Visual Tracking for Autonomous Driving

Henry Schneiderman and Marilyn Nashman

National Institute of Standards and Technology
Robot Systems Division
Building 220, Room B127
Gaithersburg, MD 20899
e-mail: hws@cme.nist.gov, nashman@cme.nist.gov

ABSTRACT

This paper describes a visual processing algorithm that supports autonomous driving. The algorithm requires that lane markings be present and attempts to track the lane markings on each of two lane boundaries in the lane of travel. There are three stages of computation: extracting edges, determining which edges correspond to lane markers, and updating geometric models of the lane markers. All processing is confined to the 2-D image plane. No information about the motion of the vehicle is used. This algorithm has been used as part of a complete system to drive an autonomous vehicle, a High Mobility Multipurpose Wheeled Vehicle (HMMWV). Autonomous driving has been demonstrated on both local roads and highways at speeds up to 80 km/h. The algorithm has performed well in the presence of non-ideal road conditions including gaps in the lane markers, sharp curves, shadows, cracks in the pavement, wet roads, rain, dusk, and nighttime driving. The algorithm runs at a sampling rate of 15 Hz and has a worst case processing delay time of 150 milliseconds. Processing is implemented under the NASA/NBS Standard Reference Model for Telerobotic Control System Architecture (NASREM) architecture and runs on a dedicated image processing engine and a VME-based[1] microprocessor system.

1. Introduction

There has been increasing interest in the development of autonomous driving in recent years. Interest has included high-speed driving on highways, urban driving, and navigation through less structured off-road environments. The primary challenge in autonomous driving is the development of perception techniques that are reliable under the extreme variability of outdoor conditions in any of these environments. Roads vary in appearance: some are smooth and well marked while others have cracks and potholes or are not marked at all. Shadows, glare, varying illumination, dirt or foreign matter, other vehicles, rain, and snow also affect road appearance.

Section 2 reviews previous work in vision-based autonomous driving. Section 3 describes this lane marker tracking algorithm. Section 4 presents the hardware and development environment used to implement the algorithm. Section 5 describes the algorithm performance.

2. Review of road perception techniques

Perception for autonomous driving has been approached with a wide variety of vision-based techniques. Most of these approaches fall into one of two categories: region-based statistical classification methods and feature tracking methods. Several other methods have been proposed, most notably neural networks.

2.1. Region-based statistical classification methods

Region-based statistical classification methods [1], [2], [3], [4], [5], [6], [7], [8] have been applied to the road perception problem. These methods share a similar paradigm in that they classify each pixel in the image as either road or non-road. This is done using classical techniques of supervised or unsupervised statistical classification [34]. The road boundaries are then determined by various methods.

In all these methods, pixels are classified on the basis of color. In addition [1] uses a measure of local texture and [8] uses the image coordinates of the pixel. The color at each

1. Certain commercial equipment, instruments, or materials are identified in this paper in order to adequately specify the experimental procedure. Such identification does not imply recommendation or endorsement by NIST, nor does it imply that the materials or equipment identified are necessarily best for the purpose.

pixel is measured in terms of three scalar components: red, green, and blue (RGB). SCARF [1], [2], [3], UNSCARF [3], [8], and FMC [5], [6] directly use the three RGB color components for classification. VITS [4] uses red and blue only. Lin and Chen [7] base classification on the values of hue, saturation, and intensity (HSI). The HSI representation is computed by a non-linear transformation from the measured RGB values. Lin and Chen claim that the HSI representation is better suited for road segmentation.

Classification involves assigning each pixel to one of a number of predefined classes. VITS, FMC, and Lin and Chen classify all pixels as one of two classes: road and non-road. SCARF and UNSCARF, however, represent both road and the non-road by multiple classes. The rationale for having multiple classes is to better represent the variation in outdoor scenes. For example, the differences in color among portions of the road that are wet, shaded, sunny, and patched are significant and are therefore probably better accommodated by separate classes rather than a singleclass.

2.1.1. Supervised classification

Most of the approaches described in [1], [2], [3], [4], [5], [6], [7] use a form of supervised classification. In supervised classification, the statistics of each class are known prior to classification. Each pixel is then assigned to the class which it most closely matches in the statistical sense. SCARF represents each class by its second order statistics (mean and covariance). VITS, FMC, and Lin and Chen divide feature space by a planar boundary between the road class and the non-road class and thereby implicitly assume that the two classes have equal covariance.

It has been reported [1], [2], [3], [4], [5], [6], [7] that fixed class statistics are not consistently reliable for road recognition. For example, the statistics that accurately segment a portion of the road exposed to the sun may not yield a reliable segmentation of a shaded portion of the road. To address this problem, these methods continually recompute class statistics during operation. After each image is classified, the classified pixels are then used to compute the class statistics that will be used for classifying the next image.

The initial class statistics can be established in various ways. The FMC classifier is initialized automatically. The first image is segmented into regions. The pixels in each region are then used to compute the initial class statistics. Other methods require a human operator to label regions in the initial image by hand.

When all pixels have been classified, various techniques can be used to find the boundary which best represents the extent of the road. SCARF uses a Hough method where each pixel votes for all values of the road shape parameters that are consistent with the location of the particular pixel. The parameter set that receives the most votes is chosen as the road model. VITS determines the road boundary by finding edges between the road and non-road regions and using a boundary tracing algorithm to follow the edges. FMC locates a number of candidate boundaries between the road and non-road and selects the boundary that best satisfies a number of geometric constraints such as smoothness, constant road width, and temporal continuity.

SCARF has been used to navigate the Carnegie Mellon University Navigation Laboratory (NAVLAB) along bicycle paths, dirt roads, gravel roads and suburban streets at slow speeds. In particular, SCARF has been successful in locating roads that are obscured by heavy shadows, but it encounters problems when the road is covered by leaves or snow. When this happens, a road class and non-road class become indistinguishable. Problems are also encountered when there are large changes in illumination between successive images. SCARF's update rate is on the order of one second per image depending on the hardware configuration [1].

FMC reports achieving real-time road following on dirt, gravel and paved roads at speeds up to 19 km/h using their vehicle, a specially instrumented armored personnel carrier. They report that their system does not reliably classify areas of road that are in shadow or that contain puddles or patches [6].

VITS was able to successfully navigate the Autonomous Land Vehicle (ALV) over a 4.2 km paved test track at speeds of 10 km/h [4].

The strength of these methods is their generality. They do not require lane markers to be painted on the road and they do not require crisp or smooth road boundaries. A potential weakness of this approach is that there is a cyclical dependency between segmentation and the recomputation of road statistics. If the segmentation is incorrect, the pixels that were misclassified will contaminate class statistics. Inaccurate class statistics will then lead to poorer segmentation. Therefore, the system will probably not recover once it misinterprets an image. This approach also requires continuity in road appearance between successive images. For instance if the sun goes behind a cloud, the color statistics computed from the image of the sunny road probably will not produce a good segmentation on the image in which the sun is hidden.

Additionally, these methods are computationally expensive and therefore are slow. A slow image processing rate limits the speed of driving.

2.1.2. Unsupervised classification

The techniques described in the last section are examples of supervised classification. In supervised classification, the parameters of the classifier are pre-specified for each new data set. In unsupervised classification, the pa-

rameters of the classifier are not pre-specified. Instead categories are formed by grouping together similar data into clusters or classes. A method called UNSCARF [3], [8] uses this technique for road recognition. It groups sets of pixels into regions of similarity and then selects the set of regions whose combined shape best forms a road. To cluster pixels, a variation on the ISODATA clustering algorithm [34] is used. In this method each pixel is first arbitrarily assigned to a class. (The number of classes must be predefined). The mean and covariance of these classes are then computed. Each pixel is then reclassified to the class for which its Mahalanobois distance is minimum. The statistics of each class are then recomputed using the new pixel assignments. This process is repeated until few pixels change class membership.

UNSCARF has been used to successfully navigate the Carnegie Mellon University NAVLAB on unstructured, paved and unpaved roads under various conditions.

2.2. Feature tracking methods

Another set of vision methods for autonomous driving could be described as feature tracking [2], [9], [10], [11], [12], [13], [14], [15], [17], [18], [19], [20], [21], [22], [23], [24], [25]. These methods locate the road on the basis of distinct features, usually the lane markings painted on the road or the boundary between the road and its surroundings. These methods exploit the temporal continuity of the image sequence to locate the desired features, i.e., the search for these features is highly constrained by their location in the previous image. Feature tracking methods also maintain a geometric model of the road that is updated over time. The differences in these methods lie in how the features are detected, how the road is modeled, and how the road model is updated.

The VaMoRs system developed at Universitat der Bundeswehr Munchen, [9], [10], [11], [12], [13], [14] was one of the first systems to demonstrate autonomous driving at high speeds. This method is based on locating road boundaries and road lane markers. These features are detected in the image using edge extraction. In the neighborhood of predicted edge location, the image is correlated with edge masks in a range of orientations about the predicted edge orientation. These extracted edges are then considered to belong to a lane marking or boundary if they form line features with low curvature, are parallel to other line features, and are almost parallel to the viewing direction. In addition to vision, vehicle motion is measured by inertia sensors.

Using the visual and vehicular measurements, the 3-D geometry of the road is reconstructed and updated for each new image. The complete model is comprised of 9 state variables. The road is modeled by 3 state variables representing the horizontal (lateral) contour of the road and 2 state variables representing the vertical (elevation) contour of the road. These contours are approximations to the clothoid model where curvature is modeled as a linear function of arc length. This model also includes 4 state variables representing vehicle steering angle, lateral offset from the center of the lane, heading angle, and slip angle.

The temporal nature of this system is modeled using a state transition equation expressing the evolution of this 9-dimensional state vector over time. A Kalman filter-like approach is used to update the state. This approach accounts for the state space equation and the relationship between the measured quantities and the state. It is not a true Kalman filter because in the state transition equation, the coefficients multiplying the state are functions of the state itself. Because of this non-linearity, such a technique is not optimal and may not be stable. The papers describing this system do not describe how the uncertainty models required for the Kalman filter, the measurement variance and signal covariance, are chosen.

The VaMoRs system has been able to achieve autonomous driving at speeds up to 100 km/h and continuous driving as far as 20 km on the German Autobahn. (This was actually demonstrated with an earlier version of their system that did not account for vertical curvature [12], [13].) They have also had success driving under various lighting and weather conditions and on unmarked cross-country roads.

The YARF system, [2], [15], [16], tracks the lane markers and the shoulders. These features are detected on this basis of known geometry, known color, and edges. The road is modelled as a flat plane and all feature points are used to find a 2nd order polynomial that best describes the path of the road. Fit is computed using both least squares and least median squares. The least median squares approach gives superior performance in the presence of outliers, but is too computationally demanding for real-time implementation. In the least squares approach, the data is rotated such that the residual is approximately normal to the polynomial. YARF has been used to autonomously drive the Carnegie Mellon University NAVLAB at speeds up to 25 km/h on public roads that included high curvature lanes and shadowing from surrounding trees.

The University of Maryland system [17], [18], [19], [20] is based on identifying the road boundary using edge detection. This method begins searching for the road boundary in a small window at the bottom of the image. The search window is chosen based on the location

of the road in the previous image. Based on the location of the road in the window, other windows are placed above this window. The new window is then used to search for road boundaries. This process continues moving from the bottom to the top of the image. The system has achieved autonomous road following over a distance of several hundred meters at a speed of 3 km/h using the Martin Marietta Autonomous Land Vehicle (ALV). The system does not work well in the presence of patchy roads, shadows, and water on the road.

The Toyota system [23] uses edge extraction to locate lane markings. This system has been able to achieve autonomous driving at speeds up to 50 km/h. Successful driving has been achieved on both sunny and cloudy days, and in the presence of shadows. The system is less reliable under more severe lighting and weather conditions including sunrise, sunset, and heavy rain.

The University of Bristol system [21] is based on locating edge points of the road boundary. The edge points are fit to a 2nd order polynomial. All points that are 3σ away from the polynomial are discarded and the least squares computation is repeated. The process of computing points and discarding outliers is repeated until the variance in the estimate stops decreasing. This method has been used to autonomously drive a small electric vehicle on paths on the university grounds.

Other researchers have worked with feature tracking approaches for driving in simulation. These include [22], [25].

The advantage of the feature tracking algorithms is that they require less computation and are therefore able to achieve image processing update rates capable of supporting high speed driving. The disadvantage is that they require specific features of the road infrastructure to be present such as clearly visible lane markers. When these features are less prominent due to wear, or obscured because of weather or lighting, these techniques become unreliable.

2.3. Other approaches

A neural network-based approach ALVINN [29], [30] has been used for autonomous driving. The input to the network is a reduced resolution (30 X 32 pixels) processed image. The network generates a steering angle as an output.It is trained by using backpropagation while a human is driving. To obtain a training sequence that includes a large variety of driving situations, several techniques are used. ALVINN also includes a training technique whereby structured noise is added to image regions where the network may draw an incorrect correlation. For example, in a short training sequence, the network may draw an undesirable correlation between the amount of grass in view and the appropriate steering angle. Without the addition of structured noise, the network may fail when the grass becomes obscured by a guardrail. ALVINN has successfully driven the Carnegie Mellon University NAVLAB for a continuous run of 34 km at speeds of up to 90 km/h. ALVINN has been successfully trained for highways, unmarked rural roads, and cross-country roads.

Other approaches proposed for road perception include image flow [28], morphological image processing [26], and combined region and boundary extraction [27].

3. Lane marker tracking algorithm

This paper describes a feature tracking approach to autonomous driving developed at NIST. The approach differs from the previous approaches in the methods used for updating the road model and in the road model representation.

The method used for updating the road model differs from others in the manner in which data is combined temporally. In the recursive estimation formulation for updating the model (see section 3.6), each image carries a weight based on the confidence in that image data. Under this formulation, an image in which the lane markers are clearly visible will carry more weight than an image in which lane markers are less visible. This produces graceful behavior when lane markers are momentarily absent or obscured. Other feature trackers do not consider image data confidence when combining image data temporally.

In this method, we represent the road by a 2D model in the image plane as opposed to a full 3D model. With this representation, the model is directly estimated from the image without any intervening geometric transformations. Other approaches transform the detected features from 2D to 3D and update a 3D road model. The 3D road model is then backprojected into 2D and used to search for features in the next image. The problem with this approach is that these transformations are never exact. They produce errors because they depend on camera calibration and approximations (the small angle approximation, the flat road assumption, linearization of a non-linear models). By representing the model in 2D, these sources of error are avoided.

The 2D representation also simplifies the geometric road model update computation. This representation allows both spatial and temporal information to be combined in one simple recursive estimation formulation instead of separate estimates of spatial and temporal model parameters.

3.1. Overview

The road following algorithm involves three successive

stages of computation:

1) Edge extraction - Extracting edge point position and orientation.

2) Data association - Determining which edges most likely correspond with each lane marker.

3) Model update - Updating the lane marker models.

This sequence of operations is repeated for each new image. Video imagery from a camera mounted above the cab of the vehicle provides the input to Stage (1) (Figure 1). Stages (2) and (3) interact with a geometric model of the road. Stage (2) attempts to associate the extracted edge points obtained from stage (1) with the models. Stage (3) updates the models using the edge points associated to each lane marker.

In this section, the road feature tracking algorithm is described in detail. Section 3.2 describes the geometric representation of the lane markers. Section 3.3 describes how the model is initialized to a road scene. Section 3.4 describes the edge extraction algorithm. Section 3.5 describes the data association algorithm. Section 3.6 describes the method for updating the lane marker models.

3.2. Lane marker models

Both the left and right lane markings in the lane of travel are modeled. These markings correspond to the white or yellow lines painted on the road. These lines are either solid or striped (Figure 2).

Each of these lane boundaries is modeled by a second order polynomial in the image plane:

$$x = a_1 + a_2 y + a_3 y^2 \qquad (1)$$

The parameters, a_1, a_2, a_3, govern the shape and position of the lane marker model. The endpoints of each lane marker model are given by the intersection of the model equation with the boundary of the window of interest (see section 3.4).

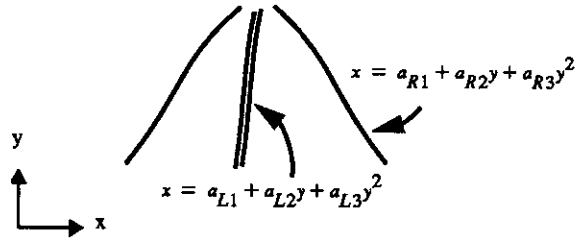

Figure 2. Lane marker models

3.3. Initial conditions

The algorithm requires an initially approximate model of the lane markers before tracking can begin. This initial model is established by a teleoperator who manually positions the models to align them with the appearance of the lane markers in the image. On the visual display, the lane marker models are represented to the teleoperator as graphic overlays on the live video image. In this way, the teleoperator establishes the initial values of the parameters a_1, a_2, a_3 in equation (1) for both lane marker models.

3.4. Edge extraction

In the first processing step, edge extraction is performed on the input scene (stage (1) in Figure 1). For every point in the image, edge magnitude and edge orientation are computed using a two-dimensional *3 X 3* spatial gradient operator. The direction, θ, of each point in the image is defined to be perpendicular to the direction of the gradient of the intensity function $f(x,y)$ at that point:

$$\theta = \operatorname{atan}\frac{\nabla f_x(x,y)}{\nabla f_y(x,y)} + \frac{\pi}{2} \qquad (2)$$

The magnitude of each edge pixel, *mag*, is given by:

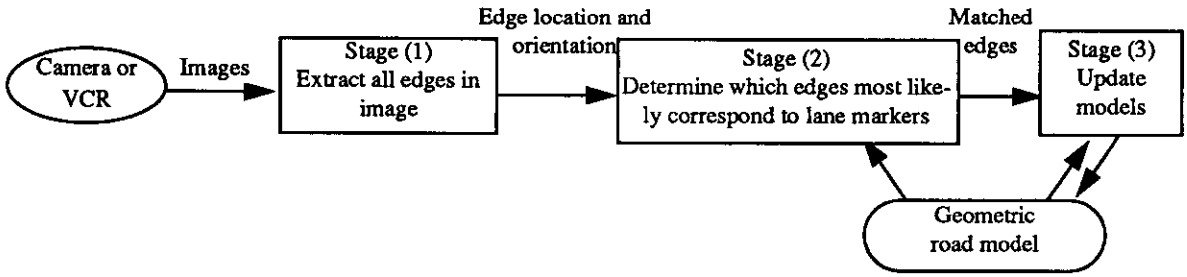

Figure 1. Processing overview

$$mag = \sqrt{(\nabla f_x(x,y))^2 + (\nabla f_y(x,y))^2} \qquad (3)$$

Using a non-maximum suppression algorithm, those edge pixels whose magnitude is greatest in the direction across the edge are selected as edge points. A description of the non-maximum suppression edge extraction algorithm can be found in [33]. A binary edge image is produced by thresholding the edge points. The threshold level is set to a value which removes weaker edges. A threshold value of 8 in a grey level range of 0 to 255 proved to be effective. The output from this processing stage consists of a list of the image coordinates of all edge points above the threshold value and the orientation of these points. It should be noted, at this stage of operation, no effort is made to distinguish lane marker edges from other edges present in the input image.

To reduce the amount of data processed by the data association algorithm (stage (2) in Figure 1), all edges that fall outside a window of interest are excluded. This window of interest is chosen to include the entire portion of the visible road but to exclude, as much as possible, the rest of the image (e.g. the hood of the vehicle, trees, grassy shoulders, houses, etc.). Figure 3a is a typical image of a road viewed from a camera mounted on a vehicle. Figure 3b is a window of interest. Figure 3c represents the results of masking the original road scene with the window of interest. During execution, the lateral position of the window of interest shifts in order to keep it centered on the road. In addition to centering, the shape of the window of interest changes as a function of the current road curvature. Currently, seven masks are used: one mask representing zero road curvature (Figure 3b), three masks representing increasing road curvature to the left, and three masks representing increasing road curvature to the right. All masks are generated off-line but are instantiated in real-time during actual image processing. The mask selection algorithm changes masks when one of the lane marker models intersects either of the vertical boundaries of the current mask. For example, if a lane marker intersects the left boundary, the mask giving the next larger increment of curvature to the left is chosen.

Prior to processing, the image is enhanced by a yellow filter in front of the camera lens. The filter is designed for spectral transmission of wavelengths from 510 nanometers (nm) into the infrared. The effect is to intensify the contrast of the yellow and white markers against the road.

3.5. Data association

The raw edges in each image are produced by various visual entities including lane markers, shadows, pot holes, cracks, and other vehicles. A data association algorithm is used to determine which of these raw edges are likely to be associated with each lane marker and to discard those edges that do not seem to be associated with either lane marker.

The data algorithm compares each edge pixel to the model of each lane marker. An edge pixel must satisfy two criteria to be associated with a lane marker. The first criterion is two-dimensional spatial proximity of the edge point to the model. The second criterion is similarity of direction of the edge point with the angular orientation of the model. Using these criteria, it is possible but unlikely for an edge point to be associated to both lane markers.

To facilitate this process, the polynomials representing each lane marker are each approximated by a set of consecutive line segments. This is achieved using a simplified version of the iterative endpoints algorithm [34]. The conglomerate of these lines is used as the model in the data association procedure.

The first step in this data association procedure compares the edge direction of the candidate edge point with the angular direction of each of these model lines:

$$|\theta_{model} - \theta_{data}| < \delta \qquad (4)$$

For each model line for which the angular criteria is satisfied, the distance d is computed between the point at image coordinate (x_i, y_i) and the model line:

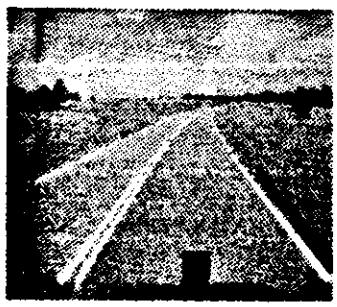

Figure 3a. Road scene

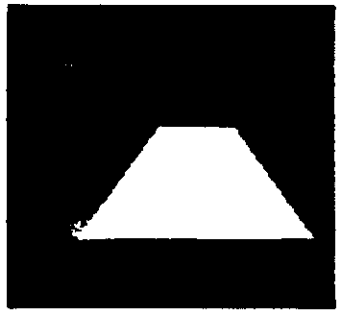

Figure 3b. Window of interest

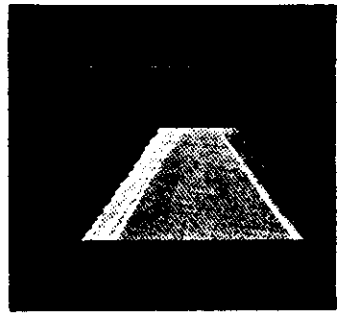

Figure 3c. Window of interest applied to road scene

$$d = \frac{A_k x_i + B_k y_i + C_k}{(A_k^2 + B_k^2)^{1/2}} \quad (5)$$

Where $(A_k x + B_k y + C_k = 0)$ is the equation for the k^{th} line in the model.

If the minimum of these distances is less than a distance threshold, ζ, the edge is associated with that lane marker.

3.6. Lane marker model update

Each of the two polynomial lane marker models is updated independently (except for the case described in 3.6.3). The parameters of each model a_1, a_2, a_3 in equation (1), are updated by an exponentially weighted recursive least squares computation (see [35]) using the edge points that have been associated to that lane marker.

It should be emphasized that in this estimation method, the estimated model parameters are based not just on the current image, but on data acquired over the entire sequence of previous images. The edge data from any one image alone may not be sufficient to obtain an accurate model of the lane markers. This data may be contaminated by noise or incorrect data associations, or the edges due to actual lane markers may be too weak to be detected. In any case, the estimate of a lane marker model can be improved by using data over a sequence of images. It can be shown by probabilistic arguments that if a measurement consists of a sum of a stationary signal and randomly distributed zero mean noise, the estimate of the signal will improve (i.e. the variance in the estimate will decrease) as more measurements are averaged [31], [32].

For feature detection in a static scene, the assumption of randomly distributed noise is somewhat problematic. Since the feature and its surroundings are in a fixed spatial relationship, any source of noise in the surroundings (e.g. a shadow) will bias the estimate regardless of how many images are used. However, for road following, this assumption of random noise holds to a much greater degree. In road following, the immediate surroundings of the lane markers (as viewed in the image) are constantly changing because of the motion of the vehicle. Since the immediate surroundings are different for each image, a better approximation to random noise, symmetrically distributed about the signal, will be achieved by using more images. Noise caused by shadows and cracks in the road also tends to be randomly distributed. However, there are exceptions such as skid marks, mud tracks, and shadows of power lines.

The success of exponentially weighted recursive least squares also is based on the assumption that the appearance of the road changes gradually over a sequence of images (the stationary signal assumption). There are, however, limits at which the assumption of continuity fails. Therefore, a compromise must be achieved between robustness and responsiveness by the relative weighting of new data with respect to older data. If new data is weighted relatively heavily, the algorithm will be very responsive to changes in the road. The algorithm will also be more susceptible to the ill-effects of outliers and sparse data. On the other hand, if new data is weighted less heavily, the algorithm will be more robust in the presence of outliers, but more inert in responding to actual changes in the image of the road.

In exponentially weighted recursive least squares, the temporal weighting of data is controlled by specifying the value of the *exponential weighting factor*, λ (also called the *forgetting factor*). The weight assigned to data from each image is:

$$\lambda^{n-m} \quad (6)$$

$0.0 < \lambda < 1.0$
n is the current time
m is the time the image was sampled

For example, if $\lambda = 0.5$, all edge points in the current image, $m=n$, have a weight of 1.0. All edge points in the image read at time $m = n - 1$ have a weight of 0.5; edge points from time $m = n - 2$ have a weight of 0.25, etc. Values of λ anywhere in the range $0.5 < \lambda < 0.75$ produced acceptable tracking.

3.6.1. Formulation of the least square problem

The model parameters are determined such that a least squares residual is minimized. To illustrate this for a simple case involving one data set, the values of a_1, a_2, a_3 in equation (1) are determined such that J_B minimized:

$$J_B = \sum_{i=1}^{N} [x_i - (a_1 + a_2 y_i + a_3 y_i^2)]^2 \quad (7)$$

N - Number of data points (x_i, y_i).

Graphically this corresponds to minimizing the sum of the squared horizontal distances between the model and the data points (Figure 4):

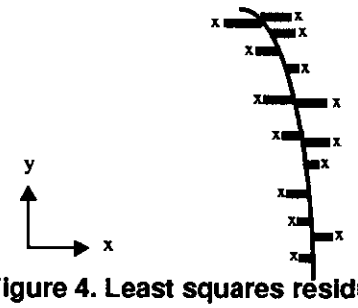

Figure 4. Least squares residual in x

The reason for minimizing the least squares residual in the x coordinate as opposed to the y coordinate is that the lane markers when viewed from an automobile centered on the road, are more nearly perpendicular to the x axis than the y axis.

In the formulation of the road model update problem, the data set includes the edge data from the entire previous image sequence and grows as new images are acquired. As new data is acquired, the older images are weighted by increasing powers of the exponential weighting factor (equation (6)) thereby giving them increasingly less weight. Therefore, after each is image is acquired, a_1, a_2, a_3, must be recomputed such that J_R is minimized:

$$J_R = \sum_{i=1}^{N_j} [x_{j,i} - (a_1 + a_2 y_{j,i} + a_3 y_{j,i}^2)]^2 + \lambda \sum_{i=1}^{N_{j-1}} [x_{j-1,i} - (a_1 + a_2 y_{j-1,i} + a_3 y_{j-1,i}^2)]^2 + \lambda^2 \sum_{i=1}^{N_{j-2}} [x_{j-2,i} - (a_1 + a_2 y_{j-2,i} + a_3 y_{j-2,i}^2)]^2 + \ldots \quad (8)$$

j - Time at which image was sampled
N_j - Number of matched edges points in image j

Each summation represents the data from one image. In addition to exponential weighting, the weight of each image j is also implicitly a function of the number of edge points, N_j, used from the j^{th} image. An image in which many edge points are associated to the lane marker will carry more weight than an image with few associated edge points. Therefore, if a lane marker momentarily disappears, few edge points will be associated to the model and the estimate will not be greatly perturbed. Also, since the variance of a least squares estimate decreases as the number of data points increases (see [31], [32]) more weight is given to an image in which there is a higher confidence.

3.6.2. Solution of the least squares problem by a recursive method

To efficiently solve equation (8) for a_1, a_2, a_3 such that the residual, J_R, is minimized, a modification of the square root information filter (SRIF) algorithm [36] is used. The SRIF provides an efficient, numerically stable, closed form solution to the least squares problem. It is also a recursive algorithm, i.e. the model is updated as new data becomes available without having to explicitly store older data. The algorithm also has the advantage that it is "recursive in batches" i.e. it can efficiently combine groups of measurements at once. Such a computational arrangement is useful for the road tracking problem and computer vision problems, in general, since each image yields a batch of data points.

The SRIF is based on solving the recursive least squares problem using the square root method [36], [37]. This square root technique is first illustrated for the ordinary least squares problem of determining a_1, a_2, a_3, such that J_B is minimized for one data set (equation (7)). Rewriting equation (7) in matrix-vector form gives:

$$J_B = (b - Ax)^T (b - Ax) = \| b - Ax \|^2 \quad (9)$$

Where

$$b = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_N \end{bmatrix} \qquad A = \begin{bmatrix} 1 & y_1 & y_1^2 \\ 1 & y_2 & y_2^2 \\ . & . & . \\ 1 & y_N & y_N^2 \end{bmatrix} \qquad x = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Where A is $Nx3$, $N>3$, rank = 3.

The Euclidean norm, J_B, of the vector (b - Ax) will not change if the vector is multiplied by an orthogonal matrix T, where T is size NxN. Therefore, x can be equivalently determined by minimizing:

$$J_B = \| TAx - Tb \|^2 \quad (10)$$

An orthogonal matrix T is chosen such that the matrix product TA is equal to R, where R is partitioned into a $3x3$ upper triangular matrix, U, and a $(N-3)x3$ null matrix consisting of the remaining rows:

$$TA = R = \begin{bmatrix} U \\ 0 \end{bmatrix} \quad (11)$$

To find such a T, the QR factorization of A can be computed:

$$A = QR = Q \begin{bmatrix} U \\ 0 \end{bmatrix} \quad (12)$$

Where Q is orthogonal and R are the same as above.

Multiplying both sides of the equation by the transpose of Q gives:

$$Q^T A = R = \begin{bmatrix} U \\ 0 \end{bmatrix} \quad (13)$$

Therefore, comparing equations (11) and (13), T is given by the transpose of Q, computed from the QR factorization of A.

The QR factorization exists for any full rank matrix, A.

A number of methods exist for computing this factorization including Gram-Schmitt, Modified Gram-Schmitt, Fast Givens Rotations, Householder Reflections, and the QR algorithm [36], [37], [38], [39]. The Householder method was used for the purposes of this problem. It offers a combination of simplicity, numerical stability and computational efficiency.

Once T is found, the residual can be rewritten:

$$J_B = (Tb - TAx)^T (Tb - TAx) = \left(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} - \begin{bmatrix} U \\ 0 \end{bmatrix} x \right)^T \left(\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} - \begin{bmatrix} U \\ 0 \end{bmatrix} x \right) \quad (14)$$

$$J_B = [z_1 - Ux]^T [z_1 - Ux] + z_2^T z_2 \quad (15)$$

Where the product Tb is partitioned into a vector z_1 consisting of the first 3 rows and the vector z_2 consisting of the remaining N-3 rows:

$$Tb = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

Since only the first term in equation (15) is a function of x, only this term has to be minimized. This term is non-negative and x can be determined such that it is equal to zero, or equivalently:

$$z_1 = Ux \quad (16)$$

This is an exact set of linear equations (3 equations and 3 unknowns) that can be solved uniquely. Moreover, back-substitution (see [39], [40]) can be used to solve for x since U is an upper triangular matrix. This x then gives the solution to the ordinary least squares problem of minimizing equation (7).

It should be noted that this square root method is one of many methods that can be used to solve the ordinary least squares problem. Most readers are probably more familiar with the process of forming and solving the normal equations:

$$A^T b = A^T A x \quad (17)$$

However, as mentioned earlier, the square root method is chosen because it allows the recursive processing of batches. It also has better accuracy and numerical stability under finite precision arithmetic than normal equations methods since it does not involve forming the squares in equation (17).

The square root information filter (SRIF) extends the square root method to recursively solve the least squares problem given by equation (8). Assume that the least squares pair given by z_1 and U in equation (16) exists for data from the previous sequence of images. To update this solution to account for data from a new image, such as A and b given by equation (9), a matrix vector pair is constructed by appending U and z_1 with additional rows corresponding to a new set of data (A, b), respectively:

$$y = \begin{bmatrix} z_1 \\ b \end{bmatrix} \qquad M = \begin{bmatrix} U \\ A \end{bmatrix} \quad (18)$$

This gives a composite data set combining the data from the entire previous image sequence (z_1, U) with the new data (A, b). Combining data in this fashion may appear non-intuitive, since (A, b) represent raw data and (U,z_1) do not directly represent raw data. However, (U,z_1) were obtained by multiplying raw data by a series of orthogonal matrices. (y, M) can then be thought of as the product of raw data with an orthogonal matrix, where the block component multiplying A is an identity matrix. From equation (10), we know that the least squares problem is unaltered if the data is multiplied by an orthogonal transformation.

Using the square root method, x is then solved such the least squares residual, J, is minimized:.

$$J = (y - Mx)^T (y - Mx) = \| y - Mx \| \quad (19)$$

This will give the new updated solution for x. The new values of z_1 and U that are generated by this process are then used to form another new pair y and M, equation (18), when the next image is acquired. This whole process is called the square root information filter. This name reflects the fact that the algorithm involves updating U, which is proportional to a square-root of x's information matrix when the errors (y - Mx) are independent and identically distributed. The information matrix is defined as the inverse of the covariance matrix, Σ, of the state, x; that is, if x is thought of as a random vector, then:

$$E((x - \bar{x})(x - \bar{x})^T) = \Sigma = (U^T U)^{-1} \quad (20)$$

To modify this algorithm to include exponential weighting, y and M are to be constructed by the alternate forms, in place of equation (18):

$$y = \begin{bmatrix} z_1 \sqrt{\lambda} \\ b \end{bmatrix} \qquad M = \begin{bmatrix} U \sqrt{\lambda} \\ A \end{bmatrix} \quad (21)$$

where λ is the *exponential weighting factor* defined in equation (6).

3.6.3. Road width constraint

Lane markers are physically constrained by constant road width in the ground plane. This translates to a fixed separation in the image plane if elevation changes as a linear function of distance along the road.

If sufficient data is matched to both lane markers, the

two lane markers are treated independently. However, a road width constraint is used when the lane marker data is sparse for one lane marker and strong for the other lane marker. This constraint is designed to handle the situation where one lane marker momentarily disappears. Currently a road width constraint is applied when the number of edge points associated to the weaker lane marker is under a predefined threshold and the stronger lane marker exceeds the threshold. The threshold is currently set to 40 points.

Road width is modeled as a second order polynomial in the image plane:

$$x_w = a_{1w} + a_{2w}y + a_{3w}y^2 \tag{22}$$

Each coefficient of this polynomial is computed from the difference of the corresponding coefficients of the two lane marker models. This difference is then averaged over time with exponential decay. For example, for the first coefficient this gives:

$$a_{1w}(j) = \frac{a_{1R}(j) - a_{1L}(j) + Na_{1w}(j-1)}{1+N} \tag{23}$$

j - Time at which image was sampled
N - Decay factor, currently N=20

These coefficients are recomputed for each new image.

The road width constraint involves using the road width model to compute a lane marker model by adding this offset to the location of the other lane marker model. The computed lane marker model is then weighted and combined with the edge data usually used for computing the lane marker model update. The weight this computed lane marker carries is a function of the number of edge points associated to lane marker versus the number of points associated to the other lane marker.

4. Hardware and developmental testbed

To run actual autonomous driving experiments requires a complete autonomous navigation system which includes a perception system, a steering/control system, and a robotic vehicle. Such a system has been developed at the National Institute of Standards and Technology (NIST) using an Army High Mobility Multipurpose Wheeled Vehicle (HMMWV). The HMMWV and steering/control system are described in [47] and [48].

The development environment for the vision system consists of a Sun SPARCstation 2, a Pipelined Image Processing Engine (PIPE), a VME-based multiprocessor system. Figure 5 shows the allocation of processing across hardware. In this figure, the large gray rectangles represent distinct software modules. Each of these modules is labelled by its functionality (SP = sensory processing, WM = world modeling) and level within the NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM) [41]. The system described in this paper is contained in a larger, multi-purpose implementation of NASREM, [42], [43], [44], [45]. Also, a complete control system architecture proposed for intelligent vehicles is found in [46].

A CCD video camera is mounted above the cab of the HMMWV. This gives a view of the road similar to that of a driver but from a higher perspective.

The video input is read into PIPE. The incoming images are digitized to provide 8-bit grayscale images that are 242x256 pixels in size. Edge extraction is performed on the images. The Iconic-to-Symbolic Mapper (ISMAP) stage of PIPE [50] then converts information from an image format to a symbolic list and is used to store the binary edge image as a list of pixel positions. In addition, the corresponding edge direction values are stored in the ISMAP iconic buffer where they are mapped onto the memory of one of the microprocessors via a specialized PIPE-VME interface board. The edge extraction and symbolic mapping operations are pipelined. They are indicated by black parallelograms in Figure 5.

The remaining processing is divided among microprocessors in the VME backplane. Most computations -- communication with the PIPE, data association, updating the model, and steering -- are pipelined. The model updates for each lane marker are computed in parallel on separate processors. All inter-processor communication is done through semaphored global memory. For a detailed description of our software engineering practices refer to [49].

The steering process computes a steering angle that will guide the vehicle down the center of the lane of travel. The steering algorithm used to guide the vehicle is called pure pursuit and is described in [48].

The display process provides a graphic overlay of the window of interest, the geometric models of the lane boundaries and the computed lane center on the live video image. The graphic overlay is used for debugging purposes and to provide a qualitative measure of performance. A Matrox VIP 1024 frame grabber (not shown in Figure 5) is used to implement the graphic overlay on the video signal.

The logging process simply logs data during operation that can be later analyzed off-line.

All program development for the VME-based multiprocessor system is done on a Sun SPARCstation 2. All code on this system is written in the Ada programming language. Program development for PIPE is done on a personal computer using the PIPE graphical programming language, ASPIPE [50].

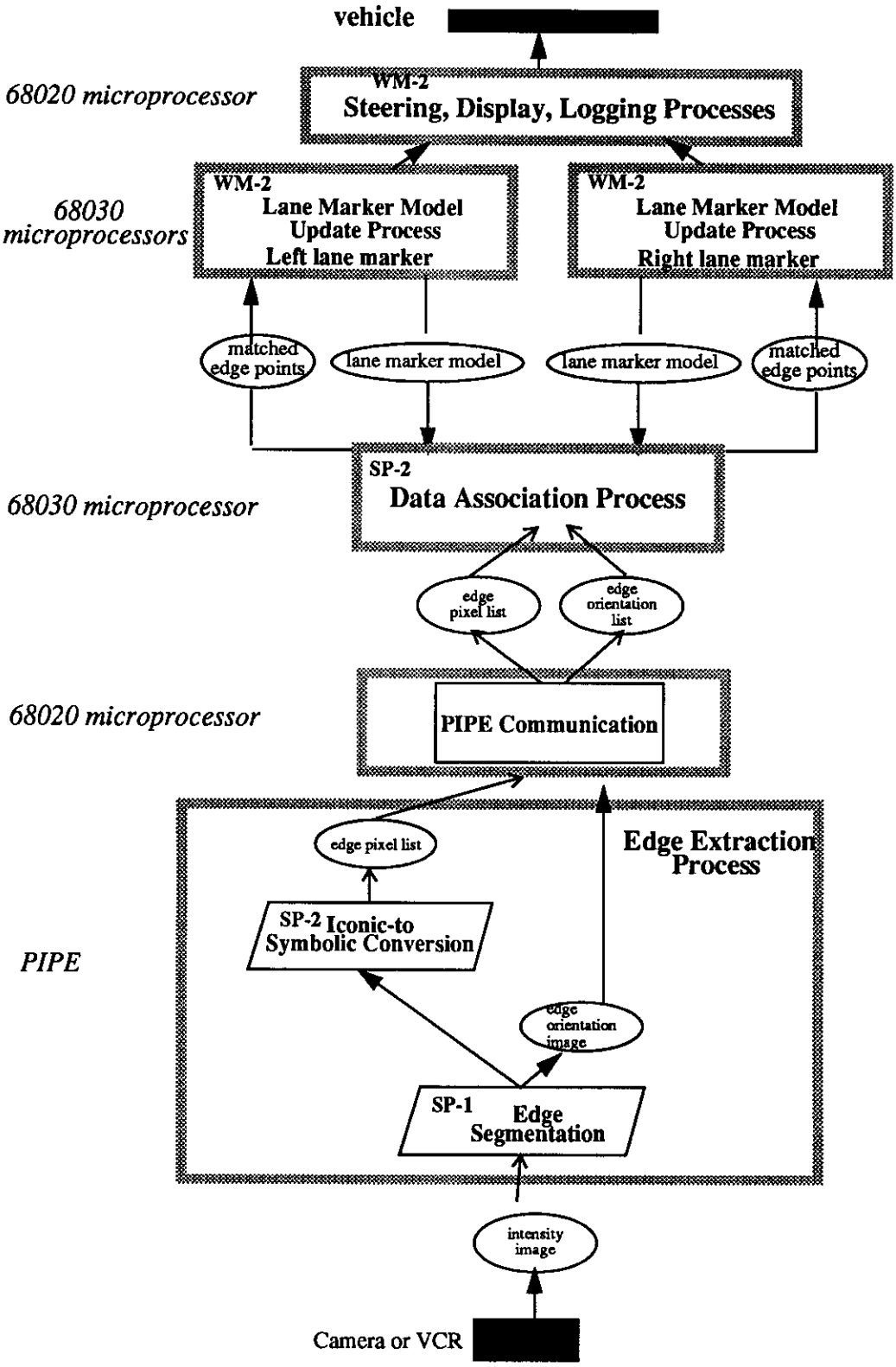

Figure 5. Distribution of processing in hardware

5. System evaluation

This algorithm has been tested and used to drive the HMMWV under closed loop control on several roads.

The algorithm has been successfully and reliably used to guide the vehicle under full closed loop control on roads on the NIST campus. This included testing during rain with wet roads, nighttime driving with headlights, and driving at dusk into the sunlight. Top speeds of 80 km/h were demonstrated. The lane markings consisted of a standard solid double yellow line down the middle of the road and a solid white line separating the road from the shoulder.

On a four lane road, Great Seneca Highway in Gaithersburg, Maryland, the algorithm has been used to successfully control the vehicle at top speeds of 80 km/h. The algorithm has guided the vehicle on both lanes of travel. The lanes of travel are separated by a dashed line. In both lanes there is a solid white line separating the lane from the shoulder. Autonomous driving was maintained over several kilometers (intersections limited the length of these stretches) that included moderate curves and shadows from trees. Autonomous driving sometimes failed on one portion of road where the pavement abruptly changed from dark asphalt to light cement for an overpass. The lane markings did not provide enough contrast to be detected on the cement pavement. However, driving was maintained through several significant gaps (6 - 7 meters) in the lane marking for small intersections and under an overpass.

The algorithm has also been used to autonomously drive the vehicle around the Montgomery County Police Test Track. The test track is approximately 4 kilometers long and is marked by a double yellow line down the middle and a solid white line separating the lanes from the shoulder. The track is designed for police training and contains many challenges to the driver such as sharp curves, poor banking on curves, poor visibility, and small steep hills. Also, the pavement is rather old and exhibits many cracks and discolorations.

The algorithm has also been tested using video taped road scenes recorded from the camera mounted on the vehicle. This provided a method for performing tests without the vehicle, thereby allowing safe testing of the vision algorithm. Tracking was maintained on video tapes of roads with sharp curves, hills and moderate shadows. However, on one portion of video taped road, tracking was temporarily lost when the vehicle travelled through a sharply curved hilly portion of road that was shadowed by a heavily wooded area. Tracking was maintained in typical traffic situations: on-coming traffic, passing vehicles, and while traveling behind other vehicles.

5.1. Timing

The update rate of the system is 15 Hz and the worst case computational delay (between image capture and computation of steering) is 150 milliseconds (ms). Edge extraction required 66.7 ms.The number of edge points extracted varies from scene to scene and the processing times for the algorithms in stages (2) and (3) (Figure 1) will vary depending on the number of data points present. For a representative road scene containing approximately 300 edge points, the edge matching is performed in 21 ms and the road model update is performed in 51 ms. The steering process requires less than 1 ms computation time.

6. Conclusion and future work

An algorithm has been described that robustly tracks road lane markers. It is assumed that the lane markers are visible with either solid, double, or dashed lines. All visual processing is done in two dimensional image coordinates. Processing is performed in sequential stages: extracting edges, associating edge points to the lane markers, and updating models of the lane markers. An exponentially weighted recursive least squares computation used to update the road model operates in both a spatial and temporal domain. The system update rate is 15 Hz.

Although this system performed very well under limited testing, its limitations under severe conditions were evident. The challenge in future work is to develop algorithms of increasing reliability and to strive toward the goal of robustness under all possible road conditions. This will involve making use of other visual cues for redundancy, such as road color, road texture, and range.

Acknowledgments

We would like to give special thanks to Thomas Wheatley, Karl Murphy, Harry Scott, Steve Legowik, and Chuck Giauque for their help. We would also like to thank all members of the Robot Systems Division at the National Institute of Standards and Technology (NIST) for their helpful comments.

References

[1] C. Thorpe, M. Hebert, T. Kanade, S. Shafer. "Vision and Navigation for the Carnegie-Mellon Navlab." IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 10, No. 3. May, 1988.

[2] C. Thorpe, M. Hebert, T. Kanade, S. Shafer. "Toward Autonomous Driving: The CMU Navlab." IEEE Expert. August, 1991.

[3] J. Crisman, C. Thorpe. "Color Vision for Road Following." In *Vision and Navigation: The Carnegie Mellon Navlab.* Kluwer.

Norwell, MA. 1990.

[4] 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. 3. May, 1988.

[5] D. Kuan, U. K. Sharma. "Model Based Geometric Reasoning for Autonomous Road Following." IEEE conference on Robotics and Automation. 1987. pp 416 - 423.

[6] D. Kuan, G. Phipps, and A. Hsueh. "Autonomous Robotic Vehicle Road Following." IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 10, No. 5. September, 1988.

[7] Xueyin Lin and Shaoyun Chen. "Color Image Segmentation Using Modified HSI System for Road Following." IEEE Conference on Robotics and Automation. 1991. pp 1998 - 2003.

[8] Jill D. Crisman and Charles E. Thorpe. "UNSCARF, A Color Vision System for the Detection of Unstructured Roads." IEEE Conference on Robotics and Automation. 1991.

[9] E.D. Dickmanns and V. Graefe. "Dynamic Monocular Machine Vision." Machine Vision Applications. Vol. 1, pp 223 - 240. 1988.

[10] E.D. Dickmanns and V. Graefe. "Applications of Dynamic Monocular Machine Vision." Machine Vision Applications. Vol. 1, pp 241-261. 1988.

[11] E. D. Dickmann, B. Mysliwetz, T. Christians. "An Integrated Spatio-Temporal Approach to Automatic Visual Guidance of Autonomous Vehicles." IEEE Transactions on Systems, Man, and Cybernetics. Vol. 20, No. 6. pp. 1273-1284, 1990.

[12] E. Dickmanns, A. Zapp. "A Curvature-based Scheme for Improving Road Vehicle Guidance by Computer Vision." SPIE Vol. 727. Mobile Robots, 1986.

[13] B. Mysliwetz, E. Dickmanns. "Distributed Scene Analysis for Autonomous Road Vehicle Guidance." SPIE Vol. 852. Mobile Robots II, 1987.

[14] E. D. Dickmanns and B. D. Mysliwetz. "Recursive 3-D Road and Relative Ego-State Recognition." IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. 14, No. 2. Feb., 1992.

[15] K. Kluge and C. Thorpe. "Explicit Models for Road Following." In *Vision and Navigation: The Carnegie Mellon Navlab.* Kluwer. Norwell, MA. 1990. Also in IEEE Conference on Robotics and Automation, 1989.

[16] Karl Kluge and Charles Thorpe. "Representation and Recovery of Road Geometry in YARF." Proceedings of the Intelligent Vehicles '92. Detroit. pp 114 - .

[17] L. Davis and T. Kushner. "Road Boundary Detection for Autonomous Vehicle Navigation." SPIE Vol. 579. Intelligent Robots and Computer Vision. (1985).

[18] Allen Waxman, J. LeMoigne, B. Srinivasan. "Visual Navigation of Roadways." 1985 IEEE conf. on Robotics and Automation. pp 862 - .

[19] Uma Kant Sharma and Larry S. Davis. "Road Boundary Detection in Range Imagery for an Autonomous Robot." IEEE J. of R and A. Oct. 88. Vol 4, No. 5. pp 515 - .

[20] A. Waxman, J. LeMoigne, L. Davis, B. Srinivasan, T. Kushner, E. Liang, T. Siddalingiah. "A Visual Navigation System for Autonomous Land Vehicles." IEEE Journal of Robotics and Automation. Vol RA-3, No. 2. April, 1987.

[21] R. Lotufo, A. Morgan, E. Dagless, D. Milford, J. Morrissey, B. Thomas. "Real-time Road Edge Following for Mobile Robot Navigation." Electronics and Communications Engineering Journal. Vol. 2, No. 1. February, 1990.

[22] S. Kenue. "Lanelok: Detection of Lane Boundaries and Vehicle Tracking Using Image-Processing Techniques - Part I: Hough-Transform, Region-Tracing and Correlation Algorithms and Part II: Template Matching Algorithms." SPIE Vol. 1195. Mobile Robots IV, 1989.

[23] Norto Komoda, et. al (TOYOTA). "Cooperative Vehicle/Highway System for Automated Vehicle." Proceedings of the Intelligent Vehicles '92. Detroit. pp113 - .

[24] Sadayuki Tsugawa, et. al. "An Automobile with Artificial Intelligence." IJCAI, 1979.

[25] Akihiro Suzuki, et. al. "Lane Recognition System for Guiding of Autonomous Vehicle." Proceedings of the Intelligent Vehicles '92. Detroit. pp 196 - .

[26] Xuan Yu, Serge Beucher, and Michel Bilodeau. "Road Tracking, Lane Segmentation, and Obstacle Recognition by Mathematical Morphology." Proceedings of the Intelligent Vehicles '92. Detroit. pp 166- .

[27] Gareth Funka-Lea, Ruzena Bajcsy. "Vision for Vehicle Guidance Using Two Road Cues." Proceedings of the Intelligent Vehicles '92. Detroit. pp 126 - .

[28] D. Raviv and M. Herman. "A New Approach to Vision and Control for Road Following." Proceedings of the IEEE Workshop on Visual Motion. Princeton, NJ. October, 1991.

[29] D. Pomerleau. "Neural Network Based Autonomous Navigation." In *Vision and Navigation: The Carnegie Mellon Navlab.* Kluwer. Norwell, MA. 1990.

[30] D. Pomerleau. "Progress in Neural Network-based Vision for Autonomous Robot Driving." Intelligent Vehicles '92 Symposium. pp. 391-396.

[31] C. Helstrom. *Probability and Stochastic Processes for Engineers.* Macmillan Publishing Company. New York, 1984.

[32] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* 2nd Ed. Magraw-Hill. New York, 1984.

[33] A. Rosenfeld, A. Kak, *Digital Picture Processing*, Volume 2, Second Edition. Academic Press, 1982.

[34] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons. New York, 1973.

[35] G. Goodwin, K. Sin. *Adaptive Filtering, Prediction and Control.* Prentice-Hall. Englewood Cliffs, New Jersey, 1984.

[36] G. Bierman. *Factorization Methods for Discrete Sequential Estimation.* Academic Press. New York, 1977.

[37] C. Lawson and R. Hanson. *Solving Least Squares Problems.* Prentice-Hall. Englewood Cliffs, New Jersey, 1974.

[38] G. Golub and C. Van Loan. *Matrix Computations.* 2nd Ed. John Hopkins University Press. Baltimore, 1989.

[39] G. Strang. *Linear Algebra and Its Applications.* 3rd Ed. Harcourt Brace Jovanovich. Orlando, Florida. 1988.

[40] B. Noble and J. Daniel. *Applied Linear Algebra.* 3rd Ed. Prentice-Hall. Englewood Cliffs, New Jersey. 1988.

[41] J. Albus, H. G. McCain, R. Lumia. "NASA/NBS Standard Reference Model for Telerobot Control System Architecture (NASREM)." NIST Technical Note 1235. Gaithersburg, MD, July, 1987.

[42] K. Chaconas, M. Nashman. "Visual Perception Processing in a Hierarchical Control System." NIST Technical Note 1260. Gaithersburg, MD, March, 1989.

[43] J. Fiala. "Manipulator Servo Level Task Decomposition." NIST Technical Note 1255. NIST, Gaithersburg, MD, October,

1988.

[44] L. Kelmar. "Manipulator Servo Level World Modeling." NIST Technical Note 1258. NIST, Gaithersburg, MD, March, 1989.

[45] A. Wavering. "Manipulator Primitive Level Task Decomposition." NIST Technical Note 1256. NIST, Gaithersburg, MD, October, 1988.

[46] J. Albus, M. Juberts, S. Szabo. "A Reference Model Architecture for Intelligent Vehicle and Highway Systems." Isata 25th Silver Jubilee International Symposium on Automotive Technology and Automation. Florence, Italy. June, 1992.

[47] S. Szabo, H. Scott, K. Murphy, S. Legowik, R. Bostelman. "High-Level Mobility Controller for a Remotely Operated Land Vehicle." Journal of Intelligent and Robotic Systems. Vol. 5, pp 63-77. 1992.

[48] K. Murphy. "Navigation and Retro-Traverse on a Remotely Operated Vehicle." Proceedings of the IEEE Conference on Intelligent Control and Instrumentation. Singapore, February, 1992.

[49] J. Fiala. "Note on NASREM Implementation." NIST Internal Report 89-4215. Gaithersburg, MD, December, 1989.

[50] Aspex Inc. "PIPE--An Introduction to the PIPE System." New York, 1987.