COLOR IMAGE PROCESSING FOR NAVIGATION: TWO ROAD TRACKERS

D. Aubert * and C. Thorpe
CMU-RI-TR-90-09

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

April 1990

This research was sponsored by DARPA, Dod, and monitored by the U.S. Army Engineer Topographic Laboratories under contract DACA76-89-C-0014, titled "Perception for Outdoor Navigation". Portions of this research were also partially sponsored by the Digital Equipment Corporation External Research Program.
*D. Aubert is on leave of absence from ITMI (Industrie et Technologie de la Machine Intelligente).

Table of contents

List of figures

Table

Abstract

To build an autonomous vehicle capable of road following, it is necessary to compute the position and the orientation of the vehicle relative to the road. One way of doing so is to acquire images from a camera on the vehicle and extract some road features from these images. Knowing the geometric transformation between the camera and the ground plane, it is easy to compute the position and the orientation of the vehicle relative to these road features. In our case, we are building a system to track "structured roads", roads that have lane markings, shoulders, and other structure. For these roads, the easiest visual features to track are white lines and the yellow lines. To deal with a variety of roads, light and weather conditions, we need to build robust extractors of these features. In the case of the white stripe, we use our knowledge about its shape to extract it. We create a mask with a shape similar to the current white stripe and convolve a search area in the image with it. The maximum correlation gives us the location of the stripe. To reduce the signal/noise ratio, and to be robust in the cases of light and weather variations, we use a large mask. By taking into account the current running total during the convolution phases, we reduce the computational time drastically and so get a fast operator to track the white stripe. In the case of the yellow stripes, the color is a strong caracteristic of this kind of stripe, and the hue is only slightly affected by lighting and weather variations (we can get in the image a dark or a light yellow, but it is always a yellow color). We build the yellow stripe tracker based on the hue information.

I Introduction

The problem of autonomous vehicles has been studied more and more during the few last years. Some research deals with cross-country vehicles [3,9], other research deals with the problem of non-structured road following[8,11,12,13,14], or with the problem of structured road following [2,5,6,7,10,11]. In the latter case, the systems are generally developed for specific road conditions. However, to build a realistic autonomous vehicle the system needs to deal with various roads, light and weather conditions.

In our case, we want to drive a vehicle (The Navlab) automatically on real structured roads. The localization of the road features relative to the vehicle can be used to determine the position and the orientation of the vehicle. This will allow us to drive it without human intervention.

Numerous road features such as the following, exist: white stripes, yellow stripes, shoulders and guard rails. In this article we are interested in the extraction of the white and the yellow stripes. The extraction of these two kinds of lines will be done in two different ways, as we will see. In each case the methods are strongly related to knowledge about the feature.

To extract these two kinds of lines, we need to create an operator which is both fast as well as reliable in a variety of conditions.

- Fast
Speed is not really our goal, but it will give us certain advantages. We can test the behavior of the vehicle controller in conditions that push it to the limit, we are able to do more experiments, and faster means less change in the image, which makes the problem easier.

- Reliable in various conditions
Almost all existing systems deal with sunny, clean and well shaped roads. However, in reality these conditions almost never exist. The roads are dirty, the lines are almost erased, shadows cover most of the road. To allow us to experiment with a large variety of roads, the tracker has to be able to deal with such cases.

These trackers have been implemented, and tested on many images. We have used them to actually drive the Navlab on city streets at speeds up to 15 km/h.

The remainder of this paper is organized in three sections. In section 2, we will present the white stripe tracker. Then in section 3, we will discuss the yellow stripe tracker. For each tracker , we will discuss the effect of the light and the weather variations on the image information, describe the tracker and the tests, and discuss the results. Concluding remarks are the subject of section 4.

II White stripe tracker

1 Specific information useful for the extraction process

Although the road belongs to a real environment, the road by itself exists in a very constrained environment that has strong geometric shapes. The knowledge of the color and the geometric characteristics of the stripe are important. The more information we have, the better the interpretation of the scene will be. In the case of the white lines, while neither the color alone nor the shape alone are unique, the combination of color and shape provide very reliable feature extraction.

1.1 Color

The color of the stripe, white, is an extreme value both for intensity spectra in general and for road scenes in particular. In a classical vision system we sample the intensity between 0 to 255 (255 = white); white is the limit of this spectrum. On the road and in its close neighbourghood the white stripe is the only feature which has this color. The road itself is generally grey, while the border has grass which is either green or brown, or plain brown mud, depending on the season. The guard rails are silver or yellow .

1.2 Shape

The white stripes on the road are also geometrically constrained. The direction of the line is the same as the road and the line has a constant thickness in the scene. However, by perspective projection, the two parallel white stripes in a real scene become two converging lines and their thickness is no longer constant. Having calibrated the transformation between the road and the image, we can predict the angle and the thickness of the projected stripe at any particular row in the image. We explicitly know the shape of the stripe and can use this information to extract it.

2 Problems that distort the information

Image information is distorted by a variety of noise-producing factors. These factors will distort the color and the geometric characteristics, and so can make the extraction of the feature harder and less reliable. The major sources of noise are listed below.

2.1 Conditions affecting information accuracy

The white stripe in the image has two characteristics, its white color and its geometrical shape, which can be affected by various conditions. These conditions are: shadows, illumination changes (for instance: clouds going in front of the sun), atmospheric conditions (rain, snow ...), spots on the road (oil, dirt, mud, leaves), bad or weathered line painting. There are two kinds of shadows: homogenous and sparse shadows. The former is easy to deal with, especially with an auto iris camera. The latter is more difficult to deal with because such shadowing creates many regions in the images, some of which have the same color as the stripes.

2.2 Bad precision

Imprecise knowledge of the angle and the thickness of the line is introduced by errors in calibration and in vehicle motion. Errors in vehicle motion are due to uncertainties in the measurement devices (such as an odometer or an inertial navigation system).

3 White stripe tracker description

3.1 Definition of the tracker

Our operator has to take into account the available information described in section II.1, but also has to be efficient in spite of the problems introduced above. A good way to cover both aspects, is to create a template having a shape identical to the feature we want to extract. This template is then convolved with a window of the image, and the location with the highest correlation is the detected feature. In addition, instead of using red, green and blue bands, we use just the blue band. Experience has shown that this band gives us the best contrast between the line and the other features on the road.

The problems described in section II.2 decrease the signal/noise ratio during the correlation. To raise the signal/noise ratio, the operator has to cover a large range of rows.

Because the correlation is expensive to compute, especially with such a big operator, we lower its cost by computing it only for a few windows located along the predicted position of the line (This introduces another potential convenience, speeding up the algorithm by implementing each window's line extraction computation in parallel). To further lower computation costs, the template is composed of only two values: +1 for pixels that should match the stripe, -1 for other pixels. This way we avoid all intermediate products. To avoid weighting the computation, we need to have the same amount of +1's and -1's in the operator. This suggests a symmetric operator on both sides of the positive band, and implies a particular shape (parallelogram) for this operator. Templates with diagonal stripes are somewhat inconvenient to handle. Instead, we shift our image windows when we fetch them, so that the predicted stripe direction is always vertical. This allows us to use a single template for all road angles. Details are given below.

3.2 Implementation of the white stripe extraction

3.2.1 Global scheme

Three items enable us to predict a small area in the image in which we can find the white stripe. They are: The calibration, the knowledge of the previous position of the vehicle, and the knowledge of the motion of the vehicle between two successive images plus an assumption of continuity. We are also able to roughly predict the thickness of the white stripe for each line in the image.

To extract the white stripe from the image, we place windows along its predicted position. The parallelogram shape of the window comes from the predicted angle of the white stripe. We fetch each row of the predicted parallelogram, and write it into a rectangular buffer. This straightforward transformation gives, if the prediction of the angle is accurate, a vertical portion of the white stripe. To enhance the white stripe in the buffer, we only have to convolve it with an operator of the same shape, that is convolve the white stripe with a vertical band of "+1" of the predicted thickness of the stripe in the window. Thresholding in conjunction with region extraction [for instance the blob-color algorithm, ref. 1] will give us what we are looking for.

The scheme below summarizes the method (fig. 1):

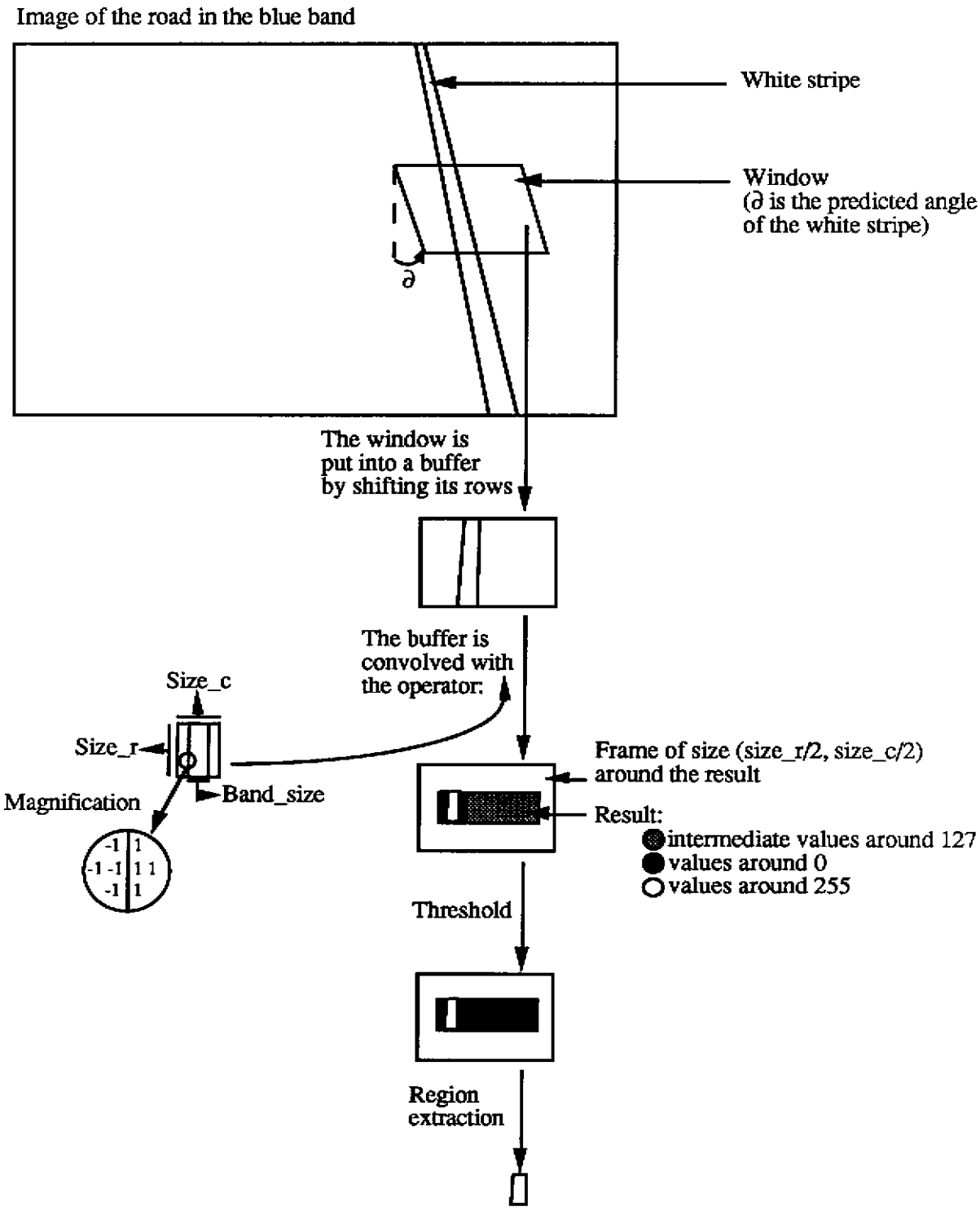


Figure 1: White stripe extraction scheme

3.2.2 Fast convolution computation

We can reduce the required computations by maintaining an array of column sums, and by maintaining running totals as we move across a row.

Calculate column sums:
Let the buffer have n rows and m columns.
Let band_size be the width of the positive operator band (band_size is always an even number).
Let the operator have size_r rows and size_c columns (size_c = 2*band_size).
n > size_r and m > size_c.
Let B(r, c) be the pixel at the position (r,c) in the buffer.

For each buffer column, sum the first size_r rows (fig. 2).
Let S(c) be such a sum for the column c.

$$S(c) = \sum_{j=0}^{size\_r - 1} B(j,c) \quad \text{,for c from 0 to m-1.}$$

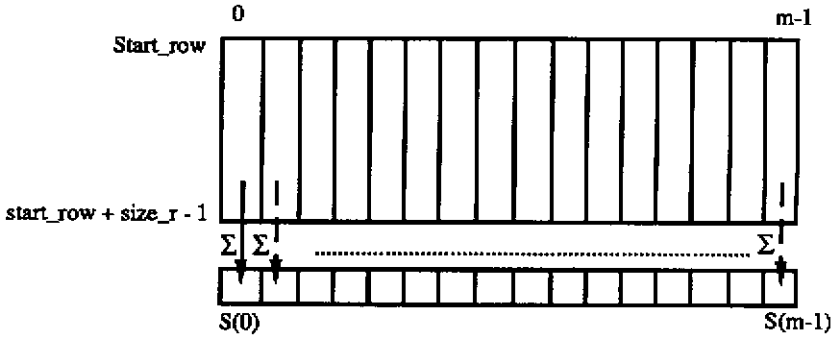


figure 2: Column sums computation

Initialize the running totals:
The negative part of the mask will start by occupying the first size_c/4 columns, the positive part the next size_c/2 columns, and more negatives in the next size_c/4 columns (fig. 3). Calculate the positive and negative sums. Their difference is the correlation value for the first position.

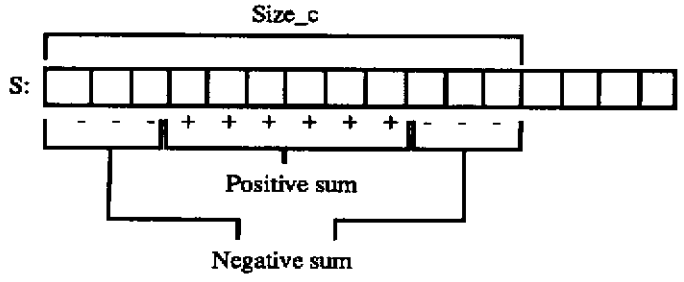


figure 3: Computation of the first correlation

The first correlation value is:

Positive_sum - Negative_sum, where

$$\text{Positive\_sum} = \sum_{i = 1/4*size\_c}^{3/4*size\_c - 1} S(i)$$

and

$$\text{Negative\_sum} = \sum_{j=0}^{1/4*size\_c - 1} S(j) + \sum_{k = 3/4*size\_c}^{size\_c - 1} S(k)$$

Step along rows:
By taking into account the previous computation of the correlation, the new one is done in few operations. The next new positive part of the correlation will be the same as the previous one, except that it will gain one more pixel value on the right and lose one on the left. Update the positive part with one add and one subtract (fig. 4).

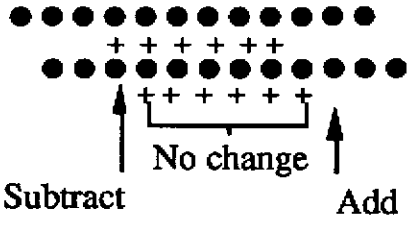


figure 4: Computation of the new positive sum

Update the negative part with two adds and two subtracts (fig. 5).

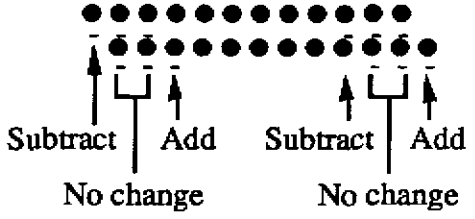


figure 5: Computation of the new negative sum

Output the new correlation value. Continue along the row.

Let c+1 be the new current column.
New Correlation = previous Correlation + 2*S(c + 3/4*size_c) - 2*S(c + 1/4*size_c) + S(c) - S(c + size_c).

Move down a row:
We update the column sums for the next row down, by subtracting the old top pixel and adding the new bottom pixel. This method lets us use masks of arbitrary size, with a marginal cost of 4 adds and 4 subtracts per mask position.

new S(c) = previous S(c) + B(size_r + r - 1, c) - B(r - 1, c).

3.2.3 Cost

From the line 0 to the line (n - size_r) we need to compute all the S(c). The first line costs m*size_r sums. Each of the (n - size_r) other lines will cost 2*m sums.

For each of the (n - size_r + 1) lines, the first convolution costs one(size_c + 1) sum, and 6 sums for each of the (m - size_c) other columns.

Then the total cost is:
m*size_r + 2*m*(n - size_r) + (n - size_r + 1)*[(size_c + 1) + 6*(m - size_c)].
Instead of (m - size_c + 1)*(n - size_r + 1)*size_r*size_c by a traditional way.

For instance the convolution of a (100, 100) buffer by a (50, 20) operator would cost only 40551 sums, instead of the 4131000 sums necessary if done in the traditional way.

3.3 Computational time

The experiment was done over 10 images, with 6 windows per image. For a 100 by 100 window, the average of the computational time of the stripe extraction is 0.11 second on sun 4/260 and 0.24 second on sun 3/260.

4 Tests

The goal of this experiment is to show the efficiency of the stripe extraction even if there is an error in angle, the variation of the band_size in thickness and its behavior on shadows.

We acquire two road images, one of them shaded. In both images, we take 3 windows along the right white stripe. The real thickness of the white stripe in the top window is roughly 8 pixels, about 10 pixels in the middle window and roughly 16 pixels in the bottom. For each image we apply the operator with a band_size of 6, 10, 15 and 20 pixels. Each operator is used with both the correct angle and 10 degrees of error.

The results are shown in the following tables:

Right angle

Stripe thickness	Mask thickness	Correlation values	
		Image without shadow	Image with shadow
8	6	**37.0**	**37.2**
	10	35.9	31.3
	15	25.0	32.3
	20	16.7	27.8
10	6	28.6	31.5
	10	**44.7**	**46.1**
	15	34.8	36.1
	20	26.7	27.0
16	6	27.6	26.3
	10	46.2	43.0
	15	**54.6**	**60.6**
	20	42.7	50.9

10 degrees of Error

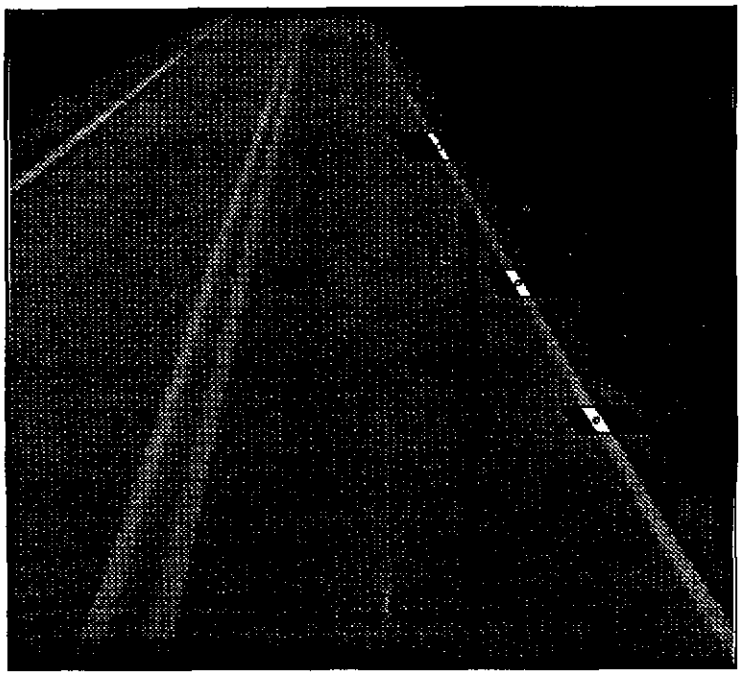
Stripe thickness	Mask thickness	Correlation values	
		Image without shadow	Image with shadow
8	6	15.9	16.6
	10	**24.4**	24.7
	15	23.4	**28.6**
	20	16.9	25.6
10	6	16.8	17.0
	10	26.4	28.9
	15	**32.0**	**33.9**
	20	26.3	27.3
16	6	17.7	16.6
	10	31.0	31.3
	15	**44.2**	**47.9**
	20	40.5	46.8

The correlation value is a ratio between the maximum value obtained during the correlation and the number of values in the positive band of the operator (which is: size_r*band_size). The bold-faced values designate the maximum correlation result obtained over all thickness experiments for a particular window.

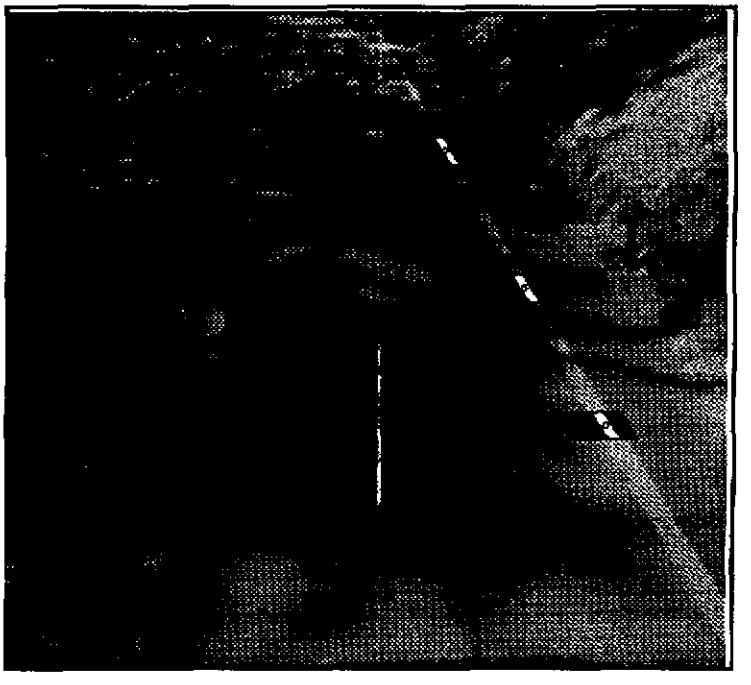
5 Consequences of these tests

From these tests we can deduce:
- The result is highly dependent on the variation of the angle, but we can still detect the white stripe with an error of 10 degrees.
- The thin stripes are more sensitive to the error in angle than the thicker stripes.
- We were succesful in detecting a white stripe in both shaded and non shaded road.
- By making the band_size a little bit wider than the real size, we get better results if the angle is bad and still get good results if the angle is correct.

6 Results on real roads



Picture of the road without shadow. The results in the 3 windows are displayed. The white regions in the windows correspond to the extracted stripe.



Picture of the road with shadow. The results in the 3 windows are displayed. The white regions in the windows correspond to the extracted stripe.

III Yellow stripes tracker

1 Specific information useful for the extraction process

1.1 Color

The color of the stripe is yellow; this is an important characteristic distinguishing it from other objects in the scene.

1.2 Shape

The white stripe's geometrical constraints also apply to the yellow stripes. The yellow stripes also have other constraints: the stripes are generally composed of two distinct, close, and parallel lines. This supplementary constraint is useful for identifying a yellow stripe.

2 Problems that distort the information

The same problems that affect the extraction of the white stripe also affect the yellow stripe, in addition to some problems inherent in the specific method used to extract such kinds of features. We base the extraction of the yellow stripe (see section III.3) on the color characteristic. The hue, being an approximation of the wavelength of the perceived color, is appropriate to perform the extraction. Hue is well characterized by the HLS color model (fig. 6). The hue, in this model, is the angle about the vertical axis from a reference hue axis (red = 0°).

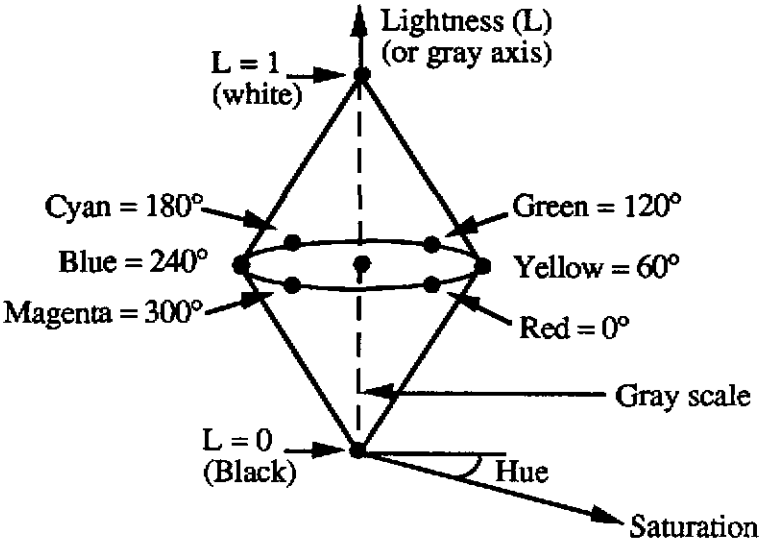


Figure 6: The HLS color model

Directly from the figure, we can see that it is impossible to define this angle for the values along the gray scale (Saturation = 0). That means, when the red, green and blue value are equal, we have a singularity in the color model. Also, due to mathematical precision and noise in the image, hue is not well defined near the gray axis.

Because grass, mud and guard rails are often yellow, the color yellow does not identify the stripes as well as the color white.

3 Yellow stripe tracker description

3.1 Definition of the tracker

Due to limitations in computation time, the extraction of these stripes is based solely on color characteristics. However, in the future, we may have to take into account the shape of the yellow stripes in order to distinguish them from other yellow objects, such as yellow grass. Currently, we relie on predicted tracker position to isolate the stripe from other yellow objects.

Due to the problems introduced in Section III.2, the hue for the pixels along the yellow stripes is sometimes not exactly 60°. That means, to perform the extraction, we need to take into account the possible variations in hue. The histogram of the distribution of the hue value along the yellow stripes gives us the extent of the variation (fig. 7,8).

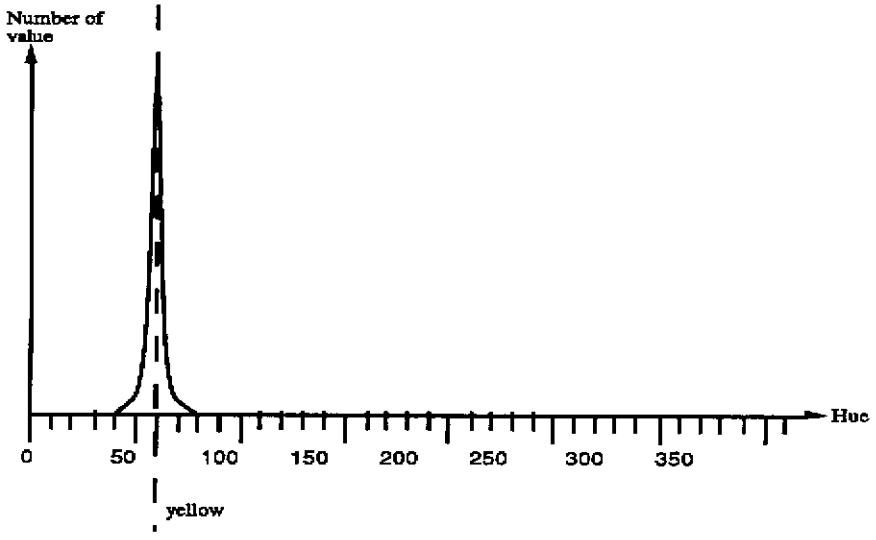


Figure 7: Histogram of the hue value for yellow lines without shadows in a sequence of images (camera sony model DXC-3000. Lens Fujinon-TV.Z). The values are symmetrically distributed around 60° with a variation of 20° on either side.

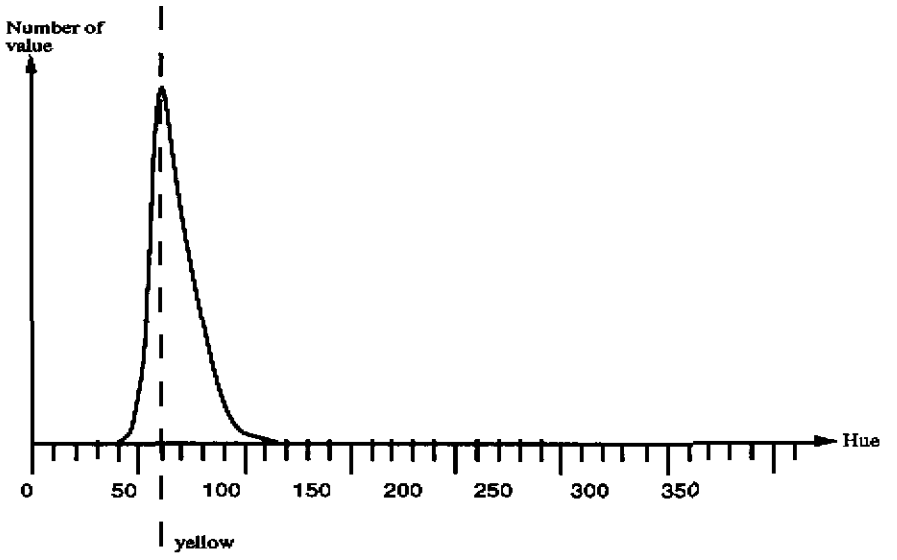


Figure 8: Histogram of the hue value for yellow lines with shadows in a sequence of images (camera sony model DXC-3000. Lens Fujinon-TV.Z). The values are no longer symmetrically distributed around 60°, but slightly skewed instead. The values lie between 40° and 120°, but there are very few values from 100° to 120°.

From these histograms, we choose to accept values between 40° and 100°.

Due to the problem of the "bad behavior" of the hue for the non saturated color, it is possible on the road (especially in the case of shadows) to have many scattered pixels that have a yellow hue but do not belong to the lines. To deal with this problem, we use a process of shrinking and growing.

To differentiate between the yellow stripes and other yellow objects such as grass, guard rails or mud, we carefully position and size the image working windows. In the future we will also use the contextual information of the position of the white stripes as compared to the yellow ones.

3.2 Implementation of the yellow stripe extraction

First we take several windows in the color image along the predicted position of the yellow stripes. For each pixel from a window we compute the corresponding hue by an efficient algorithm [4]. Then each value between 40° and 100° is set to 255, the others are set to 0. This process gives us a binary window with white regions corresponding to the yellow stripes, and some small white regions due to the "bad behavior" of the hue for the gray value. Such small regions are destroyed by using a process of shrinking and growing. Finally, an extraction of regions (for instance, the algorithm Blob color in [1]) gives us the portion of the line in the window. The scheme below summarizes the method (Fig. 9):

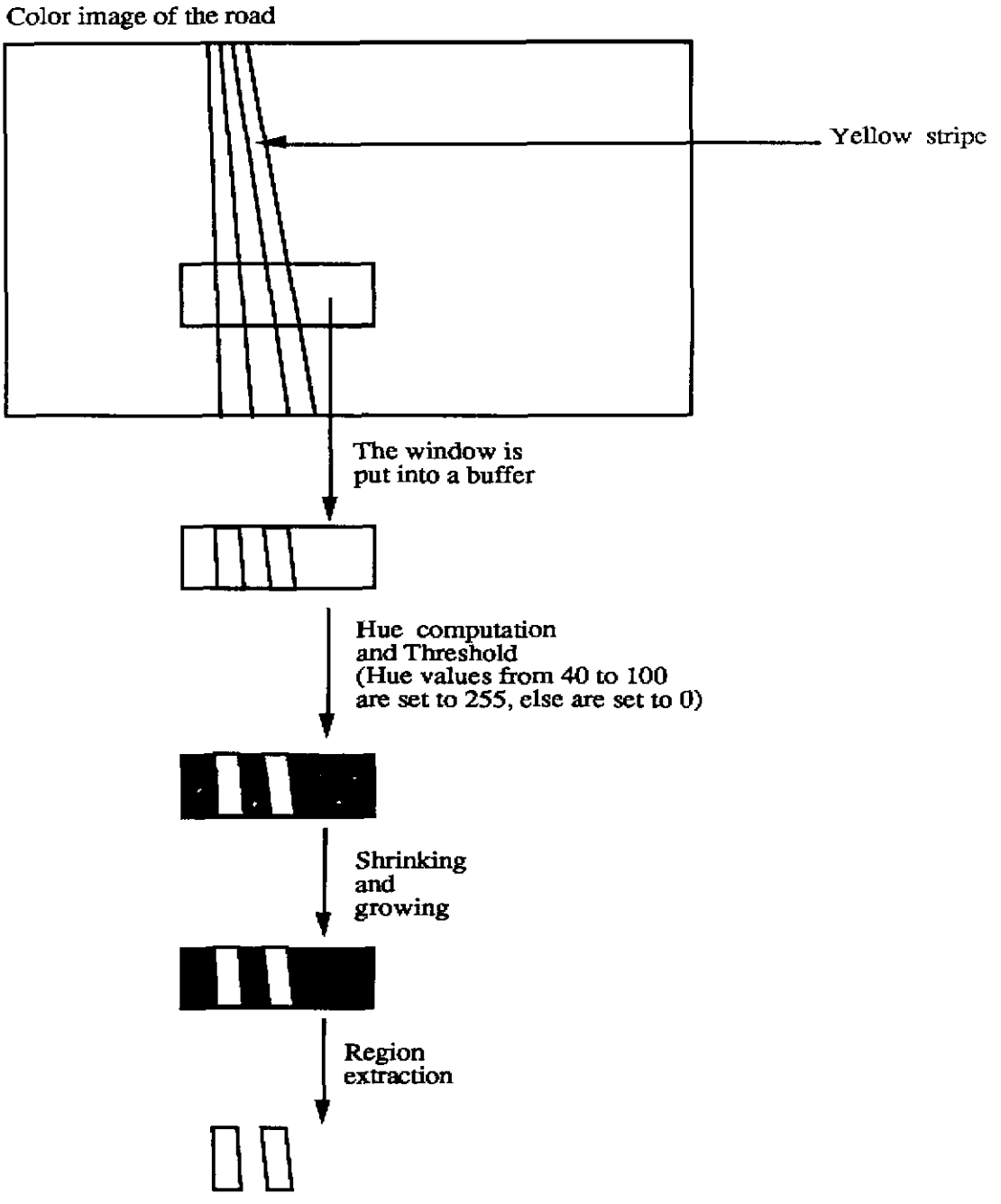
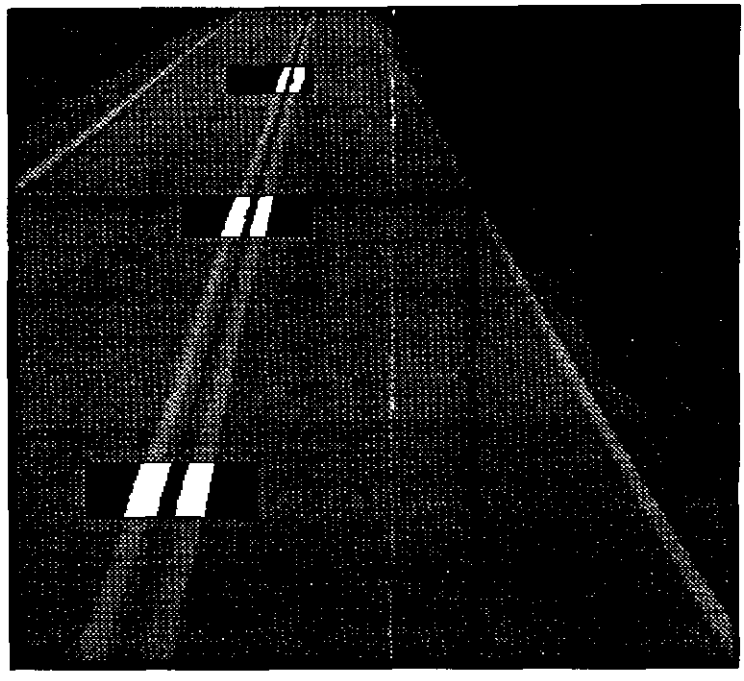


Figure 9: Yellow stripe extraction scheme

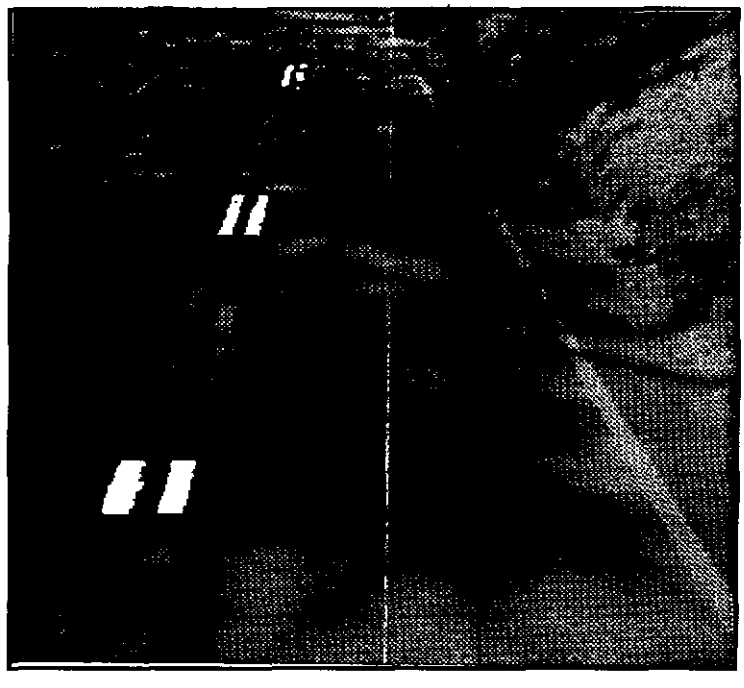
3.3 Computational time

Experiments were performed with a window size of 50 by 100. For any given sequence of images, the average computational time (to extract a stripe) is 0.57 second on sun 3/260, and 0.25 second on sun4/260.

4 Results on real roads



Picture of the road without shadows. The results of the 3 windows are displayed. The windows' white regions correspond to the extracted stripes.



Picture of the road with shadows. The results of the 3 windows are displayed. The windows' white regions correspond to the extracted stripes.

IV Conclusions

Our experiments using these trackers to drive the vehicle (NAVLAB) in various road, lighting and weather conditions, give us good confidence about their performance. However, the computational time to extract the stripes using a single sun4 does not allow us to drive the vehicle correctly at speeds higher than 15 Km/h.

To deal with this problem, we will compute the stripe extraction in parallel. The Navlab currently has 5 suns on board. We are building a parallel version of stripe extraction that assigns individual windows to each processor, and collects the results on a single master processor. The computation can be almost completely parallelized, so we anticipate near-linear speedup. Then we will push further by computing in parallel some specific phases of the stripe extraction. For instance, it is possible to parallelize the hue computation for each pixel (in practice for n pixels if we have n processors), as well as the thresholding step. In the case of the white stripe extraction, it is possible to compute in parallel all column sums (S(c)).

Acknowledgements

The authors would like to thank T. Kanade to give us the opportunity to conduct this research in his laboratory. Thanks also to K. Kluge and T. Druffel who are working on this project and to M. Hebert for his helpful discussions.

References

[1] D.H. Ballard and C.M. Brown,
Computer Vision,
Prentice-Hall, 1982, pp. 151.

[2] E.D. Dickmanns and A. Zapp,
A curvature-based scheme for improving Road Vehicle Guidance
by Computer Vision,
"Mobile Robots", SPIE-Proc. Vol. 727,
Cambridge, Ma., USA, Octobre 26-31, 1986.

[3] M. Hebert, I. Kweon and T. Kanade,
3-D vision techniques for autonomous vehicles.
Vision and Navigation, Edited by C. Thorpe, Kluwer Academic Publishers.

[4] D. Hearn and P. Baker,
Computer Graphics,
Prentice-Hall, pp. 304.

[5] S.K. Kenue,
LANELOK: detection of lane Boundaries and Vehicle Tracking Using
Image Processing Techniques.
Part I: The hough transform, Region Tracing and correlation Algorithms,
SPIE Conference on Mobile Robots IV, Phil., PA, Nov. 1989.

[6] S.K. Kenue,
LANELOK: detection of lane Boundaries and Vehicle Tracking Using
Image Processing Techniques.
Part II: Template Matching Algorithms,
SPIE Conference on Mobile Robots IV, Phil., PA, Nov. 1989.

[7] K. Kluge and C. Thorpe,
Explicit Models for Robot Road Following,
Vision and Navigation, Edited by C. Thorpe, Kluwer Academic Publishers.

[8] D. Kuan, G. Phipps and A. Chuan Hsueh,
Autonomous Robotic Vehicle Road Following,
IEEE trans. on PAMI, Vol. 10, No. 5, Sept. 1988.

[9] I. Kweon, M. Hebert and T. Kanade,
Perception for rough terrain navigation,
In Proc. SPIE Mobile Robots, Cambridge, Ma., 1988.

[10] T. Ozaki, M. Ohzora, K. Kurakashi,
Image Processing System for autonomous Vehicle,
Mobile Robots IV, SPIE, November 1989.

[11] D. Pomerleau,
Neural network Based Autonomous Navigation,
Vision and Navigation, Edited by C. Thorpe, Kluwer Academic Publishers.

[12] C. Thorpe and J. Crisman,
Color vision for road following,
Vision and Navigation, Edited by C. Thorpe, Kluwer Academic Publishers.

[13] C. Thorpe, M. Hebert, Takeo Kanade and S. Shafer,
Vision and Navigation for the Carnegie-Mellon Navlab,
IEEE trans. on PAMI, Vol. 10, No. 3, May 1988.

[14] M.A. Turk, D.G. Morgenthaler, K.D. Gremban and M. Marra,
VITS - A vision system for autonomous land vehicle navigation,
IEEE trans. on PAMI, Vol. 10, No. 3, May 1988.