ELVIS: Eigenvectors for Land Vehicle Image System

John A. Hancock
Charles E. Thorpe

CMU-RI-TR-94-43

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

December 1994

This research was partly sponsored by: DARPA, under contracts "Perception for Outdoor Navigation" (contract number DACA76-89-C-0014, monitored by the US Army Topographic Engineering Center) and "Unmanned Ground Vehicle System" (contract number DAAE07-90-C-R059, monitored by TACOM); DOT/National Highway Traffic Safety Administration, "Run-Off-Road Counter Measures", (contract number DTNH22-93-C-07023,); NSF, "Annotated Maps for Autonomous Underwater Vehicles" (contract number BCS-9120655); and the Dept. of Transportation, "Automated Highway System."

Table of Contents

Figures

Abstract

ELVIS (Eigenvectors for Land Vehicle Image System) is a road-following system designed to drive the CMU Navlabs. It is based on ALVINN, the neural network road-following system built by Dean Pomerleau at CMU. ALVINN provided the motivation for creating ELVIS: although ALVINN is successful, it is not entirely clear why the system works. ELVIS is an attempt to more fully understand ALVINN and to determine whether it is possible to design a system that can rival ALVINN using the same input and output, but without using a neural network.

Like ALVINN, ELVIS observes the road through a video camera and observes human steering response through encoders mounted on the steering column. After a few minutes of observing the human trainer, ELVIS can take control. ELVIS learns the eigenvectors of the image and steering training set via principal component analysis. These eigenvectors roughly correspond to the primary features of the image set and their correlations to steering. Road-following is then performed by projecting new images onto the previously calculated eigenspace. ELVIS architecture and experiments will be discussed as well as implications for eigenvector-based systems and how they compare with neural network-based systems.

1.0 Introduction

Researchers at CMU have been working on autonomous driving systems for nearly a decade. One of the most successful robot road-following systems is ALVINN, a simulated neural network built by Dean Pomerleau at CMU. ALVINN uses a color video camera mounted above the passenger compartment of the vehicle to watch the road. A steering wheel encoder allows ALVINN to observe human steering. After observing the road and the human steering responses for approximately two minutes, ALVINN can operate the steering wheel to follow the road on its own.

Unfortunately, why ALVINN works has remained somewhat of a mystery. ALVINN has several components which contribute to its success: careful generation of training image sets, image subsampling, color balancing, output representation, and the neural network. One model of how the neural network in ALVINN works is that the first set of weights between the input and hidden layers learns a reduced representation of the training set representing the important image features. The weighted sums at the output calculate the steering based on those image features present in a new image. ELVIS (Eigenvectors for Land Vehicle Image System) seeks to verify this model: it calculates the eigenvectors of the training set which form an explicit reduced representation of that training set. Projection of a new image onto the eigenspace produces a steering output. ELVIS also attempts to answer the question of whether the neural network itself is the key to ALVINN's success, or whether it is possible to design a system that can rival ALVINN, using the same input and output, but without using a neural network.

In its original design, ELVIS replaced ALVINN's neural network with an eigenvector representation that uses the same inputs and outputs as ALVINN. In successive versions, alterations have been made to ELVIS to improve performance and to learn more about its capabilities and limitations. Most of the experiments have centered on improving the video preprocessing since evidence suggested that improvements in the preprocessing stage could significantly enhance the performance of ELVIS.

In this paper we first introduce the components and structure of ELVIS. The training and processing methods that ELVIS uses to drive the vehicle are then explained. Section three describes the various preprocessors used with ELVIS. We finish with a comparison of the video preprocessors and a discussion of the merits of ALVINN and ELVIS. We explain in what types of scenarios it is possible to replace a neural-network based system like ALVINN with an eigenvector system like ELVIS.

2.0 ELVIS

2.1 Common Components of ALVINN and ELVIS

Both ALVINN and ELVIS are designed to interpret video images of roads and produce steering commands. The method each uses to calculate steering output given the same image is different, but the overall structure of the two systems is the same, and ELVIS was designed to use some of the ALVINN modules. A comparative block diagram of the two systems is given below.

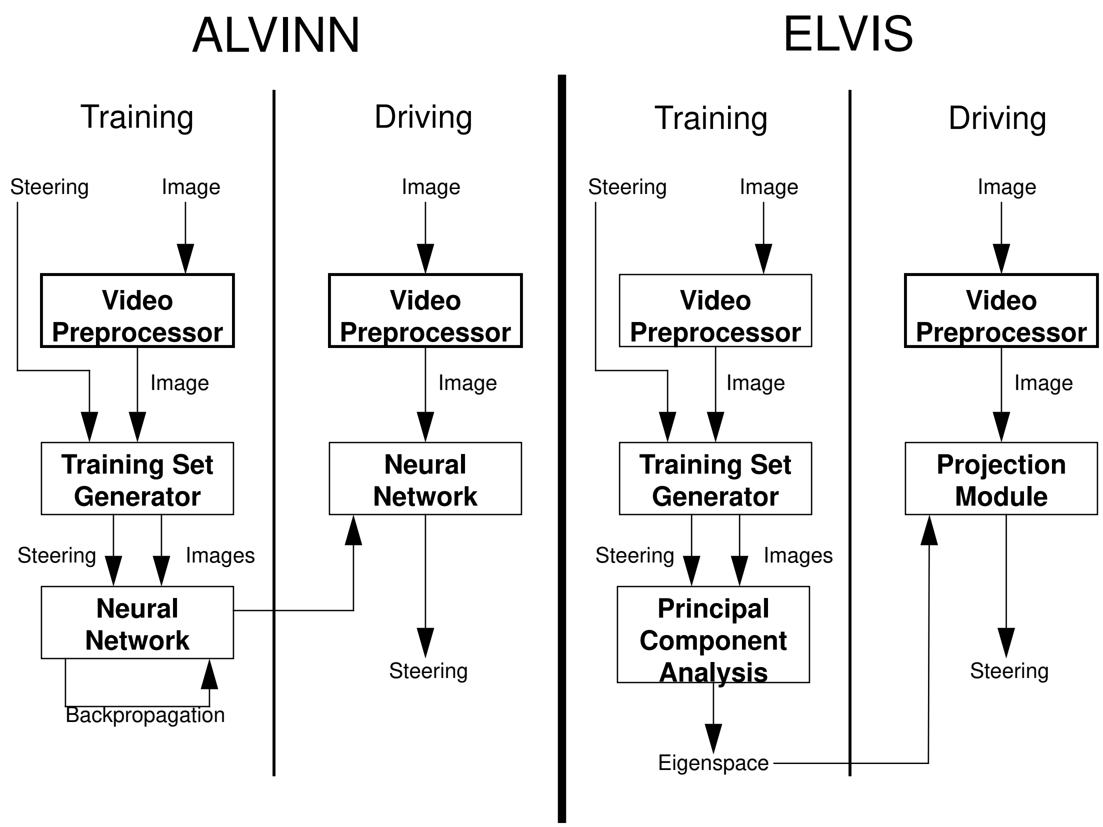


Figure 1. Block diagrams of ALVINN and ELVIS for both the training and driving phases of the process. Note that ELVIS replaces the ALVINN neural network with a principal component analysis in the training phase and with a projection module in the driving phase.

Color Balancing

In the ALVINN system, the three-band RGB color images produced by the camera are preprocessed to produce a single-band image. The color transformation is performed to reduce the amount of data, and more importantly, to enhance the image features important for road-following. Although numerous transformations of the color space have been tried with ELVIS, the color-balanced images produced through the ALVINN preprocessing work best in ELVIS as well. Each pixel is normalized to have a value between 0 and 1. The normalized value is given by:

$$v = [\alpha \times B / 255] + [(1 - \alpha) \times B / (R + G + B)]$$

where R,G, and B are the raw red, green, and blue values for a given pixel, and α is a weighting factor between 0 and 1. The partial normalization by intensity provides some tolerance to lighting variation within a given image, helping to filter out variations caused by shadows. To heighten contrast within images and decrease variations in overall intensity between images, the normalized one-band pixel values are modified further based on the histogram of the pixel values. A fixed percentage (generally 10%) of the top and bottom values are set to 1 and 0, respectively, while the other values are stretched to span this range. Empirically, it has been determined that the blue band contains the most useful contrast for road following.

Subsampling

To reduce the computational expense of processing large images, the video preprocessing must reduce the dimensions of the 480 x 512 digitized camera image. The neural network in ALVINN uses a 30 x 32 input image layer, and this has been the resolution typically used for ELVIS as well. Rather than average each pixel region or randomly sample the image, the preprocessor compromises. A small percentage of the pixels within each region in the original image is randomly sampled and averaged to produce the reduced image pixels. This combination of averaging and subsampling blurs the image slightly to reduce the effects of noise without eliminating important image variations or making road features unrecognizable.

Output Representation

The output for each system is a 50 element vector in which each element represents the strength of votes for a particular steering direction. During training, the correct steering direction is represented by a gaussian set of votes within the output vector, centered at the actual steering direction. This output representation has several advantages over using a single-valued output to indicate steering direction. First, the single-valued output cannot represent both the network decision and the network confidence in that decision. A single-valued output for a completely recognized scene indicating a shallow right turn might well be indistinguishable from the output for a partially recognized scene which calls for a hard right turn[7]. This is especially a problem for ALVINN which does not have an independent method of computing a confidence level. A second problem with a single-valued output representation is that it would not allow an ALVINN hidden unit or an ELVIS eigenvector to vote for more than one steering direction[7]. It is important to allow an eigenvector to vote for multiple directions because an eigenvector does not necessarily correspond to one feature or one type of image. Finally, the gaussian output results in a robust, distributed system so that even if one output unit fails, it is still possible to drive. This would be especially important if either of these systems were implemented in hardware.

Training Set Generation

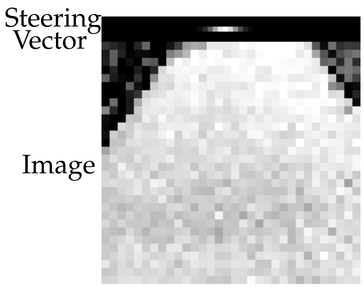
There are several pitfalls in the generation of training image sets. The first is bias: if the training used only images from left turns, the system would learn that always turning left minimizes output errors. Thus, the images selected must be balanced, including all ranges of steering positions with backgrounds or off-road areas that reasonably span the space of expected driving situations. Both ELVIS and ALVINN will do poorly if trained on a paved road, and then expected to drive on a dirt road. The other problem in training is that, in general, the human trainer drives too well so the system is never shown how to recover from minor steering errors. The solution both systems use is to create derived training images from the actual images. A geometric transform is applied to the input image to rotate or shift it slightly as if the vehicle were slightly off the desired path. The steering angle is corrected correspondingly. This provides a much broader training set for ALVINN and ELVIS. The geometric transforms for image and steering are more fully described by Pomerleau[6].

2.2 Training

During training, ELVIS learns eigenvectors of the image and steering training set. The eigenvectors are a set of basis vectors that guarantee the best linear image reconstruction, on average, given limited representational power. These eigenvectors not only represent the principal features of the image set but also these features' correlations to steering (see Figure 4 for some example eigenvectors). Given a new image we use the eigenvectors to produce the best reconstruction of the image and its features. Since the eigenvectors also tell us how these features correlate to the steering, this allows us to compute the proper steering position. The principal eigenvectors model large, common features which we assume are useful for driving.

We represent each two-dimensional image with B color bands, R rows, and C columns, as an $n = B \times R \times C$ element one-dimensional vector. For training purposes, we add the steering vector elements to the end of the image vector. This image/steering vector combination is a training vector. The number of elements in each training vector is N = n + d where d is the number of steering units. For a monochrome 30 by 32 image with 50 output units, N is 1010. Given a set of M (typically M = 400) training vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots \mathbf{v}_M$, the average of the training set is defined by $\mathbf{a} = (1/M) \sum \mathbf{v}_i$. By subtracting the average vector from each vector, we can obtain the difference vectors $\Delta_i = \mathbf{v}_i - \mathbf{a}$. Given the Δ_i we form the covariance matrix $\mathbf{C} = \mathbf{A}\mathbf{A}^T$ where $\mathbf{A} = [\Delta_1, \Delta_2, \ldots]$.

We illustrate the formation of the matrix $\mathbf{A}$:



$$\longrightarrow \Delta_i = \begin{bmatrix} I_1 \\ I_2 \\ . \\ . \\ I_n \\ S_1 \\ . \\ . \\ S_d \end{bmatrix} - \mathbf{a} \xrightarrow[\text{pairs}]{\text{x } 400} \mathbf{A} = \begin{bmatrix} \Delta_{1,1} & \Delta_{2,1} & \cdots & \Delta_{400,1} \\ \Delta_{1,2} & \Delta_{2,2} & \cdots & \Delta_{400,2} \\ . & . & \cdots & . \\ . & . & \cdots & . \\ . & . & \cdots & . \\ \Delta_{1,N} & \Delta_{2,N} & \cdots & \Delta_{400,N} \end{bmatrix}$$

Figure 2. Forming the matrix A, the precursor to the covariance matrix

The covariance matrix $\mathbf{C}$ is N by N which is very large, and it is computationally expensive to find its eigenvectors. There is a way to greatly reduce the amount of computation. Since there are only M images, there can be at most M independent eigenvectors. Since M is generally much less than N, it is advantageous to find the principal components of the M by M matrix $\mathbf{U} = \mathbf{A}^{\mathbf{T}}\mathbf{A}$ and then convert them into the eigenvectors of matrix $\mathbf{C}$ rather than calculate them directly from $\mathbf{C}$. To convert the eigenvectors of $\mathbf{U}$ to eigenvectors of $\mathbf{C}$, we simply multiply each vector by $\mathbf{A}$: if $\mathbf{x}$ is an eigenvector of $\mathbf{U}$, it is simple to show that $\mathbf{A} \bullet \mathbf{x}$ is an eigenvector of $\mathbf{C}$ [4],[9]:

$$\mathbf{Ux} = \lambda\mathbf{x} \to \mathbf{AUx} = \mathbf{A}\lambda\mathbf{x} \to \mathbf{AA}^{\mathbf{T}}\mathbf{Ax} = \mathbf{A}\lambda\mathbf{x} \to \mathbf{C}(\mathbf{Ax}) = \lambda(\mathbf{Ax})$$

It is important to note that the eigenvalue of the eigenvector will be the same in either $\mathbf{U}$-space or $\mathbf{C}$-space, so that the vectors will be found in the same order whether we use $\mathbf{U}$ or $\mathbf{C}$.

We then find ten to fifteen eigenvectors of the matrix $\mathbf{U}$ with the largest eigenvalues. This is done by the power method. The vector $\mathbf{U}^{n} \bullet \mathbf{x}$ for an arbitrary vector $\mathbf{x}$ will tend to converge towards the eigenvector of $\mathbf{U}$ with the largest eigenvalue as n becomes large. In practice, sufficient convergence occurs for n between 5 and 50, depending on the eigenvalues of the eigenvector and its nearest competitors. By orthogonalizing the matrix $\mathbf{U}$ with respect to this eigenvector, we can repeat the process to obtain the orthogonal eigenvector with the next highest eigenvalue and so on. The orthogonalized matrix $\mathbf{U}_{i+1}$ is simply $\mathbf{U}_i - \lambda\mathbf{x} \bullet \mathbf{x}^{\mathbf{T}}$, where $\mathbf{x}$ is the eigenvector and λ is its eigenvalue. This orthogonalization process does not affect the other eigenvectors, but removes the dimension of the eigenspace that lies along $\mathbf{x}$.

Because we must change the original matrix $\mathbf{U}$ for each eigenvector we obtain, we gradually degrade the numerical accuracy of the matrix and therefore the accuracy of each eigenvector is, in general, worse than the previous one found. The power method has the additional problem that it will be very slow to converge to an eigenvector if there is another with a similar eigenvalue. This will often result in the power method finding some linear combination of the two eigenvectors. For these reasons, the power method is usually not recommended for obtaining more than about three eigenvectors. For our purposes, however, the eigenvectors are well-behaved and finding the exact eigenvector is unimportant; obtaining eigenvectors that have slight numerical inaccuracies or that are actually a linear combination of two eigenvectors poses no problem for a few reasons. First, the information in an image is very distributed and mostly redundant, so small inaccuracies do not lead to any real loss of information. Secondly, since our input images themselves are noisy, there is no reason to attempt to eliminate inaccuracies completely. Finally, obtaining linear combinations of eigenvectors is not a problem because we linearly recombine all the eigenvectors in order to steer. Empirical evidence supports these claims as well. The power method has the advantage that it is simple to implement, and faster than many other methods.

Since the inception of ELVIS, the training phase speed has been improved greatly. With less-efficient methods, training ELVIS used to take on the order of 10 to 20 minutes for a typical batch of 400 ALVINN pre-processed images. Presently, this takes only 2 minutes on a Sparc 10. Most of the time is spent on the large matrix multiplication. Since this matrix multiplication is linear with respect to the number of elements in a training vector and quadratic with respect to the number of images, training with 3-band color images takes 6 minutes.

2.3 Road-Following

Once ELVIS finds the principal eigenvectors, $\mathbf{e_i}$, and the average training vector, $\mathbf{a}$, of the roadspace, it can steer on its own. To drive, ELVIS takes a new image, $\mathbf{x}$, performs the same preprocessing as in training, and then projects it onto the eigenspace formed by the principal eigenvectors (usually ten of them). To project the image onto the eigenspace, we perform the calculation:

$$\mathbf{v} = \mathbf{a} + \sum_{i=1}^{10} ((\mathbf{x} - \mathrm{image}(\mathbf{a})) \bullet \mathrm{image}(\mathbf{e}_i))\,\mathbf{e}_i$$

where image($\mathbf{z}$) is simply the image portion of the composite vector $\mathbf{z}$. In other words, the average image is subtracted from the new image. To calculate the projection of this difference image onto the eigenspace, its dot product with the image portion of each of the eigenvectors is calculated. These dot product results become the coefficients of the eigenvectors in the summation. Then the average training vector is added to the summation result to complete the vector reconstruction.

In this way, a composite vector, $\mathbf{v}$, is formed which consists of both the reconstructed image and the steering vector. A single steering command is then computed from the steering subvector (the last 50 elements of $\mathbf{v}$) by calculating the center of mass of the peak of activation surrounding the output unit with the highest activation level. This steering command is then sent to the vehicle controller module. Using the center of mass of the activation rather than the most active output unit allows for sub-unit steering resolution, thus improving driving accuracy[6].

Besides knowing the proper steering direction, the system should also have a measure of the system's confidence in that calculation. A reliability estimate is important because it allows the system to disregard the new steering instruction if confidence is low, and it lets the user know that the system should be retrained if the average confidence becomes low. One simple way to measure reliability is to compute the sum-squared difference error between the image and its reconstruction. This image reconstruction error measures how closely the reconstructed image resembles the original. If the error is low, then the new image must resemble some of the images in the ELVIS training set, and so confidence should be high. If the error is high, then the image does not match closely with training set images and so the system confidence in the steering direction should be low.

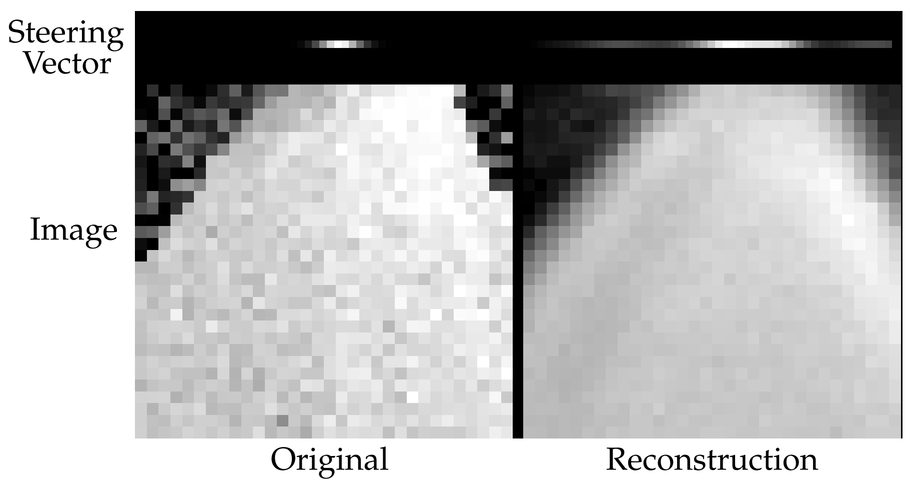


Figure 3. Original image and steering vector and their reconstructions

As can be seen from Figure 3, the reconstructed image tends to be smoother than the original image, generally capturing the essence of the geometry of the road while not reconstructing noise or fine details. The reconstructed steering vector also tends to be flattened. This can result in rather broad peaks. Wider peaks can cause steering errors of up to three units (out of 50), but these errors are manageable. More difficult to handle are multiple-peaked responses. Although these occur infrequently, they can lead to drastically incorrect steering vectors if ELVIS chooses the wrong peak (ELVIS computes the center of the peak with the largest amplitude). Since road-following is a continuous process that does not require sudden changes in steering (except at intersections which ALVINN and ELVIS are not equipped to handle), it would be possible to create a heuristic that would choose the peak that is closest to the present steering direction or place a threshold on the change in steering direction. This may help in avoiding the multiple peak problem. Of course, the independence of the individual results is one of the strengths of the system and removing this independence could introduce other problems.

3.0 Experiments

The video preprocessing is an important factor in ALVINN's success. ALVINN would never learn to drive if the road or road features were indistinguishable from the rest of the image. The quality of the separation of road from non-road is the measure of success of the video preprocessor. As described below, tests using simulated data with ELVIS revealed that the ALVINN video preprocessing did leave room for improvement. As an attempt to both improve ELVIS accuracy and further understand how and why ALVINN works, we measured ELVIS performance using a variety of image types and video preprocessing parameters. All of the tests were performed on the same batch of images taken of a one-lane road near CMU. These images were often characterized by very harsh shadows and varying lighting conditions. This presented quite a challenge to the preprocessors.

3.1 Simulated Data

ELVIS was first tested on synthesized road images. The simulator produced monochrome road images with appropriate steering vectors calculated by geometric transformations. Pixels were perfectly classified; road pixels were white and non-road pixels were black. After subsampling, the images were still mostly white and black with some gray pixels along lane borders. By testing on perfectly classified data, we could show ELVIS concept viability and establish a baseline performance level against which we could judge the quality of various preprocessing techniques. ELVIS performance with the simulated data was good, with the standard deviation of the steering direction error being less than 0.8 steering units out of 50 (which corresponds to the difference between driving straight and on an arc of 350 m radius). After 2.5 m of travel, (the maximum distance travelled between images) this error leads to a displacement of 0.9 cm from the road center and an error of 0.41^{o} in heading. Given real data, a perfect preprocessor would provide a clear separation between road and non-road such as that present in the simulated images.

3.2 ALVINN Color-balanced Images

Measuring ELVIS performance on image sets using the ALVINN color-balanced preprocessor, as described previously, provided the second baseline for our tests. Steering results were good, but not as accurate as with simulated data (see Table 1 on page 12). The standard deviation of the steering direction was several times that found with simulated data, typically around 3.0 steering units out of 50 for 10 eigenvectors (an error in curvature of 10.7 km^{-1}). The ALVINN system itself performs somewhat better -- typically producing errors with a standard deviation of 2.7 units out of 50. We expected to surpass the performance of the ALVINN color-balanced scheme by providing ELVIS with more information.

3.3 Color Images

Providing color images seemed an obvious first step to boost ELVIS performance. We hypothesized that the use of 3-band color images would improve the accuracy of ELVIS by providing ELVIS with important cues to distinguish between road and off-road pixels. Color provides humans with many obvious cues for driving. Green grass and yellow and white lane markers contrast well with black asphalt. However, ELVIS performance declined when we replaced the ALVINN pre-processed images with sub-sampled RGB color images. The standard deviation in the steering error for color images was approximately twice as great as that for the ALVINN images (see Table 1 for ELVIS results with different preprocessors). Histogramming and normalization of the color images to reduce intensity variations between images, as done in ALVINN pre-processing, improved the results only slightly. Although reconstruction of the input images themselves was quite good, the reconstruction of the steering vectors was poor. Often broad peaks or multiple peaks occurred in the reconstructed steering vectors, causing large steering errors. Apparently, the RGB values were not well correlated with the steering values. We thought that perhaps ELVIS needed color to be provided in non-linear combinations which would better distinguish between road and non-road.

3.4 Images in Intensity-Saturation-Hue (ISH) Space

After the disappointing performance of ELVIS using color RGB images, we tried to train and test ELVIS with images in the ISH color space as defined by Ballard and Brown[1]. First, we trained and tested ELVIS on monochrome images representing each of the intensity, saturation, and hue bands. ELVIS performed miserably with intensity images, far worse than with color images. Intensity was not a good feature for distinguishing road from non-road. Part of the problem was that image reconstruction in intensity space became a matter of shadow matching, which had no correlation to steering. Of course, poor performance of intensity relative to color was expected since RGB values provide far more information than intensity.

Saturation was a better cue, but still did not perform well. Road pixels tend to be less saturated than the green grass and yellow lane-marker pixels, so saturation preprocessed images tended to have very dark road regions and lighter regions for grassy areas. This resulted in good reconstruction of the image and steering vector for most inputs. ELVIS with saturation information failed horribly, however, in the infrequent cases where the background was less saturated than the road. Brownish dirt on the side of the road which was less saturated could cause a color reversal in the saturation images: the road would be lighter than the background rather than darker. Rather than adding eigenvectors with dark road sections to the average image, good image representation in these cases required the multiplication of these eigenvectors with negative weights, resulting in a well-reconstructed image but an inverted steering vector which would have steered the vehicle off the road.

Calculating hue is an expensive operation requiring both a square root and an inverse trigonometric function. Hue is also a very noisy operation for low saturation pixels, which results in large variations on the road portions of the images. The noise led to large image reconstruction errors because the smooth eigenvectors were unable to reconstruct the noisy data. Despite these drawbacks, hue did provide the best information for driving of the three ISH bands. So, although the large image reconstruction errors indicated an unreliable steering vector, the steering reconstruction was reasonably accurate (though still not as good as the ALVINN preprocessing). Wraparound of hue values was ignored.

Finally, 3-band ISH images and the three 2-band combinations were tested. The 2-band combination of saturation and hue images performed the best of the two and three band ISH band images. This indicates that intensity was a misleading cue in the presence of better information.

3.5 The Ohta Color Space

Neither RGB nor ISH information was able to segment the road pixels from the non-road pixels as reliably as ALVINN's color-balancing scheme, and it was still unclear whether ELVIS could perform well without non-linear transformations of the color space. So we decided to try a color representation proposed by Yuichi Ohta[5]. Ohta performed trials to derive linear color features with large discriminant power for segmenting outdoor color scenes. He found that all his test images could be segmented near-optimally if he used a transformed color space. The 3 axes of this space were intensity, (R-B)/2, and (2G-R-B)/4.

We trained and tested ELVIS using the images in the transformed color coordinates. Results were better than with either RGB or ISH information, although still worse than the ALVINN preprocessing. As in the ISH images, results were improved slightly when intensity information was dropped altogether. Additionally, performance was only slightly worse if we dropped the third band, (2G-R-B)/4, as well. (R-B)/2 seemed to segment the image fairly well because the road pixels tended to have more blue than red, while background pixels tended to be more reddish. This is similar to the way in which the ALVINN preprocessor and the Martin Marietta ALV road-follower function[8].

3.6 Increased Resolution

The digitized images are 480 rows by 512 columns, and are normally reduced to 30 by 32 by a combination of averaging and subsampling to smooth the image and to allow for computational tractability. This coarse resolution, however, often blurs lane markings, which provide important cues for driving along multi-lane roads. To improve ELVIS performance, especially for multi-lane roads, we performed similar pre-processing operations to produce higher resolution 60x64 images in both ALVINN color-balanced and RGB modes. Performance decreased, however, on the one-lane road and only improved slightly on the two-lane road. Despite the blurring in the low resolution images, the lane markings were still quite recognizable. Hence, the lack of improvement might have been predicted: for ELVIS to work, the steering vector must depend on large portions of the image and not just small features or details, as we will more fully explain later.

3.7 Steering Vector Length

Increasing the number of output steering units seemed to decrease the output accuracy. Doubling the steering vector length more than doubled the standard deviation of the steering error (measured in steering units). Why this is so is not entirely clear. It is possible that the greater the number of steering units, the greater the interference these output units cause in the principal component analysis in terms of finding image features to rely upon. Since a longer steering vector also slows down training, a 50-unit length vector was used for all tests rather than the 100-unit vector used presently in ALVINN. Smaller vectors were also tried, but we felt that fringe effects might outweigh the slight advantage given by the shorter vectors.

3.8 Dimensionally-Altered Images

We should expect that some portions of our field of view are more important than others for the purpose of road-following. Looking behind us, to the side, or down at our feet is less helpful than looking straight ahead. Thus, it is important for a road-following system to know where to look. Positioning the camera and setting a reasonable field of view solves much of the problem, but not all of it. ALVINN can be taught to "ignore" portions of the image which are unimportant for driving by assigning these areas low weights in the network, and it can learn to pay more attention to other areas of the image. ELVIS has no such ability. It treats the entire image equally in trying to minimize the image reconstruction error. ALVINN trains by explicitly minimizing the error between its output and the correct output given an input image. ELVIS, on the other hand, minimizes the error between an image and its projection onto the eigenspace. It is then hoped, but not guaranteed, that good reconstruction of the image leads to construction of an appropriate steering vector. Thus, we might expect that if we could filter out the unimportant or possibly misleading portions of the images, that ELVIS would not only perform faster, but would also produce more accurate steering vectors. We would not expect to see improvement by ALVINN (except with respect to computational speed) by removing unimportant portions of the images since it can ignore them. The ELVIS images were modified so that a given portion in each image was thrown out before training and testing. For the case of deleting the bottom quarter of each image, ELVIS steering accuracy improved. As predicted, elimination of the top portion of the image led to a dramatic decrease in performance, since this is the portion of the road that a driver uses most in determining the correct steering position.

3.9 Results

TABLE 1. Steering Error Results on One-Lane Road for Various Video Preprocessors

Preprocessing Method	Vector Size	10 eigenvectors Average Error	10 eigenvectors Standard Deviation	15 eigenvectors Average Error	15 eigenvectors Standard Deviation
Simulated Data	1010	0.046	2.592	-0.036	2.091
ALVINN (Color Balanced)	1010	-0.089	11.115	-0.238	8.843
RGB, normalized	2930	-1.806	22.436	-0.786	18.233
Intensity	1010	-0.455	39.417	2.624	35.940
Saturation	1010	2.752	28.128	3.851	23.772
Hue	1010	-0.523	11.879	-0.441	10.912
Intensity, Saturation, Hue	2930	0.892	13.916	0.427	11.140
Ohta Color Space	2930	-0.327	11.264	-0.178	10.158
Ohta Space, bands 2 and 3	1970	-0.288	10.713	0.025	9.326
ALVINN, high resolution (60x64)	3890	-0.210	13.312	-0.647	11.115
ALVINN, 200 steering units	1160	-1.84	19.28	-0.675	13.906
ALVINN, top 3/4 of each img	786	0.046	11.089	-0.583	9.397

TABLE 2. Steering Error Results on Two-Lane Road for Various Video Preprocessors

Preprocessing Method	Vector Size	10 eigenvectors Average Error	10 eigenvectors Standard Deviation	15 eigenvectors Average Error	15 eigenvectors Standard Deviation
Simulated Data	1010	0.043	4.473	0.036	3.918
ALVINN (Color Balanced)	1010	-0.789	6.389	-0.654	4.519
RGB, normalized	2930	-3.154	10.322	-2.574	8.761
Intensity	1010	-0.960	10.251	-0.946	8.899
Saturation	1010	-0.896	8.811	-0.601	7.516
Hue	1010	-0.626	6.439	-0.629	5.248
Intensity, Saturation, Hue	2930	-0.661	8.700	-0.348	6.549
Ohta Color Space	2930	-0.267	7.396	-0.380	5.881
Ohta Space, bands 2 and 3	1970	-0.363	7.349	-0.388	4.782
ALVINN, high resolution (60 x 64)	3890	-0.359	6.457	-0.366	5.628
ALVINN, 200 steering units	1160	-0.555	6.909	-0.623	5.580
ALVINN, top 3/4 of each img	786	-0.939	6.364	-0.740	4.395

Note: The tables above give the average and standard deviation of the steering error, measured as the difference between the computed curvatures, and are in units of $(km)^{-1}$. An error of 10 km^{-1} corresponds to a difference between driving straight and on an arc of 100 m radius. After 2.5 m of travel (the maximum distance travelled between images) this error leads to a displacement of 3.1 cm from the road center and an error of 1.43^{o} in heading.

4.0 Discussion

ALVINN's success provides no guarantee that ELVIS will work. First, since ELVIS is calculating linear functions of the inputs, it assumes that non-linear combinations are not required. Second, the eigenvector decomposition finds the best representation of the covariances among all the data. This will give the best linear reconstruction of the data, but will not necessarily find the best mapping from inputs to outputs. For a general data set, there is no reason to assume that globally similar inputs should produce globally similar outputs. It could be, for instance, that large portions of the input data are highly correlated, but that the outputs depend solely on a small portion of the input which has no relationship to the rest of the input. If the road-following problem had this kind of structure, we would expect better performance from the neural nets than from the eigenvectors, since the network training explicitly minimizes the error in the outputs.

It is possible to reformulate ELVIS so that it minimizes output error. One way is to zero those portions of the covariance matrix that represent image-image or steering-steering correlations. In this way, ELVIS learns only information that correlates steering output to image input. This has been tried, but steering results became worse rather than better. Evidently, the correlations between image pixels are important to the driving task in general, though they may not improve results on the training set. A second, and perhaps better, method would be to learn a least-squares mapping matrix of image input to steering output. This would correspond to learning a 50 by 960 matrix. This matrix might be further broken down into eigenvector components. Finding the least-squares mapping was performed and is described later in Section 5.0.

The fact that ELVIS does usually calculate accurate steering responses, in spite of the above reservations, gives insight into the nature of the problem. First, linear mapping is sufficient. Steering direction is a smooth function, as represented by ALVINN and ELVIS. A different output representation, such as turn radius, may cause many more problems: near "straight ahead", the turning radius gets increasingly large, then jumps from positive infinity to negative infinity. Representing such an output might require mechanisms beyond the linear calculations of ELVIS. Second, the inputs and output of ELVIS are correlated with each other. Small changes in input typically map into small changes in output; highly correlated input images have highly correlated output vectors[7]. This is a fortuitous characteristic of the road-following problem. It is also apparent that the output depends on large-scale features that involve much of the image; small details are unimportant. The fact that performance did not improve significantly with higher resolution images illustrates this as well.

4.1 Comparing the Preprocessors

The optimal preprocessor, measured in terms of ELVIS performance, is the one that manages to put all the relevant information into the least number of vector elements. This cuts down on both computation costs and on inaccuracies introduced by irrelevant or misleading information. The trade-off is that this optimal preprocessor is often computationally expensive, less intuitive, and difficult to design. Obviously, the optimal preprocessor gives us the solution directly, and to design it, we must solve the problem. The video preprocessor should only seek to enhance or suppress certain image features, and not solve the problem itself. In general, we have thus settled for preprocessors that are non-optimal but that are fairly easy to calculate, intuitive, and perform satisfactorily. Although we cannot build the absolutely optimal preprocessor, it would be possible to build the optimal linear preprocessor or a multi-stage preprocessor. We could calculate the linear color combination that would give us the best separation of road pixels from non-road pixels. Another possibility would be to use clustering techniques to provide good separation of road from non-road. By first learning the common color clusters and assigning each cluster to the two superclusters road and non-road, the preprocessor could then achieve very good classification. This clustering technique has, in fact, been used before for autonomous driving at CMU, but using five to twenty clusters and a model-based approach after the clustering has been performed[2],[3].

The ALVINN preprocessor is relatively inexpensive to calculate and it performs the best of the preprocessors we tested. Unfortunately, it is rather unintuitive since it is based on empirical findings, and so we might have hoped we could design something more intuitive that would have worked as well. We found, however, that the other preprocessors simply could not separate road pixels from non-road pixels as reliably.

Direct use of RGB data requires minimal preprocessor calculation and its success would have been theoretically satisfying, but individual red, green, and blue values simply could not distinguish between road and non-road and did not correlate well with the steering direction. Instead, an individual red, green, or blue value was more likely to indicate whether the pixel lay in a shadowed or non-shadowed than whether it lay in the road or off the road. Intensity presented the same problem, only heightened. Finally, saturation, which seemed an intuitive way to distinguish road from non-road, simply was not reliable enough.

Hue is perhaps the most intuitive cue, since hue measures what humans tend to think of as color. But, as mentioned earlier, there are several drawbacks to using hue information. The first is that it is expensive to calculate. The second is that since hue is measured in radians, a hue of zero is the same as a hue of 2π, and care must be used in placing this discontinuity in an area of the color spectrum that is not commonly found in the road images since ELVIS can not recognize that these values are the same. Rather than artificially creating large diffences in hue space between points close together in RGB space because of the discontinuity, the system could use sin(hue) or cos(hue). This, however, would have the opposite problem: that of creating small differences in sin(hue) space between points far apart in RGB space. The third problem with hue is that the calculations are very sensitive to small changes in pixel values for pixels with low saturation. Despite these problems, ELVIS performed well using hue information.

The Ohta color space, like the ALVINN color-balancing, was a less intuitive preprocessing method that gave better results than more traditional color schemes. The Ohta color space has the advantages that it requires almost no calculation in the preprocessing stage (which would be very useful if the system were to be implemented in hardware), and that since it is a linear transformation of the color space, the transformation is reversible. Since the Ohta color space is somewhat warped via the histogramming and normalization process, transformation back into RGB space does leave the colors somewhat stretched, but it still leaves the image features quite recognizable (see eigenvectors in Figure 6). The disadvantage of using the Ohta color scheme over the ALVINN scheme is that it requires nearly twice as much calculation during driving (if using two bands), and that accuracy is not quite as good. However, ELVIS results are still reasonably good when decreasing the computational costs by using only the red minus blue band of the Ohta space.

Eliminating actual portions of the image was an additional step towards putting the relevant information into the least number of vector elements. It is a relatively simple matter to make some guess as to what portions of the image might be eliminated without degrading performance. To take this further, we may examine the picture of the covariance matrix itself, and remove the portions of the image which do not seem to correlate with steering. However, removing numerous pieces from each image complicates matters and eventually destroys the simplicity of the system. The example in Table 1 shows that performance was not significantly affected by eliminating the bottom quarter of each image. Reducing the size of the image also allows us to use more eigenvectors to improve performance without increasing computation.

4.2 One-Lane Versus Two-Lane Results

After ELVIS had been trained and tested with a variety of preprocessors on the same image set of the one-lane road on Flagstaff Hill, we tested the preprocessors on another set of images. Our second batch was of images taken of a two-lane road in Schenley Park. As the tables above show, all of the preprocessors performed well on the two-lane road images. In fact, ELVIS with the worst of the preprocessors on the two-lane road performed comparably to ELVIS with the best preprocessor on the one-lane road. There are probably several reasons for this.

It is evident from examining the eigenvectors obtained from the two-lane road images (see Figure 5 and Figure 6) that the most important image feature is the yellow lane marker. Each of the eigenvectors is reasonably featureless except for bands near the location of the marker. The absence of harsh shadows on the two-lane road made it a reasonably simple matter to find lane markers by virtually any preprocessing method. The eigenvectors from the one-lane road image set show that the locations of road and non-road patches are the most important features. However, off-road areas in these images varied between grass, leaves, dirt, and trees, making them far more difficult to detect from color information than the lane markers in the two-lane image set which are very consistent in color.

Thus, the lane markers were more reliable as driving cues. Additionally, the two-lane road had shallower curves, so the images tended to be more consistent and the range of possible steering angles was decreased.

5.0 Additional Linear Algebraic Methods

ELVIS demonstrated that linear methods were sufficient to learn to drive autonomously. In order to improve results, we formulated other linear algebraic methods for solving the road-following problem. We implemented one such system which learned a transformation matrix as a direct mapping from input image to output steering vector. A system which learned a direct mapping was guaranteed to provide optimal performance in terms of least-squares error on the training set for a linear method and provided the most intuitive way of calculating the steering given a new image.

The system learned an m by n mapping matrix $\mathbf{W}$, where m is the number of steering units and n is the number of image elements, so that given an n x 1 image vector $\mathbf{x}$, we can compute the m x 1 steering vector $\mathbf{y}$ by simply computing $\mathbf{W} \bullet \mathbf{x} = \mathbf{y}$. There are multiple ways to calculate $\mathbf{W}$. The problem can be solved as an iterative learning perceptron problem or directly as a batch process. For the batch process, consider the set of input images as an input matrix $\mathbf{X}$ and the set of output vectors as an output matrix $\mathbf{Y}$, where each column of $\mathbf{X}$ and $\mathbf{Y}$ corresponds to a single input image and output steering pair. Then

$$\mathbf{W} \bullet \mathbf{X} = \mathbf{Y}$$

and we can compute $\mathbf{W}$ by finding the pseudoinverse of $\mathbf{X}$. Since the system solves for $m \times n$ unknowns (the number of elements in $\mathbf{W}$), we must have at least $m \times n$ equations to fully constrain the problem. Each input-output pair provides us with m equations (one for each steering unit), so we must have at least n input-output pairs. Since n was 960 for our ALVINN-preprocessed images, we used approximately 2000 images to provide adequate constraints. We then computed a standard least-squares fit for $\mathbf{W}$ by computing the pseudoinverse of $\mathbf{X}$:

$$\mathbf{W} = \mathbf{Y} \bullet \mathrm{PSI}(\mathbf{X}) = \mathbf{Y} \bullet \mathbf{X}^{\mathbf{T}} \bullet \left(\mathbf{X} \bullet \mathbf{X}^{\mathbf{T}}\right)^{-1}$$

Surprisingly, performance using this method was somewhat worse than ELVIS. Although it performed successfully on the training set (the standard deviation of the steering error was less than one unit), it did not perform well on the the test set data. There were several reasons why we expected the least-squares method to outperform ELVIS. Since we were training on far more images than with ELVIS and on twice as many as were needed to constrain the system, we expected that the least-squares fit should be able to generalize fairly well. The images in the two batches were rather similar, so that the same input-to-output mapping should have worked for the test batch as well as the training batch. Finally, the least-squares fit was calculating and using nearly five times as many coefficients to achieve a good mapping from input to output as ELVIS did. Evidently, however, the least-squares fit of **W** computed on the training set was not as generalizable to other road data as the eigenvectors were. Most likely, the least-squares method was partially cueing off of noise. Besides giving worse performance, the least-squares method required many more images and far more computation both in training and at run-time.

6.0 Conclusion

ELVIS demonstrates that it is unnecessary to use a neural network to achieve a good direct mapping from image input to steering output, but we should not expect ELVIS to outperform ALVINN. While ELVIS principal component analysis minimizes the total error in the image reconstruction and steering vector, ALVINN directly minimizes the steering output alone. Since there are bound to be some small areas of the image which are not highly correlated with the steering output, ALVINN is able to produce better steering results. ALVINN has additional advantages. First, although the speed of training of the two systems is roughly equal, ALVINN is faster at run-time. ALVINN requires approximately 40% of the calculations of ELVIS (using 10 eigenvectors) to compute the steering output, though ELVIS could sacrifice accuracy by using just 4 eigenvectors to make the computational load equivalent to that of ALVINN[7]. Furthermore, since ALVINN training is an incremental rather than batch process, it is simple to train until performance is satisfactory, then stop. Although it would be possible to train ELVIS incrementally, training speed would decrease as more images were added. This makes it impractical to use much more than 400 images in training ELVIS. In contrast, it takes the same amount of time for ALVINN to update its neural network during training on the 400th image as it does on the first image.

The primary advantage of ELVIS is its simplicity and lack of pre-defined structure. We can use ELVIS with a variety of different eigenspaces with a different number of image elements in each. We must only perform the necessary preprocessing and send each image to the appropriate ELVIS module and eigenspace. It would be possible to then combine steering results from several ELVIS modules (given enough computational power). Perhaps the only other advantage of ELVIS is that the eigenvectors hold more symbolic meaning for the system user than the ALVINN hidden units. The eigenvectors give the user an understanding of what features are important for image reconstruction, and indirectly, important for driving.

In considering the use of eigenvectors for other applications in the place of a neural network, the scenario must be one in which the input and output are highly correlated. The output should depend on large features in the input, and the output representation should also be smooth so that a linear solution will be adequate. Eigenvectors might be very successful in an application where there are several types of distributed, highly-correlated information, any of which could be input or output. The eigenspace could be used to function as an associative memory. It is relatively trivial to recover a good approximation of 10% of an individual data entry (like the image/steering vector) if we have 90% of it (just the image) and the eigenvectors of the data set, no matter which portion of the data is missing. Neural networks would not be as successful here because there is a well-structured concept of input and output.

It makes no difference to ELVIS which portion of the vector is missing. For example, we could recover a portion of the image, given the rest of the image and the steering vector. We would calculate the dot products of the portions of the vector which are not missing with the corresponding pieces of the eigenvectors. This projects the vector onto the eigenspace, giving us an approximate reconstruction of all of the information. Turk and Pentland demonstrated with their eigenface work that it was possible to recover an approximation of a person's face (and recognize it) even when a significant portion of it was occluded[9]. To obtain a good approximation, of course, it would be necessary to already have the data entry (a face in this case) in the database when the eigenspace was calculated. But even without any prior knowledge of the face, it might be possible to create a reasonable reconstruction since all faces are relatively similar.

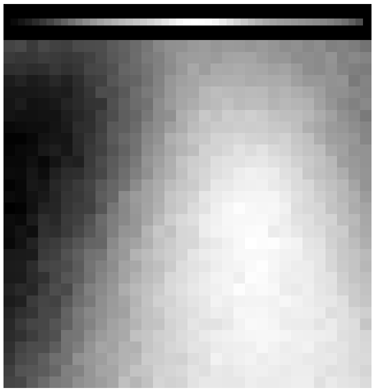
The concept of using eigenvectors to recover missing data could extend, however, to the case where there was not just image information, but multiple types of distributed information, stored in a single entry or vector. How much missing information we could recover would depend on how similar individual entries were and the ratio of the number of eigenvectors to the total number of vector entries. A traditional neural network architecture, however, would demand that we know a priori which (and how much) information was being provided and which was missing; input and output are more restrictive concepts.

In short, the neural network is well-suited to this task, but it is not the most critical part of ALVINN. The network does not differ greatly from the eigen calculations. Much of ALVINN's power comes from the robust representation, careful training set generation, and a good choice of video preprocessing. There is room for improvement in the video preprocessing, and it should be possible to provide a better separation of road and non-road. However, if a better segmentation is achieved, it might be more advantageous to approach the problem from a model-based method (such as in Crisman's SCARF and UNSCARF systems[2]) rather than using a neural network or eigenvector system.

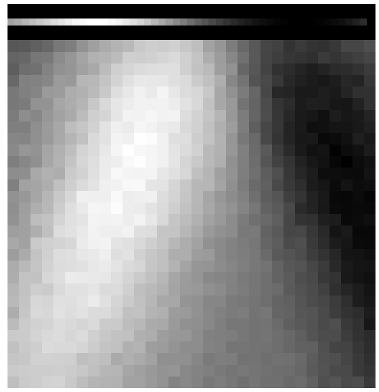
7.0 Acknowledgements

Many thanks go to Dean Pomerleau and Todd Jochem for their help with the ALVINN code, both on and off the vehicle. Keith Gremban was helpful in the selection of an algorithm for computing the eigenvectors. The authors also wish to thank Shumeet Baluja, Martin Martin, and Garth Zeglin for their helpful comments. Support for this research came from: DARPA, under contracts "Peception for Outdoor Navigation" (contract number DACA76-89-C0014, monitored by the US Army Topographic Engineering Center) and "CMU Autonomous Ground Vehicle Extension" (contract number DAAE07-90-C-R059, monitored by TACOM); DOT/National Highway Traffic Safety Administration, "Run-Off-Road Counter Measures", (contract number DTNH22-93-C-07023,); NSF, "Annotated Maps for Autonomous Underwater Vehicles" (contract number BCS-9120655); and the Dept. of Transportation, "Automated Highway System."

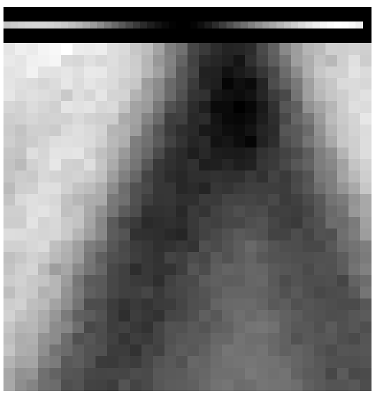
Average Image and Eigenvectors for One-Lane Road



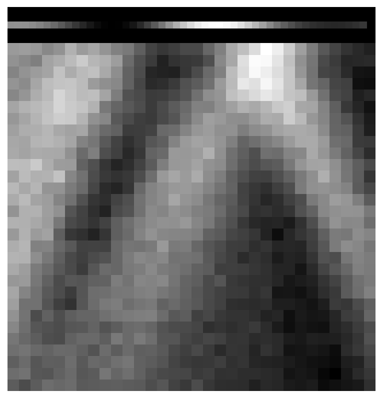
Average Image



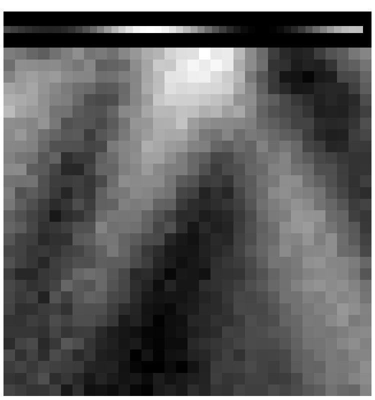
Eigenvector 0(largest eigenvalue)



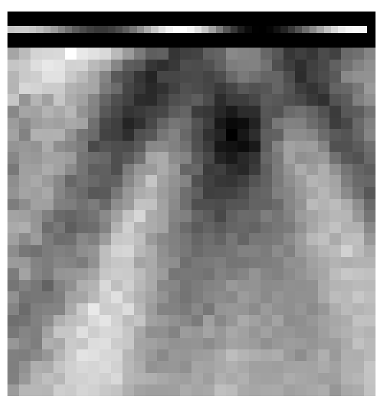
Eigenvector 1



Eigenvector 2



Eigenvector 3



Eigenvector 4(smallest eigenvalue)

Figure 4. These eigenvectors were formed with the ALVINN color-balanced preprocessing method from a batch of images taken on Flagstaff Hill. The portion at the top of each image represents the steering vector. The lighter areas correspond to road regions.

Average Image and Eigenvectors for Two-Lane Road

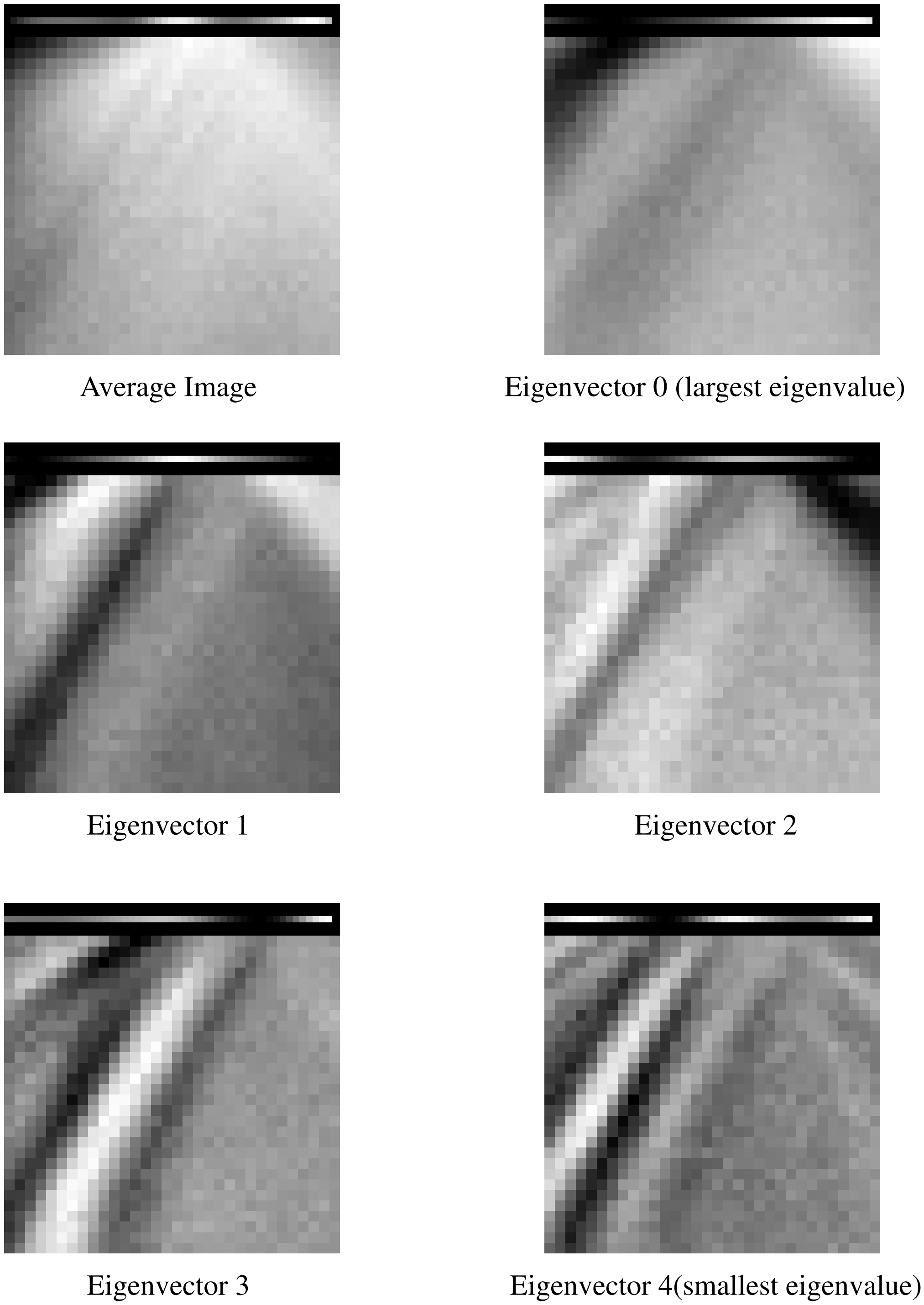
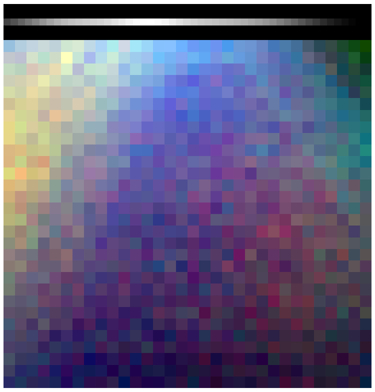
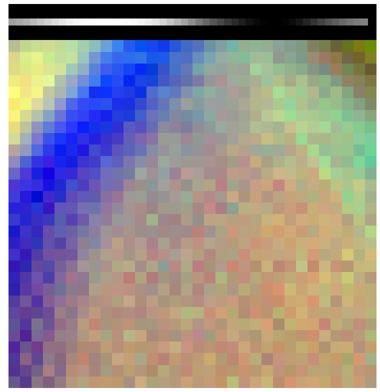


Figure 5. These eigenvectors were formed with the ALVINN color-balanced preprocessing method from a batch of images taken on Schenley Drive. The bands towards the upper-left of each eigenvector represent the location of the lane markers.

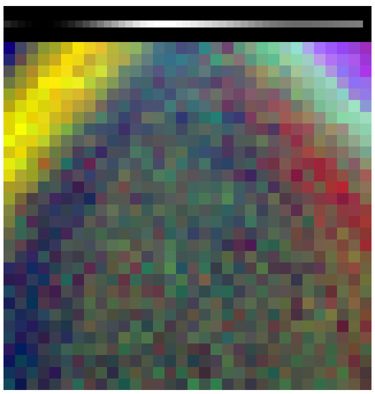
Average Image and Eigenvectors for Two-Lane Road



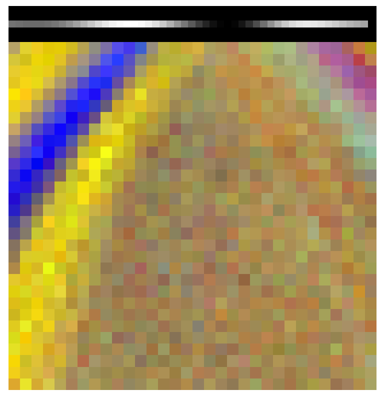
Average Image



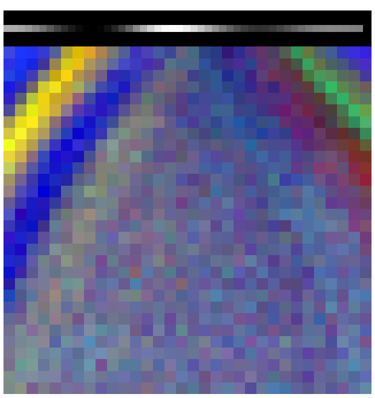
Eigenvector 0 (largest eigenvalue)



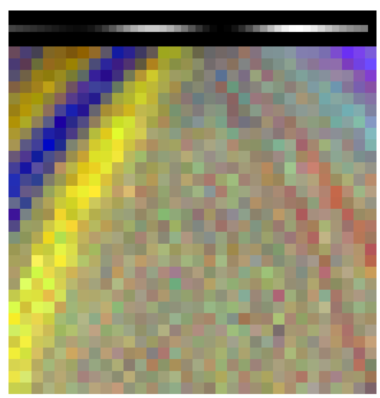
Eigenvector 1



Eigenvector 2



Eigenvector 3



Eigenvector 4(smallest eigenvalue)

Figure 6. These eigenvectors were formed with the Ohta color preprocessing method from a batch of images taken on Schenley Drive. The eigenvectors were then transformed back into RGB space for display purposes. Again, the bands representing the lane markers are the clearest image features.

8.0 References

[1]D. Ballard and C. Brown. *Computer Vision*. Prentice-Hall, 1982.

[2]J. Crisman. *Color Vision for the Detection of Unstructured Roads and Intersections*. Ph.D. Thesis, Electrical and Computer Engineering Department, Carnegie Mellon University, 1990.

[3]T. Jochem and S. Baluja. *Massively Parallel, Adaptive, Color Image Processing for Autonmous Road Following*. Carnegie Mellon Technical Report

[4]Murase and S. Nayar. *Parametric Eigenspace Representation for Visual Learning and Recognition*. Columbia University Technical Report CUCS-054-92.

[5]Y. Ohta. *Knowledge-Based Interpretation of Outdoor Natural Color Scenes*. Pitman, 1985.

[6]D. Pomerleau. Knowledge-based Training of Artificial Neural Networks for Autonomous Robot Driving. In *Robot Learning*, J. Connell and S. Mahadevan (eds.), Kluwer Academic Publishing, 1993.

[7]C. Thorpe. *Machine Learning and Human Interface for the CMU Navlab*. Carnegie Mellon Technical Report

[8]M. Turk, D. Morgenthaler, K. Gremban, and M. Marra. VITS -- A Vision System for Autonomous Land Vehicle Navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 2, 1988.

[9]M. Turk and A. Pentland. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience,* Vol. 3:1, pp. 71-86, 1991.