishing Co., NY, 1994.

[16] J. P. Holman, *Experimental Methods for Engineers*, 5th ed., McGraw-Hill Book Company, NY, 1989, pp. 57-61.

[17] R. Holdaway , M. White, "Computational Neural Networks: Enhancing Supervised Learning Algorithms via Self-Organization." *Int. J. Biomed Comput.*, 1990, pp. 151-167.

[18] J. A. Janét, R. C. Luo, et al., "Sonar Windows and Geometrically Represented Objects for Mobile Robot Self-Referencing." *IEEE/RSJ International Conference on Intelligent Robotics and Systems*, 1993.

[19] J. A. Janét. Global Motion Planning and Self-Referencing for Autonomous Mobile Robots, MS Thesis. North Carolina State University, Raleigh, NC, 1994.

[20] J. A. Janét, S. G. Goodridge, Ren C. Luo, "Dynamic Motion Control of Sensor-Based Autonomous Mobile Robot." *APS International Conference on Mechatronics and Robotics*, April 1994, Aachen, Germany.

[21] J. A. Janét, R. C. Luo, M. G. Kay, "Autonomous Mobile Robot Motion Planning Enhanced with Extended Configuration-Space and Half-Planes." *IEEE/SICE/AEI Int. Conf. on Industrial Electronics*, Sept. 1994, Bologna, Italy.

[22] J. A. Janét, R. C. Luo, M. G. Kay, "Traversability Vectors Make Autonomous Mobile Robot Motion Planning and Self-Referencing More Efficient." *IEEE/RSJ International Conference on Intelligent Robotics and Systems*, Sept. 1994.

[23] R. Kuc, "A Spatial Sampling Criterion for Sonar Obstacle Detection." *IEEE Trans. on Pattern Analysis and Machine Intelligence* Vol. 12 no. 7, July 1990, pp. 686-690.

[24] R. Kuc and V. Viard, "A Physically Based Navigation Strategy for Sonar-Guided Vehicles." *Int. Journal of Robotics Research* Vol. 10 no. 2, April 1991, pp. 75-87.

[25] Y. LeCun, B. Boser, et. al. "Backpropagation Applied to Handwritten Zip Code Recognition." *Neural Computation*, Vol 1, 1989, pp 541-551.

[26] Y. LeCun, B. Boser, et. al. "Handwritten digit recognition with a back-propagation network." *Advances in Neural Information Processing Systems 2*, (D. S. Touretsky, ed.) pp 396-404, San Mateo, CA: Morgan Kaufmann, 1990.

[27] J. Leonard, H. Durrant-Whyte, et al., "Dynamic Map Building for an Autonomous Mobile Robot." in *IEEE Int. Workshop on Intelligent Robots and Systems*, 1990.

[28] J. Leonard and H. Durrant-Whyte, "Mobile Robot Localization by Tracking Geometric Beacons." *Trans on Rob. and Aut.* Vol. 7 no. 3, June 1991, pp. 376-382.

[29] J. Leonard and H. Durrant-Whyte, *Directed Sonar Sensing for Mobile Robot Navigation*. Kluwer Academic Publishers. Norwell, Massachusetts. 1992.

[30] R. P. Lippman, "An Introduction to Computing with Neural Nets." *IEEE ASSP Magazine*, April 1987, pp. 4-22.

[31] H. Moravec and A. Elfes, "High Resolution Maps From Wide Angle Sonar." in *Proc. IEEE Int. Conf. Robotics and Automation*, 1986, pp. 116-121.

[32] Polaroid, *Ultrasonic Ranging Sys* Cambridge, MA, 1982.

[33] E. Säckinger, B.E. Boser, J. Bromley, Y. LeCun and L.D. Jackel, "Application of the ANNA neural network chip to high-speed character recognition." *IEEE Transactions on Neural Networks*, pp 498-505, 1992.

[34] J. M. Zurada, *Introduction to Artificial Neural Systems*. West Publishing Company, St. Paul, MN, 1992.