Progress in Neural Network-based Vision for Autonomous Robot Driving

Dean A. Pomerleau
Carnegie Mellon University, School of Computer Science
5000 Forbes Ave., Pittsburgh, PA 15213-3890
Phone: (1)412-268-3042, Fax: (1)412-621-1970, Internet: pomerlea@cmu.edu

Abstract

This paper describes recent improvements to the ALVINN system (Autonomous Land Vehicle In a Neural Network) for neural network based autonomous driving. We have recently reported a technique which allows an an artificial neural network to quickly learn to steer by watching a person drive. But the faster the network is trained, the less exposure it receives to novel or infrequent scenarios. For instance, during a typical four minute training run, the network sees few if any examples of passing cars. When a rare situation like this occurs during testing, its lack of coverage in the training set can result in erratic driving. By modeling the appearance of infrequent scenarios and then using the model to augment the training set, we can teach the network to generalize to situations not explicitly represented in the live training data. Using this technique, a network trained over a two mile stretch of highway was able to drive autonomously for 21.2 miles at speeds of up to 55 miles/hour.

1. Introduction

The ability of an artificial neural network to perform a task is heavily influenced by the quality of its training set. If the training set contains examples taken from the full range of situations the network is expected to handle, the network will learn to perform the task accurately. But if important situations are missing during training, the network is likely to perform poorly when it later encounters the novel circumstances.

In the domain of autonomous driving, we have shown that the connectionist architecture shown in Figure 1 can quickly learn to steer by watching a person drive [2]. The network receives live input from a camera on the vehicle. The network is trained using back-propagation [3] to activate the output unit representing the driver's current steering direction. After approximately four minutes of watching a person drive on a particular type of road, the network is able to take over for itself and drive at up to 55 miles per hour. Individual networks have been trained to drive in a wide variety of situations, including single and multi-lane roads with and without lane markings.

2. Transitory Feature Problem

Certain types of transitory driving situations have proven troublesome for this connectionist approach to autonomous driving. Two examples of temporary but problematic situations are illustrated in the real video images of Figure 2. The left image shows a typical multi-lane highway scene used to train the network. It has features including the lane markings down the left, right and center of the road, and a patch of grass from the median in the upper left corner. The other two images illustrate deviations from this typical scene. The center image shows a jersey barrier

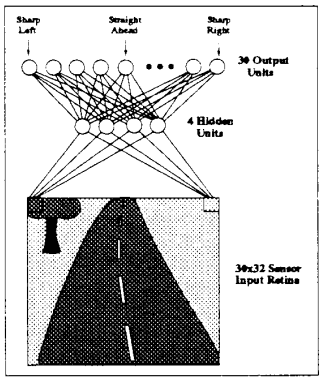

Figure 1: Neural network architecture for autonomous driving.

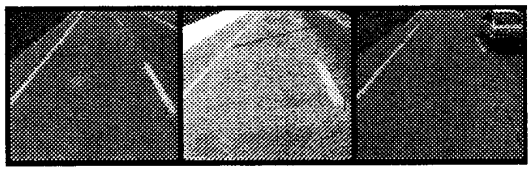

Figure 2: Three video images of a multi-lane highway.

instead of a patch of grass in the upper left corner as the vehicle drives over a bridge. The right image shows the vehicle passing another car.

The reason this type of transitory disturbance causes trouble is that the network is trained over a relatively short stretch of road (< 2 miles). As a result, during training the network is not exposed to all the possible driving situations it might encounter when driving autonomously. In particular, since situations like the two illustrated in Figure 2 are relatively rare and limited in duration, even if the network sees them during training, it doesn't learn enough from its brief exposure to handle them appropriately. As a result, while the network is capable of reliably driving in situations closely resembling those it was trained on, when it encounters novel situations like the ones illustrated above, it frequently steers incorrectly.

The influence spurious image features can have on driving performance can be seen in Figures 3. The left image in Figure 3 illustrates a typical reduced resolution multi-lane highway image like the ones ALVINN was trained on. The dark triangle in the upper left is the green grass on the left side of the road. Notice that ALVINN's output response is nearly identical in position to

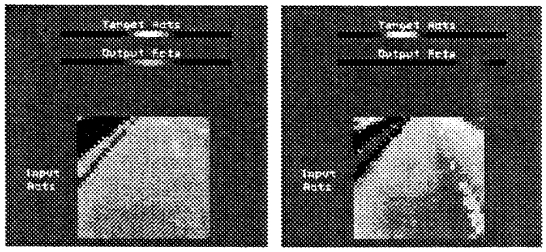

Figure 3: Two low resolution images of a multi-lane highway and the network's response. The image on the left is a typical highway image, with no unexpected features. As a result, the network responds correctly. The image on the right contains a guardrail on left side of the road (the white stripe in the upper left corner). Since the network did not see a situation like this during training, its steering response is far from correct.

the target, indicating ALVINN is steering in the correct direction. The image on the right of Figure 3 has a guardrail on the left side, which appears as a white stripe in the upper left corner. Notice ALVINN's steering direction is significantly disturbed by this relatively minor change to the image.

The reason for this disturbance is illustrated in Figure 4. Each of the rectangles labeled "Input-Hidden1 Weights" through "Input-Hidden4 Weights" represents the weights projecting from the input retina to one of the four hidden units in the network. White squares within each of these rectangles represent excitatory weights, black squares represent inhibitory weights, and grey squares represent weights with small magnitude. The connections from the pixels in the upper corners of the image to each of the hidden units have relatively large magnitude, indicating the network is using those pixels to determine the correct steering direction. The reason the network came to rely heavily on those pixels is that during training, no guardrails or passing cars appeared in the periphery during training. In the case of the left periphery, it contained only grass over the entire training sequence. As a result, the size of the grass patch was a good indication of how sharply the vehicle should turn. The larger the patch, the more towards the left the vehicle was, and therefore the more the vehicle should turn to the right. Because of the high correlation between the size of the dark patch and the correct steering direction, the network learned to use this feature to determine how to steer. Therefore, the connections from the upper left pixels were given large weights, while the center and bottom portions of the image were largely ignored. The same argument hold true for the pixels in the upper right corner: the network relies heavily on them because of their consistency during training.

The quantitative effect transitory image features like guardrails have on driving performance is shown in the leftmost two bars of the graph in Figure 5. These two bars represent the performance of a network trained without noise on a sequence of multi-lane images like the ones shown in Figure 2. The leftmost bar shows the network's average steering error on a disjoint set of 180 multi-lane road images. The steering error represents the curvature difference between the steering arc

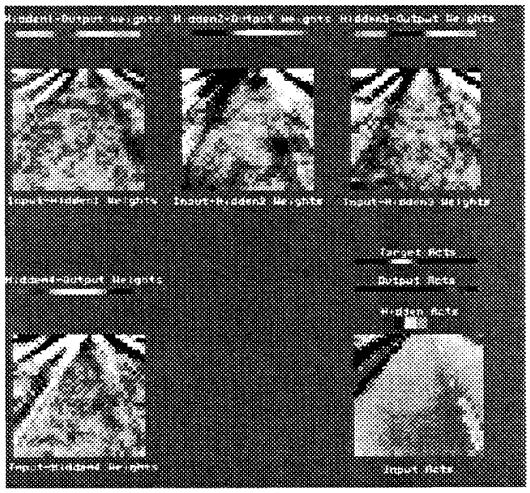

Figure 4: Weight diagram of a network trained without noise.

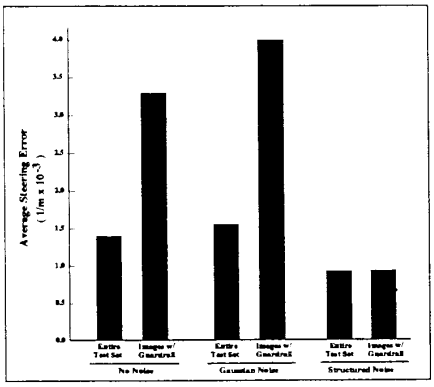

Figure 5: Graph illustrating the performance of networks trained using three different techniques on a set of 180 images, and on a subset of 32 images containing a guardrail.

suggested by the network for an image and the arc the person was driving along when the image was taken. The bar next to it, labeled "Images w/ Guardrail" shows the average steering error of the same network on a 32 image subset from the test sequence which contained a feature not present in the training images, namely a guardrail on the left side of the road. The network's steering error more than doubles on images containing the novel feature, clearly illustrating the detrimental impact these features can have on performance. If allowed to steer autonomously over the stretch of road where the guardrail images were taken, the network would have steered off the road.

The problem caused by transitory image features results from insufficient diversity in the training examples. The network does not encounter these transitory features frequently enough during the short training period to learn to ignore them. One useful aspect of the problem, exploited in the solution below,

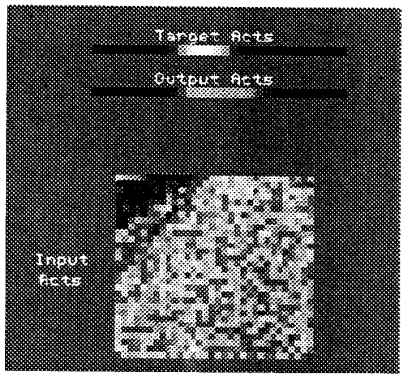

Figure 6: A multi-lane highway image corrupted with gaussian noise.

is that these transitory features do not radically alter the overall appearance of the image. Due to the redundancy of features in the image, if the network could learn to ignore spurious features, it should be capable of driving accurately.

3. Training with Gaussian Noise

A commonly employed technique for improving generalization from a limited amount of training data is to add uncorrelated gaussian noise to the training patterns [4]. The idea is that adding noise to the input prevents the network from relying on idiosyncrasies in the training patterns to perform the task. Sietsma and Dow found a dramatic improvement in generalization when noise was added to the training patterns on a frequency classification problem. In their task, the input was a 64 unit vector whose input activation pattern formed a sine wave of a particular frequency. The task was to classify an input sine wave according to its frequency, regardless of its phase within the input field. They found that when gaussian noise was added to the training patterns, the resulting network made dramatically fewer classification errors on novel, noise-free input patterns (0.5% vs. 17%).

We performed a similar experiment by adding various amounts of gaussian noise to the road images used for training. The noise had a mean of 0 and a standard deviation ranging from 0.4 to 1.2 (The pixel intensity values ranged from -1.0 to 1.0). Figure 6 shows one corrupted road image used for training.

Surprisingly, networks trained with this noisy input performed uniformly worse than network trained without noise, as illustrated by the bars in Figure 5 labeled "Gaussian Noise". They represent the *best* performance of any of the networks trained with gaussian noise. Both on the entire test set, and particularly on the guardrail subset of images, the networks trained with noise steered less accurately than the network trained without noise.

The reason for this drop in performance is evident in Figure 6. The finer image features such as the lane markings, which were visible in the noise-free image of Figure 3, have been obscured by the noise. The only visible feature is the patch of grass in the upper left corner. The network had no choice but to key on the size and location of the grass patch to determine the steering direction. This degraded performance on the test set because despite its large size, the grass patch is actually a less reliable feature than the lane markings, since it is frequently obscured by guardrails and jersey barriers.

Gaussian noise is a poor model of the important "noise" that occurs in the input while driving. The noise that matters results from the appearance or disappearance of coherent 2-D features such as guardrails and other cars. Modeling these irrelevant features and adding them to the input while training can improve generalization dramatically, as illustrated by the last two bars in Figure 5. Networks trained by adding structured noise to the input, as described in the next section, generalized better on the test set as a whole than networks trained without noise or with gaussian noise. Even more significant was the performance improvement of the network trained with structured noise on images with spurious features in them, as illustrated by the low error of the network on the guardrail images. In fact, the network's steering performance on the novel guardrail images was not significantly different than its performance on the test set as a whole, demonstrating that it has learned to ignore spurious image features (See Figure 5).

4. Characteristics of Structured Noise

A number of straightforward characteristics of structured image noise can be employed to improve network generalization. The most obvious features is the high degree of spatial correlation. Important noise does not appear as corruption of random image pixels. Instead, the physical object (like a car or guardrail) causing the noise makes a 2-D projection into the input image. As a result, the important image noise takes the form of a two-dimensional regions of nearly uniform intensity. While objects with multiple or variable intensities do occur, their rarity makes a uniform intensity model a reasonable approximation.

A more subtle characteristic of structured image noise results from the fact that objects can change in three ways. Objects can suddenly appear in the image, obscuring part or all of a previously visible feature. An example of this effect is when a car passes and obscures the lane markings. Objects can also disappear from the image, making previously hidden features visible. This effect might occur when the guardrail disappears, revealing the grass on the other side. Finally, a feature can change color or brightness, as when the centerline changes from white to yellow. Shape is not considered a changing feature characteristic, since objects visible when driving are assumed to be rigid.

Another useful domain-specific characteristic of structured image noise is that when driving, irrelevant features are more likely to occur in the periphery of the image than in the center. Even if during training the appearance of the terrain off to the side of the road remains constant, it is helpful for the network to learn that the appearance of the periphery can change dramatically. This way the network learns not to rely on the position of the guardrail or the size of the grass area in the off-road region, since they may be obscured or disappear at any time.

The size of significant image features can be constrained by estimating the size and distance to noise features. The retinal size of significant features ranges from a few tens of pixels for

the lane markings or guardrail to several hundred pixels for large objects such as passing trucks. The *shape* of significant image features is also constrained in the domain of autonomous driving. Image features are frequently oriented with their primary axis pointed towards the vanishing point of the image. The technique for incorporating these characteristics of image features when generating noise is described in the next section.

5. Training with Structured Noise

Structured noise characteristics are used during training to determine the appearance of the noise to be added to the input patterns. Instead of adding gaussian noise to each pixel, the following technique is employed to add or remove coherent two-dimensional features from the training patterns.

First, for each pattern on each epoch of the back-propagation, a decision is made whether to add noise to that pattern or not. Empirically, adding noise on 3 out of every 4 presentations of a pattern seems to be a good compromise between teaching the network to be insensitive to noise and teaching it to process clean images correctly.

Once the decision is made to add noise to a pattern, the next question is where to put it. We have found that when trained on images with a single noise feature, a network is able to generalize to images with multiple noise features. Therefore, at most one noise feature is added per image during training. The location for this single noise feature is selected randomly with a bias towards the periphery of the scene, corresponding to the upper corners of the image. That is, the likelihood that a pixel will be chosen as the starting point for the noise feature is proportional to its proximity to one of the upper corners of the image. This periphery bias roughly models the tendency of noise features to appear away from the path directly ahead of the vehicle.

After determining the starting location for the noise feature, a decision is made whether a new feature should be added or an existing feature should be removed from that position. This choice is made randomly, but with a bias towards deleting a feature if one exists at that location, and towards adding a feature if that location appears to be "feature free". The judgement concerning whether a feature exists at a location is made based on the size of the uniform region that location is part of. First, a region is grown around the chosen location to encompass all the contiguous pixels whose intensity is within a fixed threshold of the chosen pixel's value. If the size of this region is within the size range of interesting features as characterized above, then the location is considered to be part of a feature and that feature is removed. If the size of the region falls outside the size range of interesting features, it is considered part of the background, and a new feature is added at that location.

Removing a feature is easy. The pixels defining the feature have already been determined in the region growing step described above. Deleting the feature involves changing the values of all the pixels within the region to a new randomly chosen intensity. Altering a region's intensity models the corresponding object changing color. By selecting the new intensity to be the same as the intensity of the area surrounding the feature, the disappearance of the feature can also be simulated. The result of using this technique to alter an existing feature is illustrated

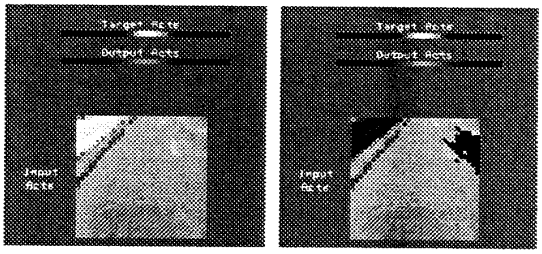

Figure 7: Two images augmented with structured noise. In the left image, an existing feature (the patch of grass in the upper left corner) has been altered by changing its intensity. In the image on the right, a new dark feature has been added on the right.

in the left image of Figure 7. The image is identical to the one on the left of Figure 3, except that the patch of grass in the upper left corner has changed intensity from very dark to very light.

Adding a new feature to model the sudden appearance of objects such as a guardrail or automobile is more difficult, since unlike in the feature alteration process described above, the shape of the feature is not known. The shape of spurious 2-D image features can in theory be arbitrary. There is the weak constraint that significant features tend to have their major axis pointing towards the vanishing point. However, this orientation specificity is difficult to implement directly in the noise generation model since it requires knowledge of the sensor geometry.

A simpler technique for generating reasonable spurious features involves using the shape of the feature detectors the network develops in its internal representation to bias the shape of the simulated noise features. As is evident in the weights from the input retina to the hidden units in Figure 4, the network develops a strong model of the shape of important image features. The network's knowledge that features tend to be oriented towards the vanishing point is demonstrated by the tendency of features in the receptive fields of the hidden units to converge towards the top.

To bias the new feature's shape using the shapes of the features in the network's internal representation, a random hidden unit is first selected. The values of the weights from the input retina to this hidden unit are then used as the "image" in which to grow the new feature. In other words, a pixel in the vicinity of the feature's start pixel is chosen to be an element of the feature if the weight of the connection from it to the chosen hidden unit is sufficiently similar in value to the weight of the connection from the feature's start pixel to the chosen hidden unit. By biasing the shape of noise features by the shape of important features in the internal representation, this technique ensures that only noise features with a reasonable shape are added to the input.

To mimic situations in which image noise does not precisely align with significant feature detectors, a spatial coherence constraint is added to the feature growing algorithm. The following equation balances the tendency of the feature growing algorithm to follow hidden unit weight contours with the tendency to create a spatially compact feature. A pixel i will be included in a newly created features if:

$$\alpha |w_{ih} - w_{sh}| + (1 - \alpha)|i - s| > T$$

In this equation, w_{ih} is the weight value of the connection originating at pixel i in the input image and terminating at the chosen hidden unit h. The variable w_{sh} corresponds to the weight of the connection originating at s, the pixel chosen as the start position of the feature, to the chosen hidden unit h. The quantity $|i - s|$ represents the Euclidean distance between pixels i and s in the input retina. The constant α is used to weight the tendency to follow hidden unit weight contours with the tendency to keep the feature spatially compact. With an α of 0, the new feature will form a circular region centered on pixel s. With an α of 1, the new feature will grow to include all the pixels in the image having connections to hidden unit h with a weight close in magnitude to the connection from pixel s to unit h. Randomly choosing an α from the range 0.1 to 0.3 for each image creates realistic looking new noise features. The threshold T is used to limit the growth of the new feature. If the weight from pixel i to hidden unit h differs significantly from the weight from the start pixel s to unit h, or if pixel i is a great distance from pixel s, then the threshold T will be exceeded by the above sum and pixel i will not be included in the new feature.

Once the position and shape of the noise feature is determined, it is made to contrast with the surrounding area by filling it with a randomly chosen uniform brightness. A real video image in which a noise feature (the dark region in the upper right) has been added is shown in the right image of Figure 7. Notice how the feature appears close to the upper corner of the image due to the periphery bias built into the noise feature generation algorithm. The right image of Figure 7 also illustrates how growing noise features along important contours in the hidden unit representation results in features with appropriate orientations for the domain. In this example, this bias results in a feature oriented diagonally towards the vanishing point. The feature is spatially compact due to the bias in the noise generation algorithm towards choosing pixels in the vicinity of the feature's start pixel. This combination of biases based on known, consistent image feature characteristics results in the generation of noise features with realistic appearance. In fact, the feature in right image of Figure 7 appears very much like a car passing by on the right side of the vehicle, as in the right image of Figure 2.

As illustrated in the bar graph of Figure 5, a network trained by adding and removing features steers more accurately than a network trained without noise, particularly on images containing spurious features. The reason for this is evident in the weight diagram of Figure 8. The network shown in this diagram was trained on the same sequence of images as the network shown in Figure 4, except that on each iteration of back-propagation, 75% of the patterns were randomly selected to have a noise feature added to their input. Varying the noise at every epoch prevents the network from learning characteristics of a single noise feature. The remaining 25% of the patterns were presented without noise to ensure the network would also handle noise free situations.

It is clear by looking in the upper corners of the input-to-hidden weight arrays that adding structured noise to the image has the desired effect. Namely, the network learned to rely less on features in the periphery than the network depicted in Figure 4. The network developed detectors primarily for the line marking the left lane boundary, since this is the feature that appears most reliably in this type of road image. Because of its frequent occlusion and absence from the image, the dashed line marking the right boundary of the lane was given little importance in the internal representation. The resulting performance improvement is illustrated by the input pattern and corresponding network response in the lower right corner of Figure 8. The input pattern is the same guardrail image that confused the network trained without noise on the right side of Figure 3. This network handles the guardrail image perfectly, since it has learned not to be disturbed by features in the periphery.

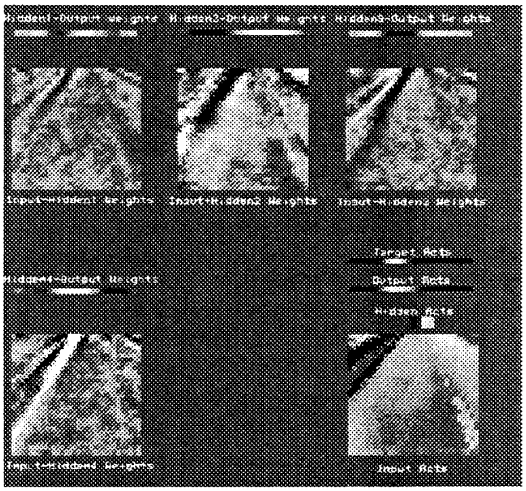

Figure 8: Weight diagram of a network trained with structured noise.

6. Improvement from Structured Noise Training

The bar graph in Figure 5 illustrates the significant increase in steering accuracy which results when structured noise is added in the training process. This steering accuracy improvement translates directly into dramatic gains in driving performance. Quantitatively, we have found that a network trained without adding structured noise on a two mile stretch of a four lane divided highway was capable of driving autonomously for only about four miles before straying from the road because of a spurious feature. Driving successfully for even this relatively short distance was somewhat fortuitous, in that it was achieved on a stretch of road free from guardrails and other potentially confusing permanent features, and at a time of day when there was very little traffic to confuse the network. When other vehicles did appear, the network trained without noise frequently swerved towards or away from them depending on their brightness relative to the background. The swerves were relatively small, so the safety driver allowed the run to continue without interference. The situation which ended the run after four miles was the one depicted in the center image of Figure 2. The vehicle encountered a bridge with jersey barriers along the edge of the road. This spurious feature caused the vehicle to swerve dramatically, forcing the safety driver to intervene.

The network trained with structured noise drives significantly

better. Its best run was 21.2 miles without human intervention, on the same highway that caused the network trained without noise to fail after 4.0 miles. It made it over the bridge that caused the previous network to fail, and through a number of other situations that would have caused trouble for the network trained without noise. The reason the run came to an end after 21.2 miles was that the width of the road changed significantly, causing the network to become confused.

7. Discussion

The appearance or disappearance of irrelevant features can disrupt a network's driving when the network's training did not demonstrate their irrelevance. Adding structured noise to the training patterns using a model describing the characteristics of irrelevant features significantly improves driving performance. A simpler gaussian model of image noise has been shown to be ineffective at compensating for this problem because it does not mimic the important characteristics of real world, structured noise.

But are there other possible solutions which do not require such complex modeling? In theory, training the network over a longer stretch of road than the two miles currently employed should result in a more representative training set and hence a more robust network. However there are a number of shortcomings to this approach. One long term goal of this work is the development of a "super cruise control system" capable of controlling both the vehicle's speed and steering. If the training period required by this super cruise control is too long, it will be impractical.

But even given unlimited training time, the scarcity of irrelevant features would make it difficult to train a network to ignore them. For instance, over many miles of highway driving, the size of the patch of grass on the left side of the image is a good indicator of the correct steering direction. Only in rare situations, like going over a bridge, will relying on this feature get the network in trouble. Even if the training period were extended to ensure encountering this type of situation, its low frequency would make it beneficial for the network to ignore these few patterns. This is because the network could lower the total error over all training patterns by employing the patch of grass to improve the steering performance on the vast majority of patterns, while suffering substantial error only on the few patterns in which the patch of grass is missing. In other words, the high correlation between an irrelevant feature and the correct output would result in the network employing the feature despite being exposed to a few situations where this degrades performance.

The current model of structured image noise has obvious shortcomings. Noise features do not always occur in the periphery. When driving in traffic, cars directly in front of the vehicle will obscure central image features. But even this simple model is sufficient to demonstrate that dramatic improvements in neural network generalization can be achieved by actively employing domain-specific knowledge during the training process.

References

[1] Pomerleau, D.A. (1992) Neural network perception for mobile robot guidance. PhD. Dissertation. Carnegie Mellon technical report CMU-CS-92-115.

[2] Pomerleau, D.A. (1991) Efficient Training of Artificial Neural Networks for Autonomous Navigation. In *Neural Computation 3:1.*

[3] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986) Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations* pp. 318-362, Bradford Books/MIT Press, Cambridge, MA.

[4] Seitzma, J., and Dow, R. (1991) Creating Artificial Neural Networks that Generalize. *Neural Networks, Vol. 4* pp. 67-79, Pergamon Press.