In Proceedings of SPIE Mobile Robots V, 1990.

Autonomous Navigation of Structured City Roads

Didier Aubert

ITMI, ZIRST, 11 chemin des Pres, B.P. 87, 38243 MEYLAN cedex, France

Karl Kluge and Chuck Thorpe

Robotics Institute, Carnegie Mellon, Pittsburgh, PA 15213

ABSTRACT

Autonomous road following is a domain which spans a range of complexity from poorly defined, unmarked dirt roads to well defined, well marked, highly structured highways. The YARF system (for *Yet Another Road Follower*) is designed to operate in the middle of this range of complexity, driving on urban streets. Our research program has focused on the use of feature- and situation-specific segmentation techniques driven by an explicit model of the appearance and geometry of the road features in the environment. We report results in robust detection of white and yellow painted stripes; fitting a road model to detected feature locations to determine vehicle position and local road geometry; and automatic location of road features in an initial image. We also describe our planned extensions to include intersection navigation.

1. INTRODUCTION

In 1985, 81.31% of the intercity passenger traffic in the United States — 1,418 billion passenger-miles — was done by private car. This translates into a tremendous amount of time spent by drivers engaged in the task of visually tracking and driving along a road. As autonomous road following programs become more competent they will be able to take over more and more of the burden of driving -- at first, in daylight under light traffic conditions, then later under more challenging illumination, weather, and traffic conditions. Driving long stretches on open freeway, while probably the easiest road following task to automate first, is only part of the larger domain of autonomous road following. The length of the average automobile trip in the United States in 1983 was just 7.9 miles [21]. In order to liberate people from the tedium of driving, road following systems will need to be able to follow city streets and maneuver through intersections, keeping track of what lanes are available for use, and not straying into lanes for oncoming traffic.

The gap between this vision of robot chauffeurs whisking people to and from work while they read the morning paper and the state of the art in robot road following is wide. While lane following on a freeway has been demonstrated at speeds of up to 96 km/hr [13], lane following is only one of the capabilities that an autonomous road following system must have. Current systems, while they have achieved fair levels of robustness in staying on the road, don't model the lane structure of the road. In order to progress toward the ultimate goal of robot chauffeurs, road following systems need improved capability to keep track of the lane structure of the road (both for purposes of lane following and for purposes of planning such as deciding if it is possible to change lanes or pass) and improved capability to detect and navigate through intersections.

Achieving these capabilities requires first the ability to robustly detect various road features (painted stripes, road/shoulder boundaries, etc.) under a variety of conditions (changes in lighting, pavement color, etc.). Dealing with different feature appearances can best be accomplished through the knowledge-based application of specialized segmentation techniques that work well in specific cases. As an example, yellow stripes can be located in both sunlit and shadowed image regions by thresholding on the pixel hue value. Given a known road geometry, the ability to reliably locate road features allows a system to determine the position of the vehicle relative to the road and drive the vehicle along the road in its lane. It is also necessary to instrument the segmentation techniques so that the system can detect when they have failed, and to implement strategies for determining the cause of failure and recovering from the problem. Lane boundaries may shift, requiring a change in the road model; markings may change in appearance, requiring a change in segmentation technique; or the vehicle may be approaching an intersection, requiring changes in sensing strategy.

Improvements in intersection navigation capability and the capability to detect and correct for changes in lane structure are particularly critical for making progress towards systems that can autonomously drive on city streets. On the highway, lane structure is relatively fixed, and the vehicle does not have to go around sharp corners or track the position of the road across a large intersection. In the city, lane edges shift as right and left turn lanes appear near intersections, and a vehicle needs to be able to maneuver its way through intersections. Intersections cover a large area on the ground, creating a need to combine information from several images in order to fix the location of the vehicle in the intersection and plan a path through it.

The problem of autonomous road following in an urban environment can be decomposed into the following subproblems: segmentation of the image data to extract road features, modeling of the local road geometry for vehicle localization and path planning, and intersection navigation. In the next section we examine representative existing systems, focusing on their approaches to these subproblems. After outlining our approach to these tasks, we present results in robust detection of painted lane markings, fitting features positions to a model of road geometry, and locating road features without using a strong a-priori model. We close by discussing our planned extensions to enable the system to navigate through intersections.

2. PREVIOUS WORK

2.1. VITS (Martin Marietta) [20]

- **Segmentation:** The road is extracted by thresholding a (red minus blue) image. The basic algorithm was extended to include two road classes, sunny and shaded, whose thresholds were found by sampling near the bottom of the image (which is assumed to be all road).
- **Road model:** The system assumes that the ground is locally flat, and projects the boundary of the road regions onto the ground. More sophisticated models of road geometry such as the hill-and-dale or zero-bank algorithm were rejected because of sensitivity to errors in segmentation and matching of corresponding points on the road edges, and because they could not handle intersections.

The VITS system was able to achieve fairly impressive performance, driving at up to 20 km/hr. on straight, obstacle-free stretches of road. While the paper referenced mentions intersection navigation as a criterion for selecting a technique for recovering the road shape, intersection navigation was not implemented. Sacrificing general capability for speed, the restriction to two road color classes limits the robustness of the segmentation.

2.2. FMC system [11]

- **Segmentation:** Regions from an Ohlander-Price style segmentation of an initial image are classified as road/non-road, and an optimal transform is derived to maximally separate road and non-road pixels. Normalized histograms of the transformed road and non-road pixels are used to generate likelihood ratios for each level of the transformed feature, which are then used to classify pixels in succeeding images. The likelihood ratios are updated for each image in order to handle changes in road color and lighting.
- **Road model:** Line segments fit to the road region boundaries are tested for continuity with the road boundaries from the previous image, consistency with constraints on the angle between successive segments on each side, and parallelism between segments separated by the expected road width.
- **Intersections:** The system examined the road region boundary segments for lines which might support an intersecting road.

The FMC system was also able to make runs at speed of up to 19 km/hr., with an image cycle time of 1.5 seconds. As in the VITS system, the road is reconstructed from the segmentation rather than fitting a road model to the results of the segmentation. Speed was a major criteria driving the design of the system, again resulting in a tradeoff between robustness and speed.

2.3. MARF (University of Maryland) [23]

- **Segmentation:** Small windows are placed at the predicted locations of the road edges at the bottom of the image. Sobel edge detecting and the Hough transform are used to determine the road edge location in the windows. The system then repeats this process, tracking the features up through the image.
- **Road Model:** Researchers at the University of Maryland investigated a number of algorithms for extracting 3-D road structure from image data. The most sophisticated algorithm [5] models the road as a horizontal segment swept perpendicular to a spine curve. Global optimization of the result is used to correct for errors in local point matching between the road edges.

The MARF (for *MAryland Road Follower*) system was ported to the Martin Marietta ALV and drove the

vehicle. The algorithms for recovery of road shape from image data are probably the most significant contributions of this work. Recently they have been working on declarative visual search strategies for road following [6].

2.4. VaMoRs (UniBw Munich) [13]

- **Segmentation:** Six 48-by-48 pixel windows selected from a grey-scale road image are convolved with one of 16 oriented bar masks for edge detection.
- **Road model:** The system uses the flat earth assumption, and models the lane followed as having parallel edges with constant separation and locally constant horizontal curvature.

The VaMoRs system combines custom hardware for image processing with an elegant control formulation to achieve runs at speeds of up to 96 km/hr. The system fits the lane edge points to a model of the road geometry rather than reconstructing the road boundaries from the segmentation results. The system does not model road structure other than the lane being followed, and does not handle intersections. The experience of the NAVLAB group at CMU with the use of oriented edge trackers [22] suggests that reliance on them as the only method of segmentation will not be robust under difficult shadow conditions, although the Munich researchers claim that they have not encountered problems with this.

2.5. LANELOK (GMR) [8], [9]

- **Segmentation:** Several segmentation methods were tested. In the first method, edges segments are extracted from a thresholded Sobel edge image, and the edge segments vote in a Hough transform for right and left lane edges. In the second method, growing and shrinking are applied to the binary edge image to thicken the lane boundaries, and region tracing is applied to extract them. In the third method, search areas are defined around the expected locations of the left and right lane edges, and template correlation is done to find the lane markers. A least-squares fit is done to the optimum correlation values to determine the lane edges. These methods have been tested independently, and are not used cooperatively.
- **Road Model:** The lane is modeled as having parallel edges separated by a constant width. Shifts in the lane markers are detected and corrected for.

LANELOK's algorithms have been tested on more than 3000 frames of videotaped data. The system is designed to track lane boundaries in a freeway environment, and appears to work well, if slowly (three seconds/image on a VAX 8600 for the template correlation algorithm, similar times for the Hough algorithm). The system also incorporates obstacle detection, using template correlation to locate other vehicles in the lane.

2.6. University of Bristol [17]

- **Segmentation:** White lane markings are detected by creating a binary image in which the selected pixels correspond to pixels in the intensity image which are brighter than a threshold value and between two strong intensity gradients of opposite sign separated by the expected lane marking width. Regions in the binary image are extracted, and shape cues are used to eliminate noise regions.
- **Road model:** The surviving regions are back-projected onto the ground plane, and a parabolic model is fit to each candidate region. Dashed lane markings are accommodated by fitting arcs to all pairs of short regions. Minimum separation and constant separation constraints are used eliminate erroneous candidate arcs, and to produce a final set of consistent lane markings.

From [17] it is unclear how much testing the algorithm has received, but the approach seems sound, and could adapt easily to use improved segmentation techniques.

2.7. ARF (CMU) [12]

- **Segmentation:** The system has two tracking algorithms, a profile correlation technique and a Sobel edge tracker. These algorithms are used in tandem to track roads in aerial images. If the results of the two tracking methods diverge, then failure analysis rules are invoked to determine the cause of the problem (intersections, changes in road width, changes in surface material, overpasses, occlusion, vehicles on the road), and appropriate corrective actions are taken.
- **Road Model:** The road is modeled as locally having a parabolic shape. No interior road structure is modeled.

The ARF system works in the domain of tracking road networks in aerial images rather than in a vehicle navigation domain, but is included because of the influence of its architecture on the design of YARF (specifically, the use of multiple segmentation techniques and explicit failure analysis).

2.8. Sidewalk II (CMU) [7]

- **Segmentation:** The system uses an earlier version of the color classification algorithm that is used in the SCARF system described below. It can also fuse the color segmentation with a range image segmentation to distinguish between stairs, a ramp, and the surrounding grass slope.
- **Road Model:** The system has a map of the geometry of the system of sidewalks it navigates on.
- **Intersections:** Line segments fit to the edges of the extracted road region are matched with expected edges from the map to determine position within an intersection.

The Sidewalk II system was designed to operate in an environment where the segmentation problem would be relatively easy, allowing exploration of the higher level issues of route planning and intersection navigation. It performed well, albeit at slow speeds, but is limited by its need for a geometric map of the intersections it will encounter.

2.9. SCARF (CMU) [4]

- **Segmentation:** An adaptive color classification scheme is used, with four to eight color classes each used to model road and non-road areas.
- **Road model:** The system assumes that the road is locally straight, with a known constant width. The classified pixels vote in a Hough scheme to locate the vanishing point and orientation of the road in the image.
- **Intersections:** Once the main road has been found, the pixels on that road are subtracted from the Hough space and further peaks corresponding to intersecting paths are searched for.

The SCARF system has been one of the most robust and successful of the road following systems developed at CMU, following the path up Flagstaff Hill under a wide variety of weather and road conditions. With its Hough voting scheme, it is the only one of the systems discussed that has an explicit representation of how certain it is that it has found the road.

2.10. ALVINN (CMU) [15]

- **Segmentation:** ALVINN does not have a fixed segmentation technique. It consists of a three-layer backpropagation neural network which is trained on the road to be followed or on simulated data.
- **Road model:** There is no model of the road, other than whatever model is implicitly encoded in the weights learned by the network.

ALVINN is *sui generis*. As mentioned above, it does not have a fixed segmentation and evidence combination strategy, but learns one from training examples. It performs very well, and has driven the NAVLAB at its top speed of 20 MPH. On the other hand, its lack of any explicit representation makes it hard to evaluate how general a road following capability it possesses (for instance, can it be trained to lane follow in any lane on roads of differing widths?)

2.11. Analysis

All of these systems use a single segmentation technique to locate the road, making them vulnerable to situations in which that technique fails. Binford made the same point with respect to object recognition programs [3]. Global color classification schemes such as those used in [20], [18], and [11] work well for segmenting the road surface from the background, but work less well at detecting painted lane marking because they look only at pixel color and fail to consider geometric constraints. Edge detectors have problems with textured areas and shadows, particularly mottled shadows from trees.

There are two classes of techniques used to compensate for errors made by the segmentation algorithms. The first is focusing, in which predictions of road feature location are used to limit the areas of the image which are examined [23], [13], [9]. The second is use of global constraints, in which a model of the road structure is used to eliminate errors in segmentation. A good example of this second approach is the Hough voting scheme used in SCARF, which uses the assumptions of constant known road width and a straight road to correctly locate the road even in cases where there are many misclassified pixels.

Few of these systems have any explicit representation of how confident they are that they located the road, making it possible for them to "hallucinate" and drive off the road. Only LANELOK (and ARF in the aerial road tracking domain) has any mechanism for detecting changes in road structure based on segmentation failures. Intersection navigation capabilities of these systems are very limited. This is largely because they process one image at a time, and real city intersections are large compared to the field of view of typical cameras. The exception to this is the Sidewalk II system, which used a second camera to see around corners at intersections.

YARF addresses these problems through the following mechanisms:

- multiple segmentation techniques which are specialized to detect particular kinds of features or to work in particular situations;
- examination of the results of the segmentation techniques and their geometric consistency with a model of the road structure to detect when the systems fails, when the road appearance or structure changes, or when the vehicle is approaching an intersection; and
- use of a local map to integrate feature location data and locations where segmentation failures have occurred over multiple frames.

The remaining sections of this paper describe our current research in implementing these mechanisms, describing the progress we have made since the initial results reported in [10].

3. ROBUST PAINTED STRIPE DETECTION

A major component of the program of research we described in [10] was the investigation of specialized segmentation techniques to robustly extract different types of road features. Our recent experiments in this area have concentrated on testing two algorithms, one for detecting yellow stripes using pixel hue, and the other for detecting white stripes using an oriented bar detector. These algorithms have been tested both in open loop mode, where they track a stripe as a human drives the vehicle, and in closed loop mode, where their results are used to drive the vehicle. The implementation of these algorithms is described in detail in [1].

Hue appears to be a very stable cue for detecting yellow stripes under a wide variety of road and lighting conditions. Putting red at zero degrees on the color wheel, pure yellow has a hue of 60 degrees. Histograms of yellow stripe pixels in both bright, sunlit images and darker, shadowed images show a peak located at 60 degrees, with a width of 30 degrees on either side. Pixels with hues between 30 and 90 degrees are classified as yellow, pixels with hues outside this range are classified as background (see figure 3-1). In order to avoid grey pixels being classified as yellow due to the instability of hue near the intensity axis, we also require yellow pixels to have a saturation of at least 0.1. The algorithm does not explicitly compute the hue of the pixels. Instead, it tests the RGB value against two planes containing the intensity axis which bound the desired section of the color cube. Those pixels whose RGB values fall on the correct side of both planes are labeled as yellow pixels, other pixels are labeled as background. The mean row and column of the yellow pixels is returned as the position of the center of the yellow stripe.

Robust detection of white stripes is done by looking for a bright bar of a specified width at a specified orientation. Using an oriented operator reduces the effects of noise such as shadows or oil stains on the pavement. Searching for a bar rather than an edge and blurring along the direction of the bar also improves the robustness of the operator. The correlation is done with the blue band of the color road image.

Two techniques are used together to achieve a fast correlation. The first is the use of only +1 and -1 as weights. This speeds up the correlation by reducing the number of additions and subtractions needed. When the mask is shifted one pixel to the right, the leftmost pixel previously included in the correlation sum is removed, the new rightmost pixel value is added in, and corrections are made for pixels whose weight changes sign when the mask is shifted. As an example, if the mask is (-1 -1 -1 1 1 1 -1 -1 -1), only four additions/subtractions are needed: one for the pixel which shifts off the left edge of the correlation window, one for the new pixel on the right edge, and one each for the two pixels whose weights change sign. The second technique used to increase the speed of the correlation is using a window which is a parallelogram parallel to either the rows or columns of the image rather than an oriented rectangle, which speeds up the correlation through a more regular pattern of pixel access.

Figures 3-2 shows these operators tracking the center double yellow line and the right and left white lines on a sunny, well-lit road. Figure 3-3 shows them tracking the center double yellow line and right white line on a road covered with mottled shadows from trees. While these algorithms do not perform perfectly, they appear to be more robust than any of the other techniques we had investigated. Detecting when these operators have failed to find the desired feature is simple. In the case of the oriented bar operator, the correlation peak will not differ sufficiently from the background level. In the case of the yellow hue operator, the area of the yellow pixel regions in the window is either very small (if there is no yellow stripe) or much larger than the road model would predict (if the window falls onto a grassy region — surprisingly, some grass has a hue very close to the hue of yellow stripes). In the next section we discuss the combination of the individual measurements of feature positions into an estimate of the local road curvature and the position of the vehicle on the road.

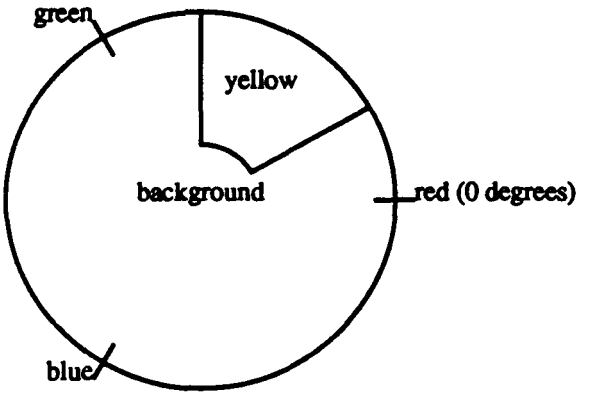


Figure 3-1: Classification by hue to detect yellow stripes



Figure 3-2: Yellow hue and white bar operators: sunlit scene

4. THE ROAD MODEL AND FITTING DETECTED FEATURE LOCATIONS

YARF models the road as a *generalized stripe* -- a one-dimensional feature cross-section which is swept perpendicular to a spine curve. The road in figure 3-2, for example, can be modeled by the following feature cross-section:

- a solid white stripe which starts -358 cm. from the spine;
- a lane of pavement which starts -342 cm. from the spine;
- a solid double yellow line which starts 0 cm from the spine;

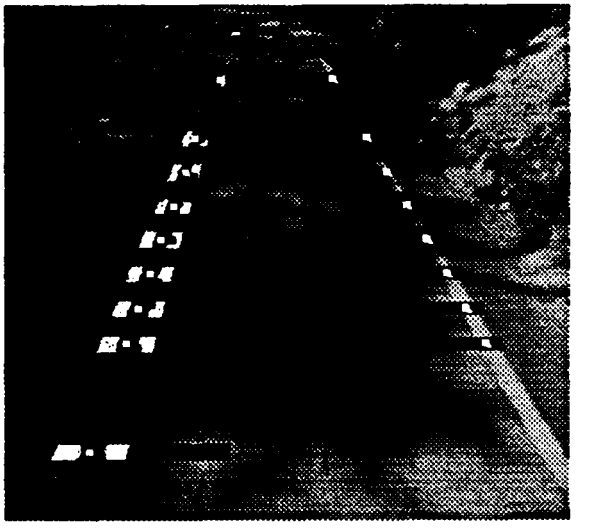


Figure 3-3: Yellow hue and white bar operators: scene with mottled shadows

- a lane of pavement which starts 50 cm. from the spine;
- a solid white stripe which starts 403 cm. from the spine; and
- a shoulder which starts 419 cm. from the spine.

An important design decision is the question of how complex a spine curve should be allowed. Should the road model allow for banking? Should it assume a locally flat ground plane, or allow changes in surface slope? Answering these questions requires considering not only how roads behave in the real world, but what kinds of models produce algorithms that are computationally tractable and results which are stable in the presense of noise and usable for navigation even if they do not reproduce the world with complete fidelity.

We have chosen to adopt a model similar to that used in the Munich VaMoRs system [13] and work at the University of Bristol [17]. The road spine is locally approximated by a circular arc, with the road lying in a flat ground plane. In order to do a linear least-squares fit, a parabolic approximation to a circular arc is made, $x = 0.5 * curvature * y^2 + slope * y + lateral_offset$. Such a model allows computationally efficient fitting, and produces results on real roads that allow robust navigation even though though the actual road may not be flat or locally a circular arc. The model of the feature cross-section is used to correct the position of the detected points so that they lie roughly along the center

spine of the road. We add the feature offset to the x coordinate given by the parabola equation above, which is a small-angle approximation to the proper correction (it assumes that the cosine of the angle between the tangent to the road and the y axis is approximately one).

Figure 4-1 shows the fit of the road model describes above to the feature positions detected in figure 3-2. The diamonds are the individual feature locations, labeled with the corresponding feature number. The black dots lie along the parabolic fit, and the equivalent circular arc spine road model is drawn to show the road features. To give an idea of scale, the tick marks on the line on the left of the drawing are spaced two meters apart.

In order to increase the stability of the parameter estimates, we fit the model to the points detected over the last several frames (typically three to six). Figure 4-2 shows data accumulated over a sequence of eight images placed into a global coordinate frame. The squares show the individual feature position estimates. The left digit of the number by each square is the frame number, while the right digit is the feature number in the cross-section model. The asterisks connected by dashed lines show the desired paths fed to the path planner, and the diamonds with lines pointing out from them show the estimates of road centerline position and direction. As can be seen from this figure, our camera calibration and the inertial navigation data produced by the vehicle are very accurate — the points from different images along the two lane markers line up very well as the vehicle goes through the gentle curve.

We have done some experiments with using robust M-estimation [2] to perform the fit. Standard least squares minimizes the square of the residuals of the data values. Robust M-estimation minimizes the sum of a function of the residuals which falls off more rapidly for large residuals, making the fit less sensitive to outlying data observations and allowing their detection. So far we do not have results which allow us to decide if this will successfully help in the detection of incorrect feature location data.

In the mode where YARF is following a road between intersections, a predict-segment-fit-move loop is used. Inertial navigation is combined with the estimate of vehicle position from the previous image to predict feature locations in the current image. When started, the system does not have a prediction of where to place trackers to locate the road features. Initially we had a human operator use a cursor to select points along one feature to provide the system with the initial vehicle position relative to the road. Now we are experimenting with a technique to automatically extract candidate road features using Sobel edge detection, Hough transforms, and shared vanishing point and global continuity constraints. We describe this algorithm in the next section.

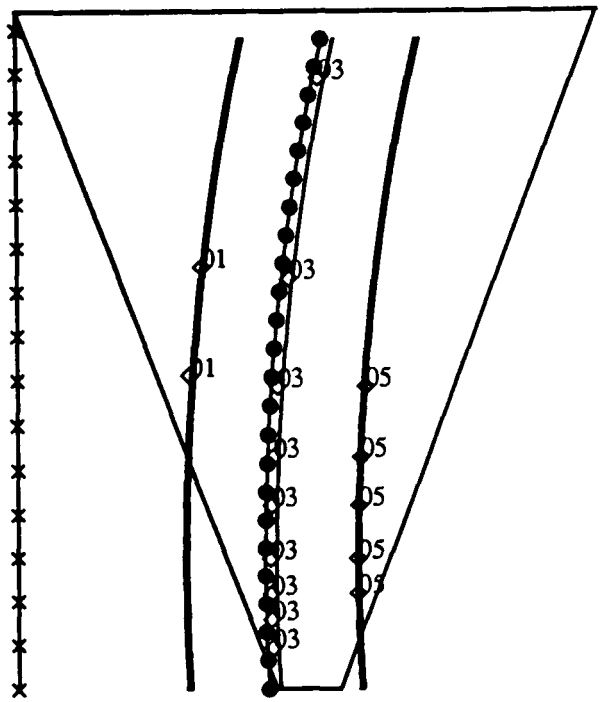


Figure 4-1: Fit of road model to detected feature positions

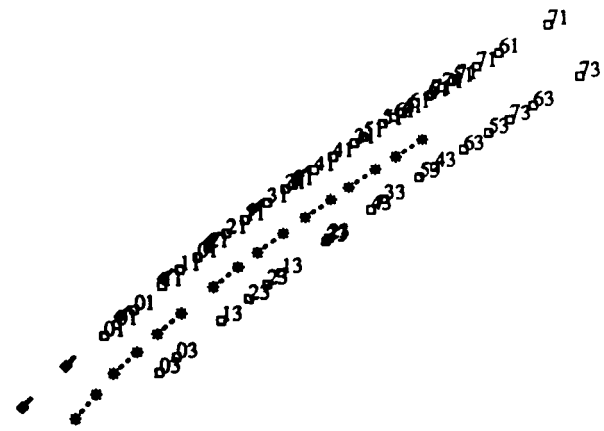


Figure 4-2: Feature locations in a sequence of eight frames

5. BOOTSTRAP LOCATION OF ROAD FEATURES

There are two main techniques to compensate for incorrect segmentation results, the use of prediction to

focus processing near features and the use of global constraint. In YARF's road-following mode, focusing is used, both to reduce computation cost and to reduce errors in feature location. In the absence of predictions of feature locations, global constraints must be used.

The features which form a straight road are parallel lines on the ground, which project into the image as lines which meet at a common vanishing point. In order to handle curved roads, the road is modeled as a sequence of straight segments by dividing the image vertically into a small number of horizontal bands and approximating the road as straight within each band, as Polk and Jain do [14]. In order to reduce the chances of noise in the image leading to the selection of a spurious vanishing point in some of the bands, a global optimization is performed which takes into account both the support for a given vanishing point within a band and the continuity of features between bands.

The Sobel edge detector is run on an image produced by preprocessing an RGB road image. We have experimented with various kinds of preprocessed images, including the red, green, and blue bands of the color image, the intensity image corresponding to the color image, and the (blue minus red) image. The gradient magnitudes are thresholded to create a binary image of candidate edge points. Any segmentation technique could be used which would give points where there are discontinuities in the image along with an estimate of the orientation of the discontinuities.

Lines in the image are represented by the column where they cross a specified row (in this case, the row which contains the horizon), and the angle they make with respect to the rows of the image. Given an edge point (*row*, *col*) with gradient direction *theta*, and a vanishing point *vp*, the line orientation voted for is $theta_{line}$ = arctan((*row* — *horizon*) / (*col* — *vp*)). The difference in angle between the line and the edge gradient estimate is $diff_{ang}$ = min(| $theta_{line}$ — *theta* |, 180 — | $theta_{line}$ — *theta* |). The vote for the line with vanishing point *vp* and orientation $theta_{line}$ has weight 1.0 — ($diff_{ang}$ / 90). In order not to bias the voting against lines near the corners of the image, the votes for a given line are normalized by the visible length of that line in the image.

The image is divided into a small number of horizontal bands (four to seven), and voting for the most popular vanishing point is done for each band. Once all the edge points have voted for all the lines they could lie on, the accumulator array is thresholded, and peaks in the accumulator array are detected. All the detected peak bins which support a given vanishing point are summed to give the total support for that vanishing point. The top three candidate vanishing points for each horizontal band in the image compete in a search for the globally best set of vanishing points. The criterion function for that search has a term for the strength of support for each vanishing point in the set, and a term which rewards continuity of features between adjacent bands in the image.

As a method for extracting road features, this technique has several advantages. The only calibration required is the determination of the horizon row, which can be easily computed from an image taken on a straight stretch of road. The model of road appearance used is fairly generic, assuming only that the road consists of features separated by constant widths and curving slowly enough that they can be approximated by straight lines within a horizontal band of the image. This eliminates the need to train the system separately for each road it encounters. It extracts as much of the overall lane and road structure as it can detect, rather than just a right and left road edge or road centerline. If given a model of the road structure so that it could label the various features found, it could steer the vehicle down a specified lane using a pure pursuit strategy, once again without having to have any calibration from the image to the world other than a single gain parameter. While relatively slow in a serial implementation, the algorithm has a great deal of parallelism that could be exploited.

Figure 5-1 shows the results of the bootstrap algorithm on the image from figure 3-2. The algorithm successfully finds the white stripes on the left and right side of the road, the double yellow line in the middle, and part of the shoulder edge. There are a few extraneous edge segments caused by noise, for instance the tree shadow which runs parallel to the road.

This part of the research is still very experimental, and quantitative performance results are not yet available. Also, the issues of the best preprocessing to apply to the color image to produce the single-band image that the Sobel is run on, and how to set thresholds used in the algorithm are still under investigation.

So far we have described how YARF finds the location of the road in an initial image, how it locates individual features given a prediction of the road location in subsequent images, and how it combines those new data points into an updated estimate of the vehicle posi-

tion on the road. Next we discuss our plans to integrate feature locations from multiple images into a local map, to reason about detected failures to locate features in order to detect changes in the road structure and the approach of intersections, and to use the integrated local map to navigate through intersections.

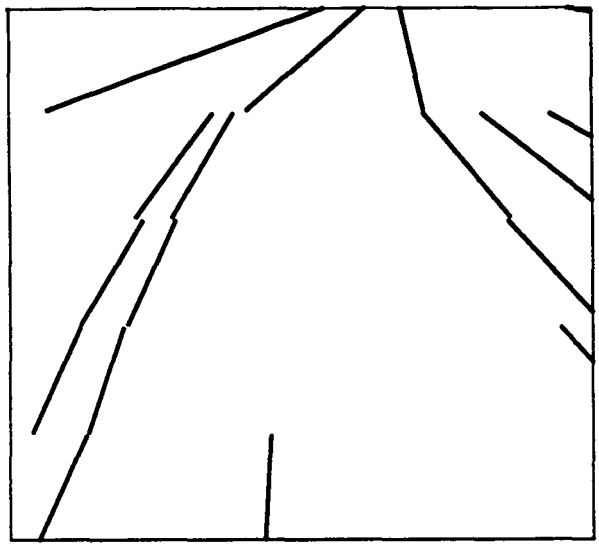


Figure 5-1: Vanishing point Hough algorithm: detected lines

6. INTERSECTION NAVIGATION

In order to navigate a vehicle through an intersection, an autonomous road following system must detect that the vehicle is approaching an intersection, use perception to locate the roads branching out from the intersection, and plan a path from the current lane through the intersection into the desired lane of the next road segment to be followed. More general intersection navigation capabilities than have been demonstrated in current systems require the coordination of perceptual data from multiple images into a single local map of the intersection.

There are several reasons for this. The first is that it gives the system a memory — once the system has detected an approaching intersection based on the disappearance of some of the road features, it does not have to devote processing cycles to remake this discovery on the following images. The second reason is that intersections of city streets cover a large area compared to typical camera fields of view. The integration of perception results from multiple images (both from the same camera over time as the vehicle moves, and from multiple cameras pointing in different directions to cover a larger field of view) is necessary in order to create a complete model of the intersection's geometry.

Part of the NAVLAB project at CMU has been the creation of utilities to support an annotated map for robot navigation [19]. The function of the annotated map is to provide a framework for the communication of the results of different perception modules through a shared geometric database. YARF will use the annotated map facility to store results from multiple images in a common coordinate system.

Our first experiments will focus on the question of whether the results from multiple images taken by multiple cameras can be combined to produce a coherent, accurate map of the scene geometry. The coherence of data from multiple frames taken by a moving camera (see figure 4-2) is promising. In our initial experiments we will run the road following process with two different cameras at the same time and examine how well the feature positions match between the two cameras.

The next step is to use local map data to detect the approach of an intersection. The annotated map will be used to store information about locations where features were not detected where they were expected. YARF will then reason about the missing feature data to determine whether the road model has changed, whether a feature has become obscured, whether a different segmentation technique should be switched to because of a change in feature appearance, or whether the vehicle is approaching an intersection.

After YARF has the capability to detect that the vehicle is approaching an intersection, the final step is to create perception strategies for locating the roads branching out of the intersection. Rather than assume complete knowledge of the intersection geometry as the Sidewalk II system did, YARF will assume only knowledge of the feature cross-sections of the roads which meet at the intersection. The current plan is to use a feed-forward tracking strategy similar to that used by MARF to follow road features around corners and through the intersection, using the feature cross-section models to predict feature locations once an initial estimate of a road branch's location is available. YARF will use multiple camera to cover a wider field of view, using calibration information to track features across the overlapping fields of view.

7. CONCLUSION

The YARF project has made substantial progress since we reported our first results. We have gone from an initial "pot-luck" collection of segmentation techniques to focused research into robust techniques to track different types of road feature. We have im-

plemented routines to fit individual estimates of road feature locations to models of generalized stripe roads whose spines are locally approximately circular arcs, and are investigating issues of filtering and the use of robust estimation to improve reliability. We have made preliminary experiments in the initial location of road features using Hough line detection techniques and shared vanishing point constraints. We have a plan of research to add intersection navigation capabilities into the system. YARF has driven the NAVLAB at speeds of up to 6.75 MPH on a public road running through a golf course near campus, and we expect speed improvements from the use of multiple processors.

Other research within the NAVLAB project at CMU has focused on planning in the domain of driving in traffic on city streets, using a simulator (PHAROS) to provide the input of the system [16]. As the YARF project progresses, it will provide some of the perceptual capabilities needed to transfer results from the research using the PHAROS simulator into the real world. It provides an open ended architecture which improved segmentation techniques can easily be plugged into to improve system performance.

8. ACKNOWLEDGEMENTS

Our thanks to Thad Druffel, who worked on various pieces of the system, and implemented a multiprocessor version of YARF.

This research is sponsored by DARPA, DOD, monitored by the US Army Engineer Topographic Laboratories under contract DACA 76-89-C-0014, titled "Perception for Outdoor Navigation".

9. REFERENCES

[1] Aubert, Didier, and Thorpe, Chuck.
Color Image Processing for Navigation: Two Road Trackers.
Technical Report CMU-RI-TR-90-09, Robotics Institute, Carnegie Mellon, April, 1990.

[2] Besl, Paul J., Birch, Jeffrey B., and Watson, Layne T.
Robust Window Operators.
In *Proceedings International Conference on Computer Vision.* 1988.

[3] Binford, T. O.
Survey of model-based image analysis systems.
Int. J. Robotics Research 1, 1981.

[4] Crisman, Jill.
Color Vision for the Detection of Unstructured Roads and Intersections.
PhD thesis, Carnegie-Mellon University, 1990.

[5] DeMenthon, Daniel, and Davis, Larry.
Reconstruction of a Road by Local Image Matches and Global 3D Optimization.
In *Proceedings 1990 IEEE International Conference on Robotics and Automation.* May, 1990.

[6] Dickinson, S., and Davis, L.
An Expert Vision System for Autonomous Land Vehicle Road Following.
In *Computer Vision and Pattern Recognition Conference.* 1988.

[7] Goto, Y., Matsuzaki, K., Kweon, I., and Obatake, T.
CMU Sidewalk Navigation System: A Blackboard-Based Outdoor Navigation System Using Sensor Fusion with Colored-Range Images.
In *Proc. Fall Joint Computer Conference.* November, 1986.

[8] Kenue, Surender K.
LANELOK: Detection of Lane boundaries and Vehicle Tracking Using Image-Processing Techniques -- Part I: Hough-Transform, Region Tracing and Correlation Algorithms.
In *SPIE Mobile Robots IV.* 1989.

[9] Kenue, Surender K.
LANELOK: Detection of Lane boundaries and Vehicle Tracking Using Image-Processing Techniques -- Part II: Template Matching Algorithms.
In *SPIE Mobile Robots IV.* 1989.

[10] Kluge, Karl, and Thorpe, Charles E.
Explicit Models for Robot Road Following.
Vision and Navigation: The Carnegie Mellon Navlab.
Kluwer Academic Publishers, 1990, Chapter 3.

[11] Kuan, Darwin; Phipps, Gary; and Hsueh, A.-Chuan.
Autonomous Land Vehicle Road Following.
In *Proceedings First International Conference on Computer Vision.* June, 1987.

[12] McKeown, David M., and Denlinger, Jerry L.
Cooperative Methods for Road Tracking In Aerial Imagery.
In *Proceedings Computer Vision and Pattern Recognition.* June, 1988.

[13] Mysliwetz, Birger D., and Dickmanns, E. D.
Distributed Scene Analysis for Autonomous Road Vehicle Guidance.
In *Proceedings SPIE Conference on Mobile Robots*. November, 1987.

[14] Polk, Amy, and Jain, Ramesh.
A Parallel Architecture for Curvature-Based Road Scene Classification.
In *Roundtable Discussion on Vision-Based Vehicle Guidance '90 (in conjunction with IROS)*. July, 1990.

[15] Pomerleau, Dean A.
Neural Network Based Autonomous Navigation.
Vision and Navigation: The Carnegie Mellon Navlab.
Kluwer Academic Publishers, 1990, Chapter 5.

[16] Reece, Douglas A., and Shafer, Steve.
An Overview of the PHAROS Traffic Simulator.
In *Proceedings of the Second International Conference on Road Safety*. September, 1987.

[17] Schaaser, L. T., and Thomas, B. T.
Finding Road Lane Boundaries for Vision Guided Vehicle Navigation.
In *Roundtable Discussion on Vision-Based Vehicle Guidance '90 (in conjunction with IROS)*. July, 1990.

[18] Thorpe, Charles; Hebert, Martial; Kanade, Takeo; and Shafer, Steven A.
Vision and Navigation for the Carnegie-Mellon Navlab.
IEEE Transactions on Pattern Analysis and Machine Intelligence 10(3), May, 1988.

[19] Thorpe, Charles, and Gowdy, Jay.
Annotated Maps for Autonomous Land Vehicles.
In *Proceedings of the DARPA Image Understanding Workshop*. 1990.

[20] Turk, Matthew A.; Morgenthaler, David G.; Gremban, Keith D.; and Marra, Martin.
VITS -- A Vision System for Autonomous Land Vehicle Navigation.
IEEE Transactions on Pattern Analysis and Machine Intelligence 10(3), May, 1988.

[21] U. S. Bureau of the Census.
Statistical Abstract of the United States: 1988.
U. S. Bureau of the Census, 1987.

[22] Wallace, R.; Matsuzaki, K.; Goto, Y.; Crisman, J.; Webb, J.; and Kanade, T.
Progress in Robot Road Following.
In *Proceedings IEEE International Conference on Robotics and Automation*. April, 1986.

[23] A. Waxman, J. LeMoigne, L. Davis, B. Srinivasan, T. Kushner, E. Liang and T. Siddalingaiah.
A visual navigation system for autonomous land vehicles.
Journal of Robotics and Automation, Vol. 3, 1987.