Final Report 2002-21

Research

Algorithms for Vehicle Classification: Phase II

Minnesota Local Road Research Board

Technical Report Documentation Page

1. Report No. MN/RC – 2002-21	2.	3. Recipients Accession No.
4. Title and Subtitle ALGORITHMS FOR VEHICLE CLASSIFICATION: PHASE II		5. Report Date November 2001 6.
7. Author(s) Robert Martin, Osama Masoud, Suresh Gupte, and Nikolaos P. Papanikolopoulos		8. Performing Organization Report No.
9. Performing Organization Name and Address Department of Computer Science and Engineering University of Minnesota 4-192 EE/CS Building, 200 Union Street Minneapolis, MN 55455		10. Project/Task/Work Unit No. 11. Contract (C) or Grant (G) No. C) 74708 wo) 158
12. Sponsoring Organization Name and Address Minnesota Department of Transportation 395 John Ireland Boulevard Mail Stop 330 St. Paul, Minnesota 55155		13. Type of Report and Period Covered Final Report 2001 14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract (Limit: 200 words)

This report summarizes the research behind a real-time system for vehicle detection and classification in images of traffic obtained by a stationary CCD camera.

The system models vehicles as rectangular bodies with appropriate dynamic behavior and processes images on three levels: raw image, blob, and vehicle. Correspondence is calculated between the processing levels as the vehicles move through the scene.

This report also presents a new calibration algorithm for the camera. Implemented on a dual Pentium PC equipped with a Matrox Genesis C80 video processing board, the system performed detection and classification at a frame rate of 15 frames per second. Detection accuracy approached 95 percent, and classification of those detected vehicles neared 65 percent. The report includes an analysis of scenes from highway traffic to demonstrate this application.

17. Document Analysis/Descriptors Vehicle Classification Vehicle Tracking Vehicle Detection		18. Availability Statement No restrictions. Document available from: National Technical Information Services, Springfield, Virginia 22161	
19. Security Class (this report) Unclassified	20. Security Class (this page) Unclassified	21. No. of Pages 26	22. Price

Algorithms for Vehicle Classification: Phase II

Final Technical Report

Prepared by:
Robert Martin
Osama Masoud
Suresh Gupte
Nikolaos P. Papanikolopoulos

Artificial Intelligence, Robotics, and Vision Laboratory
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

November 2001

Published by:

Published by:
Minnesota Department of Transportation
Office of Research Services
First Floor
395 John Ireland Boulevard, MS 330
St Paul, MN 55155

Table of Contents

CHAPTER 1 - INTRODUCTION 1

OVERVIEW 1
BACKGROUND 2

CHAPTER 2 – CAMERA CALIBRATION 5

INTRODUCTION 5
APPROACH 5
SUMMARY 7

CHAPTER 3 – VEHICLE DETECTION 9

INTRODUCTION 9
APPROACH 10
ADAPTIVE BACKGROUND UPDATE 10
DYNAMIC THRESHOLD UPDATE 12
REGION TRACKING 12
VEHICLE CLASSIFICATION 14
SUMMARY 15

CHAPTER 4 – SHADOW HANDLING 17

INTRODUCTION 17
APPROACH 17
SUMMARY 19

CHAPTER 5 – CONCLUSIONS 21

SUMMARY 21
FUTURE WORK 23

REFERENCES 25

List of Figures

Figure 1 – Screen shot of GUICar with lane markings 6
Figure 2 – Flow diagram of detection and classification system 9
Figure 3 – Current frame and computed instantaneous background 10
Figure 4 – Region tracking. 13
Figure 5 – Scenes showing proper classification 15
Figure 6 – Scenes showing improper classification 15
Figure 7 – Improper classification due to the presence of shadows 17
Figure 8 – Shadow handled scenes. Note that peaks in the histogram correlate strongly with vehicle edges 18
Figure 9 – Successful shadow handling, before and after 19
Figure 10 – Screen shots of processed video from I35W. 22
Figure 11 – Scenes showing missed detection due to occlusion and improper classification 23

Executive Summary

This report presents a real-time system for vehicle detection and classification in images of traffic obtained by a stationary CCD camera. Vehicles are modeled as rectangular bodies with appropriate dynamic behavior. Processing of the images is done on three levels: raw image, blob, and vehicle. Correspondence is calculated between the processing levels as the vehicles move through the scene.

This report also presents a new calibration algorithm for the camera. Previously, one needed to know explicitly certain camera parameters such as height, zoom, and pan or tilt angle. A Graphical User Interface (GUI) interface has been implemented that allows a user to extract useful information from an image of the scene without the prior knowledge of those parameters.

This system was implemented on a dual Pentium PC equipped with a Matrox Genesis C80 video processing board. It was able to perform detection and classification at a frame rate of 15 frames/second. Detection accuracy approached 95% and classification of those vehicles detected neared 65%. Scenes from highway traffic have been analyzed and included to demonstrate this application.

Chapter 1 - Introduction

OVERVIEW

Traffic management has been an important discipline almost since the advent of the automobile. Information gathered has been used to such ends as keeping historical records of traffic volumes, predicting the need for repairs, preventing traffic congestion, and ensuring pedestrian safety [16].

Currently, traffic management systems rely on a variety of sensors to gather data. Starting in the 1940's, traffic flow was monitored with magnetic sensors and within twenty years the market expanded to also included ultrasonic and microwave sensors [16]. Today, magnetic loop detectors are the most common sensor and are used to count vehicles passing over them. However, there are a number of limitations to the effectiveness of loops that has caused researchers to look at other methods.

In addition to gathering vehicle counts, it is necessary to gather other statistics such as speed, lane changes, or vehicle type. Also, this data needs to be collected in all weather extremes and in a variety of traffic conditions. Loops simply do not provide the robustness needed for these requirements. They are prone to faulty readings under adverse weather conditions and sometimes cannot detect stopped or slowly moving vehicles.

In addition, loops cannot classify those vehicles it does detect. The ability to classify the traffic can improve the decision making process when it is again time to repair or replace a given road. Our classification system can provide this important data.

Our system uses a single camera mounted above a roadway. It works for multiple vehicle lanes and arbitrary vehicle direction. Once an image from the scene has been calibrated using the graphical user interface, there is no more interaction between the user and the application.

BACKGROUND

There is an extensive body of work in the area of vehicle detection using vision-based techniques. The thrust of the research has been towards monocular systems with a stationary camera although there have been attempts at other methods. One of the main problems that faces both areas is how to discriminate a moving object from the background. For that, various algorithms have been applied with success. Kilger [8] uses simple frame differencing to distinguish vehicles from the road. Vehicles are then treated as blobs with some coarse properties which describe the motion of a car. Friedman [5] also uses this method and notes that this technique has been around since the 19th century. There has also been success with optical flow, as in Masoud [11], although such methods are often computationally expensive. Dellaert [2] reports on a system that uses the image gradient to find the edges of objects of interest in a scene.

Another difficulty in segmenting objects from natural scenes is that of maintaining an accurate background image. Even small errors can lead to incorrect detection and erroneous classification. In [5] a technique is proposed that probabilistically classifies each pixel as foreground, shadow, or background. A similar method by [15] uses the time-varying characteristics of each pixel to assign a probability distribution to both foreground and

background pixels. Both [6] and [10] use a combination of the previous background and the current image to compute a new background image.

There is also a wide-ranging array of techniques used to detect vehicles from segmented images. In [2], vehicles are found by using the gradient to define a bounding box. Kalman filtering and error minimization are used to properly place and keep the bounding box around a vehicle. For detection and classification, [7] also uses gradients and attempts to match a 2-D parameterized vehicle template to the resulting edges for classification. It achieves above a 90% classification rate but is impossible for real-time systems. The work in [9] also uses parameterized templates, although the authors use a more general 3-D model. Some, like [4], ignore the whole vehicle and focus only on small features of the vehicle.

Chapter 2 – Camera Calibration

Introduction

In many vision-based classification systems, calibration of the cameras requires explicit knowledge of the camera position and orientation with respect to the road. Once away from the scene, these parameters are difficult to measure directly. It should be noted that even at the scene, the user may have difficulty in measuring these parameters. One way to compute them is to look at the known facts about the scene. For example, we know that the road, for the most part, is restricted to a plane. We also know that the lane markings are parallel and lengths of markings as well as distances between those markings are known numbers. We can use these to calculate the position and orientation of the camera with respect to those lane markings.

Approach

Camera calibration in general involves calculating the intrinsic and extrinsic parameters of the camera [12]. In general, the intrinsic parameters are the focal lengths in pixels along the horizontal and vertical axes of the image, the angle between these two axes, and the pixel coordinates of the intersection of the optical axis with the image. Since most cameras have right angles between the horizontal and vertical axes of the pixels, we can eliminate one of these parameters and greatly simplify our calculations. Also, we can assume that the optical axis goes through the center of the image, which further eliminates two more parameters.

The extrinsic parameters describe the pose (rotation and translation) of the camera in some real world coordinates. This adds six more parameters to calculate; there are the three rotations around the *XYZ* axes, respectively, and the three translations along those axes. If we assume that traffic is constrained to a planar surface, we can eliminate the two parameters that define that plane.

The Graphical User Interface (GUI) requires the user to first open a bitmap image of the scene. The user is then able to draw different lines and optionally assign lengths to those lines. From these lines, we can calculate the five unknown parameters. These numbers are then entered into the vehicle classification program to recover vehicle length and width. See figure 1 for a screen shot of the calibration program.

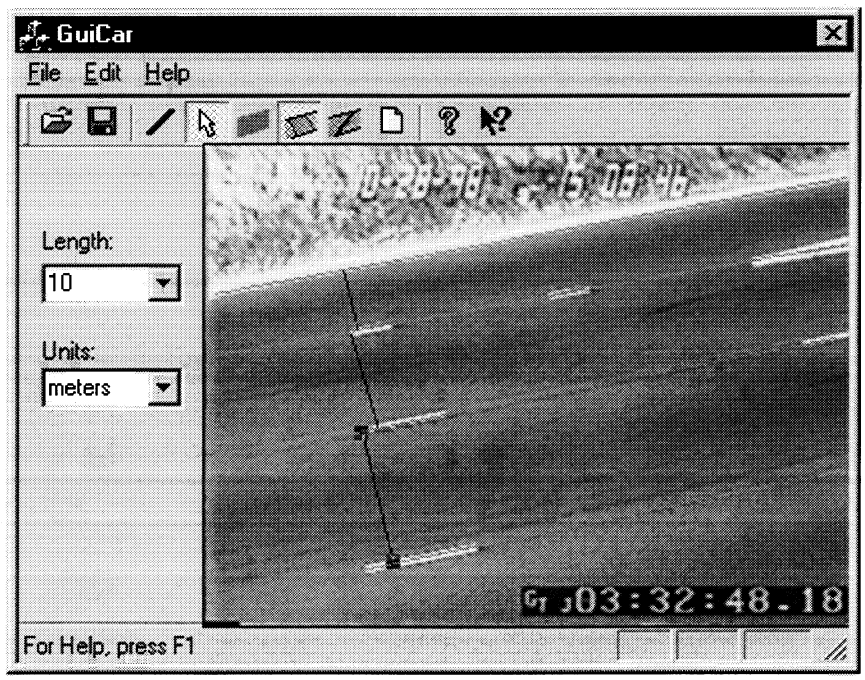

Figure 1 – Screen shot of GUICar with lane markings.

Summary

The GUI interface proves to be much more intuitive than the previous method. However, determine the vanishing point in some images proved to be a challenge. This can be improved by choosing images from the video sequence that improve the observer's perception of scene geometry. The other difficulty arose with respect to accuracy in determining distances in the direction of the road. Some of these inaccuracies arise because the markings on the road themselves are not precise. Another part of the inaccuracy arises because of the user's ability to precisely mark endpoints in the image.

In general, in spite of the inaccuracies discovered, this method of calibration proved to be much quicker than the previous method, and in general more accurate, and more adaptable to generic scenes.

Chapter 3 – Vehicle Detection

Introduction

This chapter describes the algorithm improvements in the vehicle classification project that we began earlier. Earlier work by our group focused on four areas of work: image segmentation, vehicle identification, vehicle tracking, and vehicle classification. A new method, adaptive background update, has been employed for better image segmentation. Rather then rely on a static background, we actively update the background as scene parameters (lighting, weather, etc.) change. Also, new methods have been tried in vehicle tracking and classification. As many of the segmented images do not contain the same number of blobs as vehicles, there is a problem of correspondence. We have tried an association graph approach with some success.

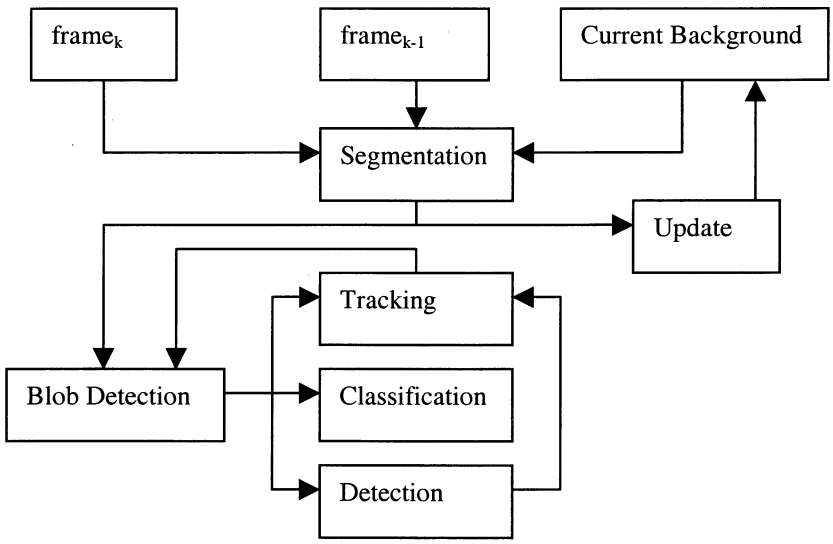

Figure 2 – Flow diagram of detection and classification system.

Approach

Adaptive Background Update

The basic principle of our method is to modify the background image that is subtracted from the current image (called the **current background**) so that it looks similar to the background in the current video frame. We update the background by taking a weighted average of the current background and the overall computed average background. However, the current image contains foreground objects. To remove foreground artifacts, we employ a binary object mask. This mask is calculated by simple frame differencing between the average background and the current frame. The object mask is used as a gating function that decides which image to sample for updating the background. At those locations where the mask is 0, corresponding to the background pixels, the current image is sampled. At those locations where the mask is 1, we again separate the pixels into two groups: those that also appeared in the previous frame and those that are new foreground pixels. For those pixels that appeared previously, we sample a buffer filled with the average pixel value of the background. For the remaining pixels, we sample the previous frame. This new image is called the **instantaneous** background. Figure 2 shows a current frame and its subsequent instantaneous background.

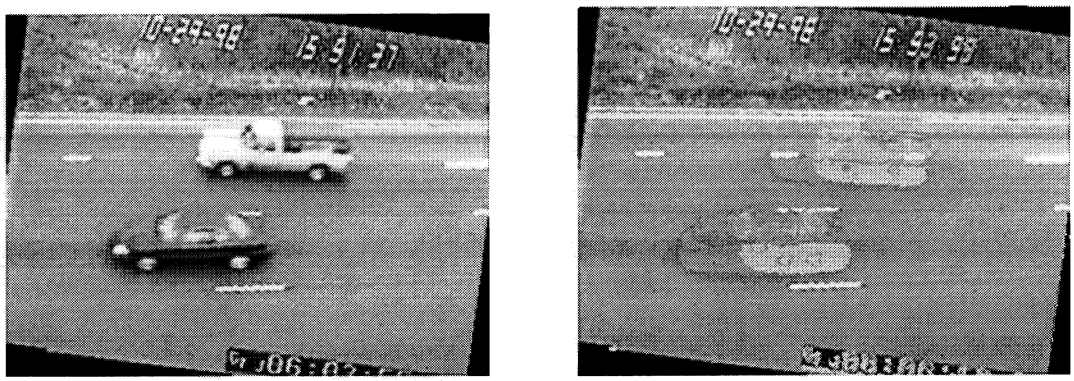

Figure 3 – Current frame and computed instantaneous background.

The current background is set to be the weighted average of the instantaneous and the current background.

$$CB = \alpha IB + (1 - \alpha)CB$$

The weights assigned to the current and instantaneous background affect the update speed. We want the update speed to be fast enough so that changes in illumination are captured quickly, but slow enough so that momentary changes (due to, say the AGC of the camera being activated) do not persist for an unduly long amount of time. The weight has been empirically determined to be 0.1. We have found that this gives the best tradeoff in terms of update speed and insensitivity to momentary changes.

We have determined experimentally that for the type of video data we tested our method with - video of freeway traffic - an update interval of one second gives good results. The update interval needs to be set depending on the speed of the foreground objects. It should be set high enough so that there is little overlap between foreground objects in the frames from two consecutive update intervals. With the update interval set at one second, we find that the current background converges to the background in the current frame within 4-5 seconds, an initially blank background converges to the background within 6-7 seconds. In practice, the first frame of the video sequence is used as the initial background, in which case the background converges even faster.

Dynamic Threshold Update

After subtracting the current image from the current background, the resulting difference image has to be thresholded to get the binary object mask. Since the background is dynamic, a static threshold cannot be used to compute the object mask. Moreover, since the object mask itself is used in updating the current background, a poorly set threshold would result in poor segmentation. Therefore we need a way to update the threshold as the current background changes. The difference image is used to update the threshold. In our images, a major portion of the image consists of the background. Therefore, the difference image would consist of a large number of pixels having low values, and a small number of pixels having high values. We use this observation in deciding the threshold. The histogram of the difference image will have high values for low pixel intensities and low values for the higher pixel intensities. To set the threshold, we need to look for a dip in the histogram that occurs to the right of the peak. Starting from the pixel value corresponding to the peak of the histogram, we search towards increasing pixel intensities for a location on the histogram that has a value significantly lower than the peak value (we use 10% of the peak value). The corresponding pixel value is used as the new threshold.

Region Tracking

A vision-based traffic monitoring system needs to be able to track vehicles through the video sequence. Tracking eliminates multiple counts in vehicle counting applications. Moreover, the tracking information can also be used to derive other useful information like vehicle velocities. In applications like vehicle classification, the tracking information can also be used to

refine the vehicle type and correct for errors caused due to occlusions. For an example of how this region tracking works, see Figure 4.

The output of the segmentation step is a binary object mask. We perform region extraction on this mask. In the region tracking step, we want to associate regions in frame i with the regions in frame *i+1*. This allows us to compute the velocity of the region as it moves across the image and also helps in the vehicle tracking stage. Since regions in frame *i* may split or merge with regions in frame *i+1*, there is a many-to-many correspondence that must be accounted for.

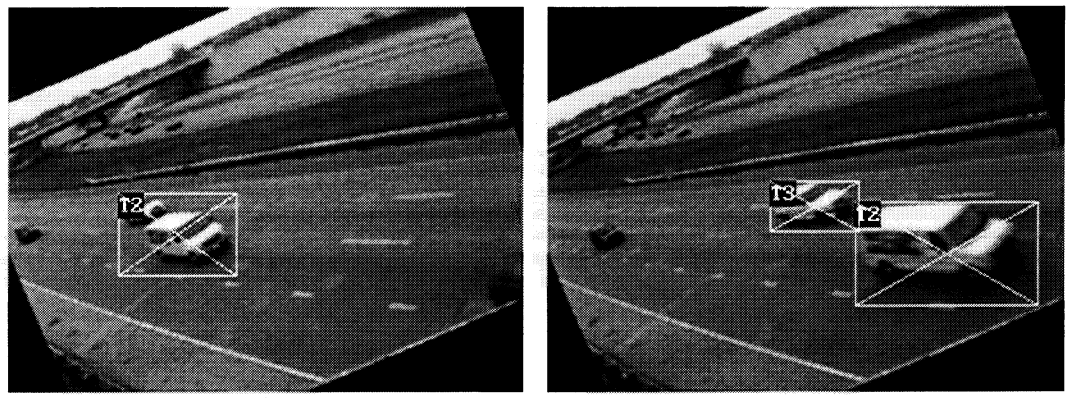

Figure 4 – Region tracking.
Both trucks were grouped as a single vehicle in the first frame. The second frame shows that region tracking correctly separated them into two distinct vehicles.

The region tracking method needs to be able to robustly handle these situations and work reliably even in the presence of these difficulties. We form an association graph between the regions from the previous frame and the regions in the current frame. We model the region tracking problem as a problem of finding the maximal weight graph. The association graph is a bipartite graph where each vertex corresponds to a region. All the vertices in one partition of this graph correspond to regions from the previous frame, and all the vertices in the other partition

correspond to regions in the current frame. An edge E_{ij} between vertices V_i and V_j indicates that the previous region P_i is associated with the current region C_j.

Vehicle Classification

The last step of this system is to take detected vehicles and to classify them as cars or trucks (non-cars). This classification is done based on vehicle lengths and widths. Using the information gathered from the camera calibration, the recovery of scene measurements from the image is easy and straightforward. Assuming a normal distribution for car and truck dimensions, one can use simple pattern classification algorithms to find a decision boundary in the dimension space. Figure 5 shows captured frames of scenes with vehicles that have been properly detected and classified.

The prior assumption in the camera calibration that vehicles are constrained to the ground plane does cause some errors. Mostly, these are caused by the modeling of vehicles as 2D entities. Since the camera is not positioned directly above the scene, it is difficult to accurately recover one or both of the dimensions as the height of the vehicle often adds to the length or width. However, as trucks are generally taller than cars, this error sometimes actually helps to separate the classes better. Figure 6 shows captured frames with errors in classification.

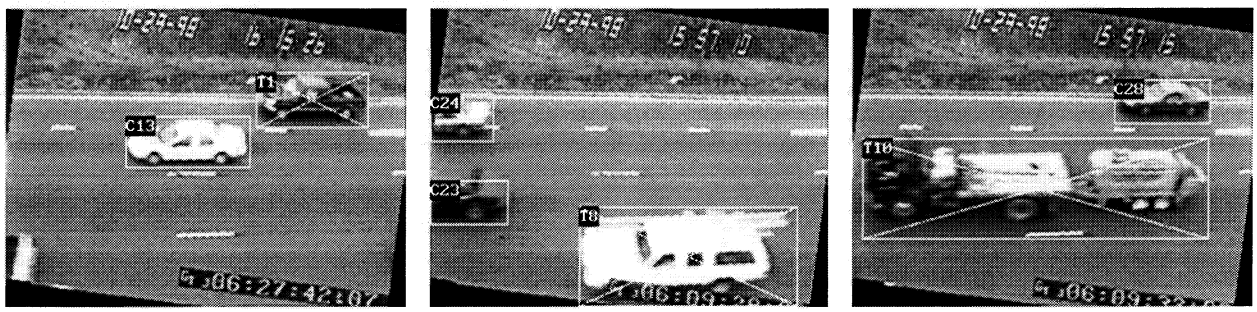

Figure 5 – Scenes showing proper classification.

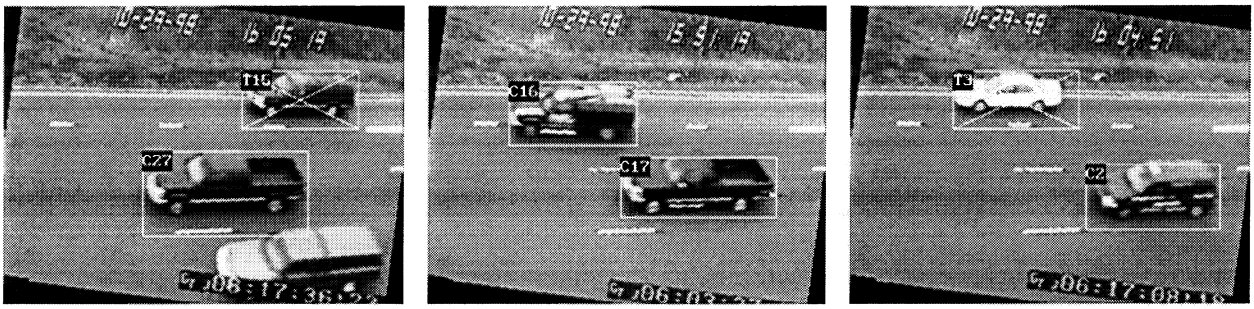

Figure 6 – Scenes showing improper classification.

Summary

Our new method of updating the background has enabled us to do a better job of segmentation. This, in turn, has reduced the error in the rest of the system. Also, our region tracking approach has improved our ability to accurately track vehicles as they move through the scene. Our classification rate remains less than hoped for despite our attempts to improve our camera calibration and implement a new discrimination algorithm. Our results and suggestions will be discussed in the conclusion.

Chapter 4 – Shadow Handling

Introduction

One of the great difficulties in monitoring traffic scenes under natural lighting conditions is to account for shadows. When a scene is undergoing segmentation, the cast shadows often distort the shapes of vehicles thus making it more difficult, if not impossible, to make an accurate analysis as shown in Figure 7. However, there have been attempts to limit the effects of strong shadowing in traffic scenes. A variation of the method proposed by [8] has been implemented in this system with some success. This has shown improvement in the classification rates of scenes involving shadows.

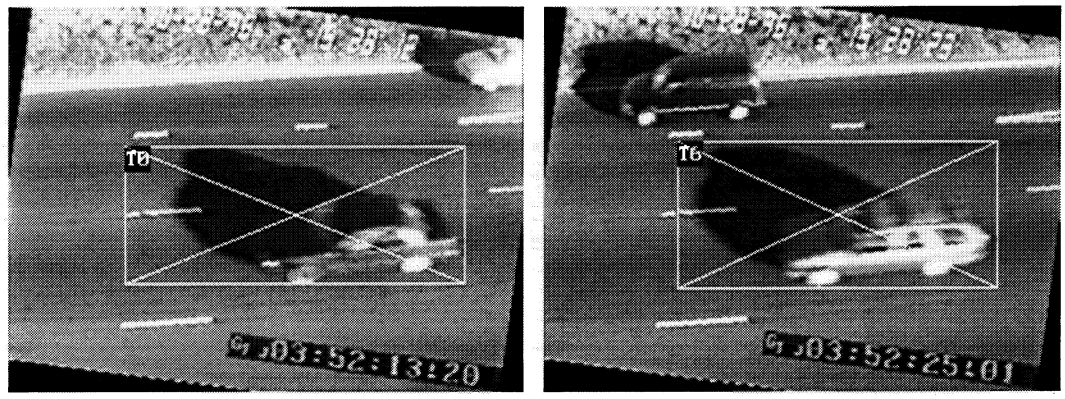

Figure 7 – Improper classification due to the presence of shadows.

Approach

It is known that even in shadows there exist detectable edges. We exploit this fact to shrink a region of interest to include a vehicle and ignore the effects of its cast shadow. If we do edge detection along one dimension of a region of interest we should expect to see stronger edges at a vehicle boundary and weaker edges at a shadow boundary. With this knowledge we can adjust the region of interest to discard extraneous areas due to shadows.

Our shadow handling is modular in that it can be turned on and off by the user when needed. When shadows are present and being handled, scenes are segmented into blobs that are assumed to include a vehicle and its cast shadow. Our shadow handler would then process these blobs and pass along information to the detection and classification components of this program.

Within each blob's region of interest, edge detection is done separately in both the x and y directions. The edges in each direction and then summed to give an edge histogram. One drawback to this method is that edges also exist in the background and can contribute to the histograms. To counteract this problem, horizontal and vertical edge detection is also done on the same region of interest in the average background image. The resulting background histograms are then subtracted from the foreground histograms. Figure 8 shows a shadow handled scene. The squiggly lines on the left side and bottom of the bounding box denote the relative strength of edges along the rows and columns, respectively.

We then search the histograms for significant peaks from which we can ascertain the vehicle boundaries. The most obvious method of finding peaks is to set a hard threshold. However, an invariant threshold favors larger and lighter colored vehicles. We favor a method in which the significant peaks are determined by the relative change in histogram values between neighbors.

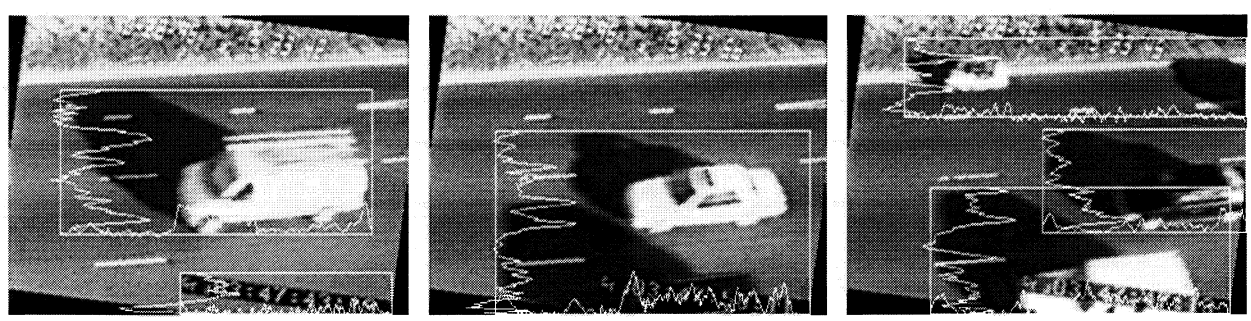

Figure 8 – Shadow handled scenes. Note that peaks in the histogram correlate strongly with vehicle edges.

Summary

Shadows generally need to be handled under strong illumination conditions. However, under these conditions image segmentation becomes more difficult and often results in an introduction of noise to the system. Our shadow handler then needs not only to deal with shadows but also with a degraded signal. Despite these difficulties, classification rates improve. See figure 9 for scenes showing correct shadow handling.

If shadow handling is to be a successful component of vehicle classification, we must continue to investigate other methods that will improve segmentation.

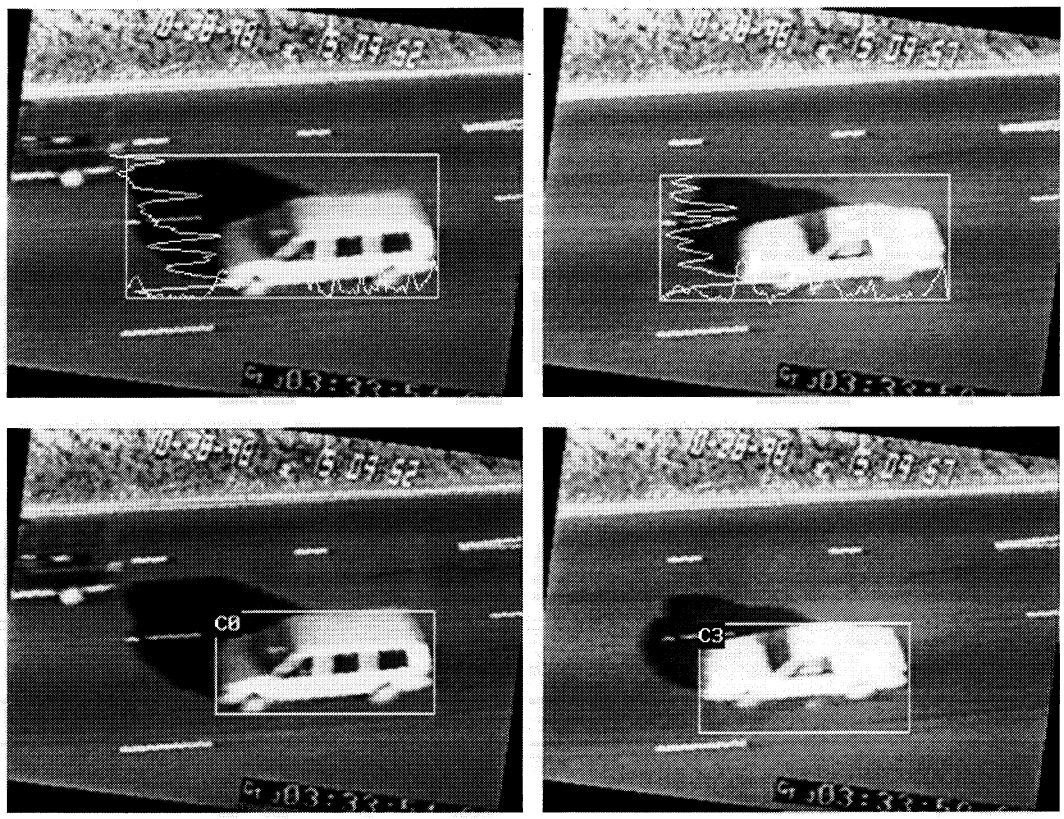

Figure 9 – Successful shadow handling (before and after shadow removal images).

Chapter 5 – Conclusions

Summary

We have implemented this system and tested it with video data provided by the Traffic Control Center of Minnesota. In almost all cases, vehicles were correctly detected. For those vehicles that were detected, the majority was properly classified as a car or a non-car. The provided video data contained artifacts (i.e., time/date stamps) that sometimes prevented detection and/or proper classification. The system also handled changing ambient intensity well although sudden and extreme changes in lighting affected the output for a few seconds. Our vehicle tracking methods also worked well although it was unable to distinguish occlusions.

We were also able to gather our own video data of traffic flow on Interstate 35W near the University of Minnesota. From this video we learned some new things. First, our program robustly handles traffic detection and monitoring in any direction. However, to get good classification, the scene needs to be rotated so that bounding boxes are aligned with the direction of travel. Second, our program can monitor traffic in multiple directions. Good examples of this are shown in Figure 10.

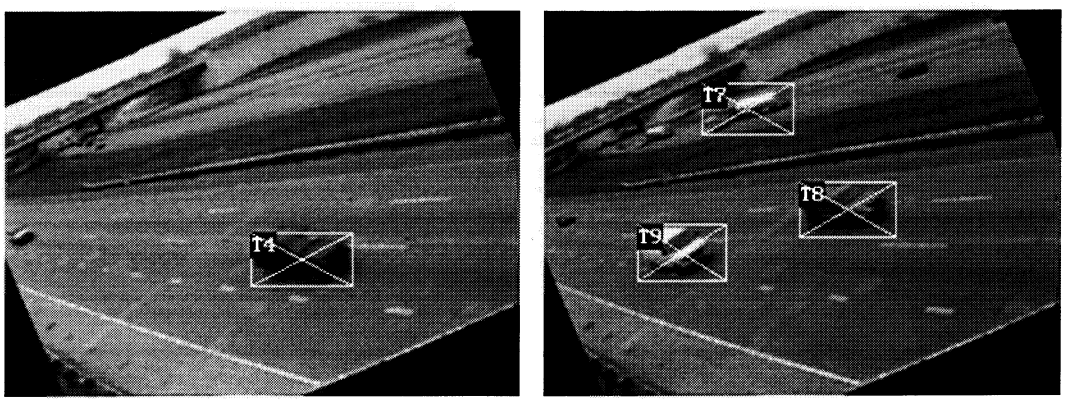

Figure 10 – Screen shots of processed video from I35W.
The picture on left shows a vehicle detected traveling left-to-right. The picture on right shows two vehicles detected in the near lane and a truck detected in the far lane. Other cars in the far lane were too small and were considered noise by the system.

As noted in [8], for proper traffic detection systems, there should be at least 100m of vehicle flow to analyze. In some of the near lanes of the TMS video, there was less than 25m and thus many vehicles traveled through the scene too quickly to be detected. In our I35W video, the perspective view did not allow many vehicles to be detected until they were halfway through the scene because of their size.

We ran the application on a series of two- to three-minute sequences of the provided TMS video. During an average sequence, 288 vehicles were counted, 247 of which were detected, giving a detection rate of 85%. Of those detected, 65% were properly classified, although there was greater variation than in detection (see Figure 11 for examples of erroneous classification).

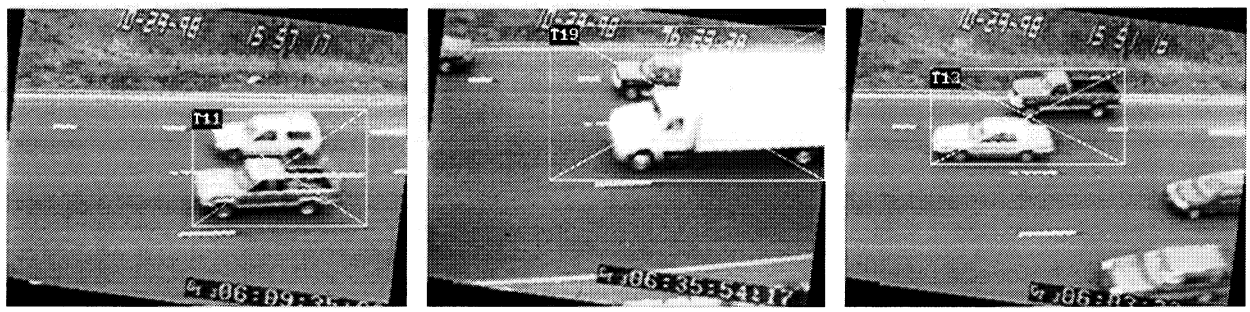

Figure 11 – Scenes showing missed detection due to occlusion and improper classification.

Future Work

One limitation of this system is that it worked best when traffic flow was aligned either horizontally or vertically. This allowed the bounding boxes to be aligned with actual vehicle dimensions. As [13] showed, it is possible to use oriented bounding boxes for real-time detection and the implementation of such may provide a more flexible application. Also, although vehicles are 3-D entities, they were modeled with 2-D bounding boxes. This forced the recovery of the vehicle dimensions to have a small degree of error. This perhaps explains some of the classification error. A better method would have been to use a 3-D bounding cube. This allows vehicle height to be recovered and thus one more dimension available for class discrimination.

There was also some error associated with the image rotation. Because rotation requires interpolation, many of the resulting segmented images contained more noise than the system could handle. If we implemented oriented bounding boxes, we could eliminate this noise.

References

1. R.Cucchiara, M.Piccardi, A.Prati, and N.Scarabottolo, "Real-time detection of moving Vehicles", in *Proc. of 10th International Conf. on Image Analysis and Processing (ICIAP 99)*, Venice, 1999.

2. F. Dellaert and C. Thorpe, "Robust car tracking using Kalman filtering and Bayesian templates", in *Proc. of the Conference on Intelligent Transportation Systems*, 1997.

3. R.O. Duda, and P.E. Hart, *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.

4. R. Fraile and S. J. Maybank "Vehicle trajectory approximation and classification", *British Machine Vision Conference*, 1998.

5. N. Friedman and S. Russell, "Image segmentation in video sequence: A probabilistic approach," in *Proc. of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 1997.

6. S. Gupte and N. Papanikolopoulos, "Vision-based vehicle classification", in *Proc. of the IEEE Int'l Conf. On Intelligent Transportation Systems*, Dearborn, MI, pp. 46-51, 2000.

7. M-P. D. Jolly, S. Lakshmanan, and A. K. Jain, "Vehicle segmentation and classification using deformable templates", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 293-308, 1996.

8. M. Kilger, "A shadow handler in a video-based real-time traffic monitoring system", in *Proceedings of IEEE Workshop on Applications of Computer Vision*, pp. 11-18, 1992.

9. D. Koller, "Moving object recognition and classification based on recursive shape parameter estimation", in *Proc. 12th Israel Conference on Artificial Intelligence, Computer Vision*, pp. 359-368, 1993.

10. D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning", Technical Report UCB/CSD 93/780, Computer Science Division (EECS), University of California, Berkeley, 1993.

11. O. Masoud and N. Papanikolopoulos, "Pedestrian control at intersections", Technical Report MN/RC – 1998/17, Minnesota Department of Transportation, 1998.

12. O. Masoud, "A camera calibration algorithm for traffic scenes", Technical Report, University of Minnesota – Twin Cities, 2000.

13. B. Maurin, “Optical flow for crowd tracking”, Technical Report, University of Minnesota, 2001.

14. I. Mikic, P. Cosman, G. Kogut, and M. Trivedi, “Moving shadow and object detection in traffic scenes", in *Proc. of the Int'l Conf. On Pattern Recognition*, vol. 1, pp. 321-324, August 2000.

15. C. Ridder, O. Munkelt, and H. Kirchner, "Adaptive background estimation and foreground detection using Kalman-filtering", in *Proc. ICRAM'95*, pp. 193-199,1995.

16. SRF Consulting Group, Inc., “Evaluation of non-intrusive technologies for traffic detection”, Report submitted to the Minnesota Department of Transportation, 2000.