FEATURE EXTRACTION FOR VISION GUIDED ROAD VEHICLES

B. T. Thomas E. L. Dagless R. A. Lotufo D. J. Milford A. D. Morgan
J. F. Morrissey

Faculty of Engineering, University of Bristol, Bristol, UK

This paper reports the early results of an investigation conducted at the University of Bristol into autonomous road vehicle control. The emphasis of the work is on real-time vision, with the objective of steering a vehicle on well-defined roads at speeds up to 60 km/hr. This report gives an overview of the computer hardware and the low-level vision algorithms currently being explored. Our algorithms fall into two categories: bootstrap and real-time analysis. For the bootstrap mode we present a basic edge-detection and surface segmentation approach and an alternative method based on texture analysis. Our current emphasis is on real-time analysis techniques. Here we present only a single straightforward real-time algorithm, more complex approaches being given in companion papers.

Over the past few years there has been growing interest in the use of computer vision techniques for autonomous road vehicle control. Work in the USA has concentrated on a thorough analysis of the road scene allowing vehicles to drive slowly but relatively safely along well-defined roads [1,2,3,4]. In Germany the emphasis has been on real-time processing of images to allow vehicles to travel at higher speeds on roads with clear edges and lane markings [5,6,7,8].

The University of Bristol has been funded to investigate some of the problems associated with navigating robot vehicles along well-defined roads at speeds up to 60 km/hr. We present work in progress on a new computer architecture and associated vision processing algorithms for autonomous road vehicle control. The vehicle will utilise a passive vision system, relying purely on the information derived from a monochrome video camera mounted at the front. During the development phase we will be using a small electric vehicle, as a demonstrator, to track roads and paths in the University grounds. This paper gives an overview of the low-level image-processing techniques we are exploring for feature extraction.

At this stage we are concentrating our efforts on the ability to identify the primary carriageway. Recognising junctions and turning right or left are deferred until we have an operational system. This ability, however, together with identifying and avoiding obstructions is very much part of our objectives and we are attempting to avoid techniques that will need to be abandoned when dealing with these more general problems.

We will give a brief description of the hardware configuration used to support the real-time analysis together with some of the algorithms we are using. The main emphasis of our present work is on real-time algorithms. These algorithms gain their speed by analysing only a portion of the image and depend, therefore, on a knowledge of the approximate road orientation and edge locations. This information is provided initially by a bootstrap mode of operation.

Our work on the bootstrap mode is only preliminary as we have not yet established the best real-time techniques, and do not yet know precisely the information that will be required. We are exploring two approaches. The first uses differential filters to locate edges combined with segmentation techniques to find the road surface. The second method utilises a texture approach to edge finding based on the formation of a covariance matrix. This is intended to provide information for both edge identification and surface segmentation in a single algorithm. It also overcomes problems associated with narrow linear feature such as white lines, which can confuse differentiating filters.

The video-rate methods we are exploring are generally very fast but are capable of loosing the edges they are tracking. For this reason we intend to employ more than one real-time technique so that failure in any one approach can be detected and corrected. Here we will present only a single real-time algorithm which scans a portion of the image in horizontal strips, identifying and confirming road-edge locations. Other real-time algorithms are presented in companion papers.

COMPUTER SYSTEM ARCHITECTURE

Figure 1 shows a schematic of the proposed system architecture. The main computational power of the system is provided by an array of Inmos transputers. Video data from a camera mounted on the front of the vehicle is digitised and then passes through a preprocessing unit. The purpose of the preprocessor will be to perform many of the convolving operations in real time prior to passing the data into one of several frame stores. It seems likely that we will use the Inmos A100 to perform the convolutions but at present we have not implemented any preprocessing stages and are performing all filtering

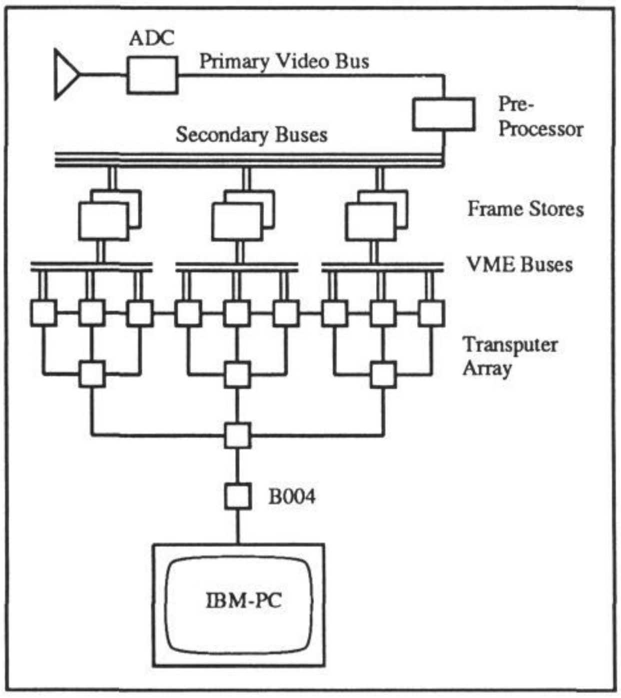

Figure 1: The System Architecture

operations in software. To overcome the problems associated with the low transputer link bandwidth we have placed the frame stores directly in the address space of the top level transputers, thus avoiding the need to pass image data over the links. Our current architecture allows several transputers to access a single frame store over a 32 bit bus, with an arbiter associated with each frame store. The contention created by this arrangement is small as long as data required by each of the top-level transputers is moved initially into local memory. In our early implementation the frame stores will contain identical images but in the final design each will have access to any of the secondary video buses. The frame stores can also be used for video generation and can, therefore, be used to generate test and diagnostic images. For the remaining transputers in the array we are using commercially available boards interfaced to an IBM PC.

The frame stores, video controller and top-level transputer boards are designed at the University of Bristol and are still experimental in nature. In future designs we have decided to increase the number of frame stores and have only one transputer associated with each. This will avoid the need to transfer data initially into local memory.

To date we have constructed one channel of this system involving a single double buffered frame store with 4 DMA transputer boards at the top level. Additional transputers are provided by Inmos B004 and B003 boards. The algorithms are first developed on Sun workstations and then coded in OCCAM for testing on the transputer array. Further details of the hardware configuration are left to a separate report (Milford et al., in preparation).

-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1
-1	-1	-1	-1	1	1	1	1

-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	-1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

Figure 2: Differentiating Filters

BOOTSTRAP MODE

Several of the techniques that we have been investigating depend on an initial knowledge of the road position. The bootstrap mode of operation is intended to provide this initial information by performing a thorough analysis of the image to determine the road surface and edges. Work on this aspect of the problem is still preliminary since we have not made a decision regarding the best real-time analysis techniques. We are currently investigating two possible approaches. The first utilises differentiating filters in order to establish the edge locations of the road together with segmentation techniques to identify the road surface. This information is then combined to provide an estimate for the driving corridor. The second approach employs a texture based method to identify regions of strong variance anisotropy.

Edge-Detection and Surface Segmentation

We make a preassumption that the vehicle is initially stationary on the road pointing approximately in the required steering direction. A single video frame is then captured and analysed. The technique has two aspects; the first involves searching for well-defined edges that have approximately the correct position and orientation to be road-edge candidates, and the second uses grey-level and variance information for segmenting the road surface. The results from both techniques are then checked for self-consistency and are used to define the driving corridor.

For the edge-detection algorithm we use 8x8 convolving filters to differentiate the image in both the horizontal and vertical directions (Figure 2). Large filters have been chosen as they are particularly sensitive to the extended edges associated with road boundaries, while suppressing trivial responses. Box filters of this kind, although very simple, are particularly sensitive to faint edges at the cost of precision in edge location. In this case we are more concerned with edge sensitivity than with determining positions to sub-pixel accuracy.

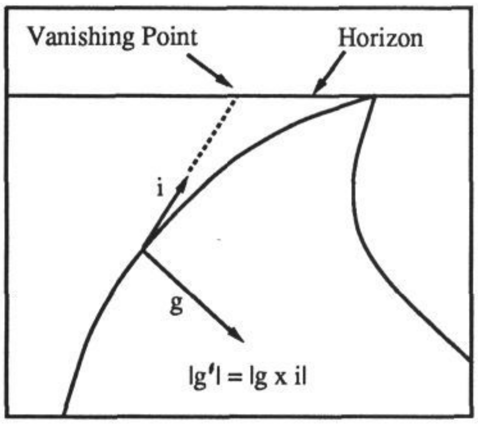

Figure 3: Orientation Dependent Edge Weighting

For computational efficiency we do not range the filter over the entire image but along every 8th row of the image. Thus, we obtain a continuous horizontal measure of the x and y derivatives at discontinuous points in the vertical direction. This technique is not good at locating exactly horizontal edges, since the filter may scan above or below the edge center, but is perfectly adequate for the inclined edges we are concerned with here.

To further increase the sensitivity to the primary road edges we weight the magnitude of the gradient by a factor depending on its orientation. This is achieved by selecting a vanishing point in the image which is on the nominal horizon directly above the image centre. We then calculate the cross product between the observed vector gradient and a unit vector pointing directly towards the vanishing point (Figure 3). The result becomes the new gradient magnitude. In this way an edge that is directly aligned with the vanishing point has its gradient unmodified while one that is perpendicular to the expected direction is totally suppressed. Any edges that lie above the horizon are also eliminated. This technique can suppress road edges near the horizon when the road is curved but in practice this is no great problem.

The resulting edges are then thinned using maximum local gradient techniques and thresholded. Automatic thresholding is problematic since there is no clear separation that emerges in the output histogram. For the present a threshold of 25% of maximum gradient is giving reasonable results.

The second aspect of this technique involves segmentation of the road surface. We accomplish this by dividing the entire image into non-overlapping squares, each 8x8 pixels. The average grey level $\overline{v}$ and standard deviation σ are computed for each square and thresholding is performed in the basis of:

$$areavalue = \overline{v} + \alpha\sigma$$

where α is a weighting factor of order unity. The threshold range is determined automatically by inspection of the *areavalue* of windows directly in front of the vehicle, which are assumed to be typical of the road surface. The resulting segmented image is then area-grown to eliminate isolated classifications.

Figure 4 illustrates the result of these two algorithms. The top-left panel shows an image of a road near the University buildings. This is meant to represent a near-worst-case situation with parked cars completely obscuring one side. The top-middle panel shows the result of the edge-detection operation with the top-right panel showing the thinned edges after thresholding. The extended nature of the detected edges represents the 8-pixel band over which the edge detector passed and does not indicate the edge orientation. As we see, the algorithm has identified candidate edges on the right hand side of the road and at the edge of the pavement. There are also a large number of possible edge candidates identified among the parked cars.

The bottom-left panel shows the result of the area segmentation algorithm. The intensity values represent the absolute difference between the *areavalue* of each box and that of a sample value from the middle of the road. Dark areas represent the greatest similarity. The bottom-middle panel shows the same image after thresholding and area-growing. The area growing algorithm requires each road candidate (dark area) to have at least two dark neighbours. With the exception of a small anomaly the resulting segmentation matches the road surface very well.

The final panel (bottom right) shows the result of combining the two techniques. It shows edge candidates that lie within 16 pixels (two box diameters) of the identified road surface. As we see, most of the edges identified are consistent with the road boundaries. There are a variety of approaches available for linking edge tokens into lines or smooth arcs. We discuss some of these in companion papers.

These algorithms take several seconds to execute for a 256x256 image on a single T414 transputer and would, therefore, require substantial parallelism to perform at video frame rates. The complete analysis of the image in this way is intended, at present, to give an initial measure of the primary driving surface and edges. Similar techniques could also be used as a background process analysing sample frames for the purpose of error recovery. In a later section we will look at a faster technique designed to analyse every frame at video rates.

A Covariance Texture Filter

One of the problems associated with differential filters is that they do not work very well with diffuse or broken edges or when two parallel edges lie close together (i.e. lane markings). The existence of two opposing parallel edges which are close together can suppress much of the response of differential filters if the two edges both lie within the filter window. Essentially the response is a superposition of the positive and negative pulses produced by the two edges, partially cancelling. In this section we explore an alternative approach to edge detection that may avoid this problem.

In an attempt to develop a filter that can cope with these situations we have turned to a texture method

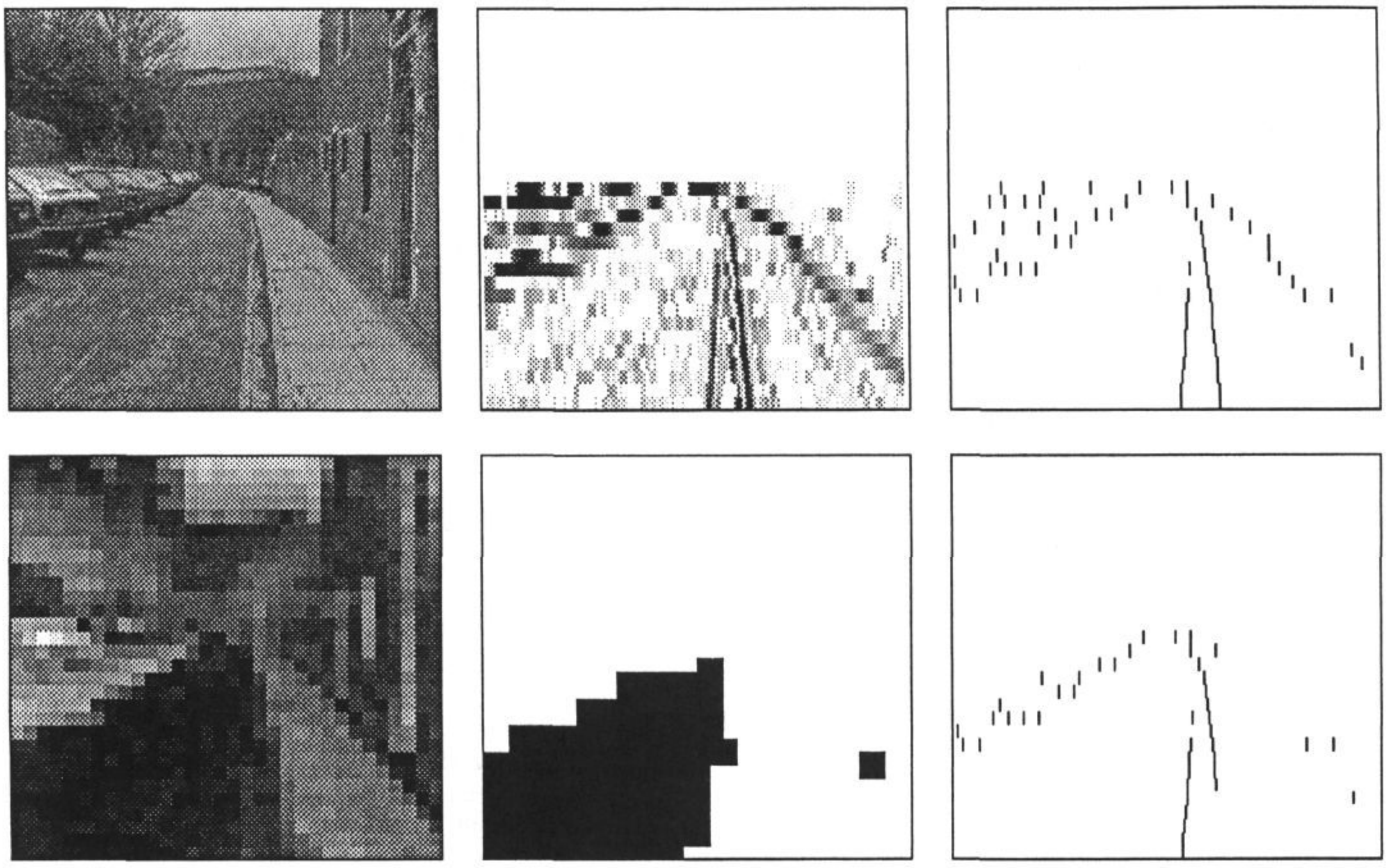
Figure 4: Analysis of a Road Scene

for searching for general linear features. The technique attempts to find regions in which the local variance is strongly anisotropic. It involves calculating the local grey-level variance in both the x-direction and the y-direction together with the covariance.

At any point $[i, j]$ in the image we can calculate the variance in the vertical and horizontal directions over a fixed range of surrounding pixels together with the covariance

$$
\begin{aligned}
(\sigma_x^2)_{ij} &= \frac{1}{n+1}\sum_{k=-n/2}^{n/2} v_{i+k,j}^2 - \left(\frac{1}{n+1}\sum_{k=-n/2}^{n/2} v_{i+k,j}\right)^2 \\
(\sigma_y^2)_{ij} &= \frac{1}{n+1}\sum_{k=-n/2}^{n/2} v_{i,j+k}^2 - \left(\frac{1}{n+1}\sum_{k=-n/2}^{n/2} v_{i,j+k}\right)^2 \\
(\sigma_{xy})_{ij} &= \frac{1}{n+1}\sum_{k=-n/2}^{n/2} v_{i+k,j}v_{i,j+k} \\
&\quad - \frac{1}{(n+1)^2}\sum_{k=-n/2}^{n/2} v_{i+k,j}\sum_{k=-n/2}^{n/2} v_{i,j+k}
\end{aligned}
$$

where $v_{i,j}$ is the grey-level intensity and n determines the size of the filter. To reduce noise these quantities are then smoothed over a small local region prior to further analysis. The variance anisotropy is determined by first forming a covariance matrix, at every point in the image.

$$\mathbf{S} = \begin{bmatrix} (\sigma_x^2)_{ij} & (\sigma_{xy})_{ij} \\ (\sigma_{xy})_{ij} & (\sigma_y^2)_{ij} \end{bmatrix}$$

This matrix is then diagonalised to find the eigenvalues and eigenvectors, $\lambda_1, \vec{V_1}, \lambda_2, \vec{V_2}$, by solving the equation:

$$\mathbf{S}\vec{V} = \lambda\vec{V}$$

The eigenvalues represent the maximum and minimum variances respectively and the eigenvectors give the corresponding directions of maximum and minimum variance. The ratio of the two eigenvalues is a measure of the local anisotropy.

Figure 5 illustrates a simple application of this filter, with $n = 8$, using the road images from figures 4 and 6. Here it has been used to find linear features purely by identifying regions of large variance anisotropy. In this analysis no account has been taken of the edge directions but they could be used, as previously, to suppress edges with the wrong orientation. As we see, particularly for the lower panel, it gives a strong response throughout regions containing multiple linear features.

This algorithm is still at the development stage and more work is required to assess its overall value. It would appear, however, to provide adequate sensitivity to road edges and overcomes the problems associated with parallel edge pairs. As with conventional first order derivative operators it provides orientation information. In addition it produces the average grey-level and local isotropic variance as a by-product and can, therefore, be used in area segmentation. Here we have passed the filter over every point in the image but it could be used to analyse bands, as in the previous section. It is several times more expensive in CPU usage than simple differential filters (since it involves products) . However, it is hoped

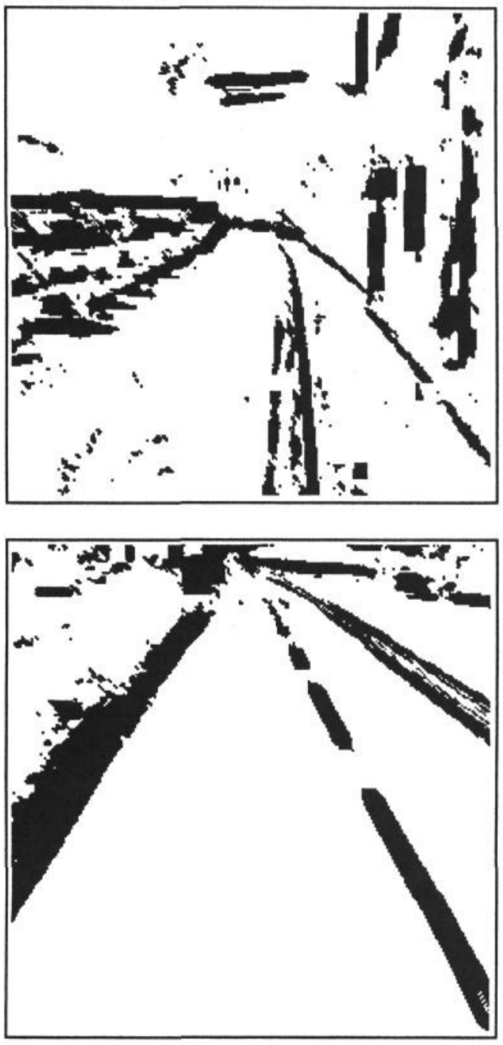

Figure 5: Edge Detection using the Covariance Filter

that its more general capabilities will provide adequate compensation.

REAL-TIME ANALYSIS

Video-rate analysis, within the current architecture, requires a substantial reduction in the computational complexity of the proposed algorithms. We have been experimenting with a number of approaches, some of which will be presented in companion papers. The approach we present here is a variant of the earlier differential edge-finder used in the bootstrap mode. Instead of analysing all adjacent bands in the image, the analysis is performed in a reduced number of horizontal strips. The bootstrap analysis will have already provided an approximate measure of the driving corridor and this mode of operation is intended to continuously confirm and update our knowledge of the road position.

One cannot entirely concentrate the analysis on the vicinity of known edges as the scene is continuously changing and the road may fork or contain obstructions. The method described here starts the analysis in the center of the known road and works outwards in both directions searching for the road edge.

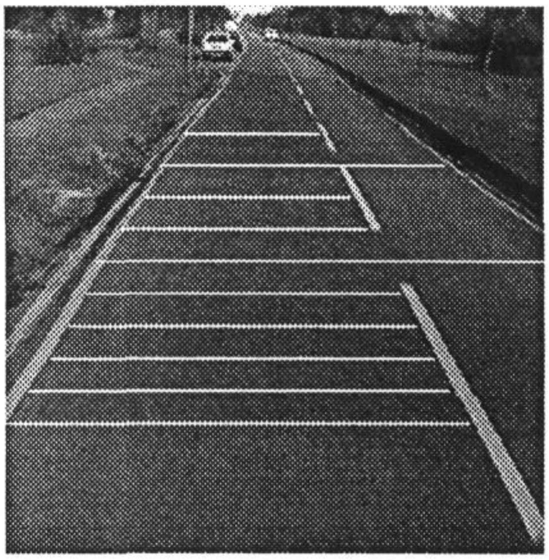

Figure 6: A Real-Time Analysis Algorithm

In this case only a vertical edge detector is used. It is a 5x5 mask of the form:

$$\begin{array}{|rrrrr|}\hline -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ \hline \end{array}$$

Figure 6 illustrates the method. Each of the white horizontal bars represents a single scan. The method is conceptually similar to the earlier technique except we have smaller filters, perform fewer scans and terminate the analysis when the edge is found. Each bar represents a single sweep with the differentiating filter, terminating when an edge response is found that exceeds the threshold value. In this case we have implemented a form of automatic thresholding. The central region of the road is assumed to contain no significant edges and is used to establish a statistical measure of fluctuations in response to the filter. Initially the central 20 pixels of the image are used to determine a variance in the filter response. Any further response along the scan line exceeding two standard deviations is then assumed to represent an edge. As the filter works outwards the variance is continuously updated so that a better statistical estimate is obtained.

This technique is still under development and we are not, as yet, comparing the results with any previous estimate of the road edge location. The scan stops as soon as a clear edge is located. This leads to problems, as we see, when the road contains lane markings. With extended white lines most of the scans end at a lane boundary, whereas for small markings most will extend the full width of the road.

In addition to differentiating the image we also calculate the average grey-level within each 5x5 window and it would be a simple extension to allow the scan to continue beyond a lane marker if the appropriate signatures are found in the grey-level averages and derivatives. One must be careful about blindly ploughing through obstacles, however, and a reasonable degree of higher level logic will be required before we can be satisfied that a potential obstacle is a standard road feature.

The major value of this algorithm, in the early stages of the development, is that one can safely drive within the region restricted by the scan lines. In an extended version of this technique a 5-frame running average is computed for the ends of each scan line. This smooths out the response to lane markings, yet still allows a sufficiently rapid response to parked or moving vehicles to allow safe navigation.

This algorithm performs at approximately 10 msec per scan (on average) on a single 20 MHz T414 Transputer. Thus, three transputers working in parallel can perform more than ten scans at video frame rates.

CONCLUSION

Here we have presented an overview of the computer architecture and low-level image processing algorithms we are developing to extract features from video images for the control of a vision-guided vehicle. The architecture is based on an array of Inmos transputers, a number of which have dual ported video frame stores in their address space. We have reported work in progress on the bootstrap mode of operation, designed to estimate the initial location of the driving surface. Two approaches have been used; the first using differential filters and segmentation techniques, the second using a texture based edge finder. We have further presented a simple but fast real-time algorithm intended to continuously update our knowledge of the driving corridor at video frame rates. Work is continuing on these and also other analysis techniques, some of which will be reported in companion papers.

ACKNOWLEDGEMENTS

We would like to thank Mr. Phillip Everson for his invaluable assistance with part of this work. This study has been funded by RARDE (Cherstey) under MoD contact ER1/9/4/2034/085.

REFERENCES

1. **Wallace, R., Stentz, A., Thorpe, C., Moravec, H., Wittaker, W., and Kanade, T.** "First results in robot road following" *Proc. IJCAI* Los Angeles, USA (1985).

2. **Wallace, R., Matsuzaki, K., Goto, Y., Crisman, J., Webb, J. and Kanade, T.** "Progress in robot road-following" *Proc. IEEE Int. Conf. Robotics and Automation* (1986).

3. **Le Moigne, J., Waxman, A.M., Srinivasan, B., and Pietikainen, M.** "Image processing for visual navigation of roadways" *University of Maryland Report, CAR-TR-138* (1985).

4. **Waxman, A.M., Le Moigne, J., Davis, L. S., Liang, E. and Siddalingaiah, T.** "A visual navigation system" *Proc. IEEE INT. Conf. Robotics & Automation* (1986).

5. **Dickmanns, E.D. and Zapp, A.** "Guiding land vehicles along roadways by Computer Vision" *Proc. AFCET, Congres Automatique* Toulous (1985).

6. **Mysliwetz, B., and Dickmanns, E.D.** "A vision system with active gaze control for real-time interpretation of well structured dynamic scenes" *Proc. Intelligent Autonomous Systems* Amsterdam (1986).

7. **Kuhnert, K.** "A vision system for real-time road and object recognition for vehicle guidance" *Advances in Intelligent Robotic Systems, Proc. SPIE* vol 727, Cambridge, Mass. (1986).

8. **Kuhnert, K.** "Comparison of intelligent real-time algorithms for guiding an autonomous vehicle" *Proc. Intelligent Autonomous Systems, Amsterdam* (1986).