2004 IEEE Intelligent Vehicles Symposium
University of Parma
Parma, Italy • June 14-17, 2004

Vehicle Wheel Detector using 2D Filter Banks

Ofer Achler, Mohan M. Trivedi

Computer Vision and Robotics Research Laboratory
University of California, San Diego
La Jolla, CA, 92093
oachler@ucsd.edu, trivedi@cvrr.ucsd.edu

Abstract

Detecting vehicles from a moving vehicle is an important task. In this paper a new vehicle detector is introduced. The new vehicle detector employs the use of the ubiquitous wheel. Every car has wheels; this wheel detector finds wheels and infers vehicle location from the wheel detection. Views from an omnidirectional camera are used to generate side view images. These images are processed using a difference of gaussian filterbank. The responses from the filterbank are applied to a precomputed set of principle components. The principle component responses are compared against a gaussian mixture model of wheels and gaussian model roadbed. Wheel candidates are chosen and tracked. Initial experimental results along with analysis are included.

1. Introduction

Driving safety is an important research objective. Computer aided driving is an attempt to aid the driver in situations that can get dangerous. Detecting vehicles around driver allows a computer to respond smartly to situations that can arise. Research to detect vehicles is done on numerous [1] vehicle testbeds. Most try to detect cars in front or in back; few try to track cars from the side view.

Using the side view can be cumbersome because a very wide field of view lens must be used. Additionally, two cameras are required to get complete coverage of the driver and passenger sides of the vehicle. In this paper, a single omnidirectional camera is used. Two virtual views from the sides of the car are generated [2]. The omnidirectional camera has the added benefit of allowing for as wide of a panoramic rectified image as desired.

In this work, the goal is to use a specific ubiquitous feature to track on all cars, wheels. All car wheels are round and have similar texture. Also, they are all framed similarly with fenders on top and roadbed on bottom. The goal is to take advantage of this information to form a robust wheel detector that can be used as part of a vehicle detector. An application of this work is for object avoidance. The algorithm can detect a wheel in the blind spot, or whether a wheel is getting too close for comfort.

1.1 Previous Work

Vehicle detection research varies from using active sensors to using passive sensors. Active rangefinding sensors have had success [3], but are quite expensive and tend to be single-purpose. They can track cars ahead, but cannot be used to detect lanes, road curve, road type, signs, and other obstacles [4].

Passive sensors (mainly, vision sensors) can be used for multiple purposes [1]. This paper deals with detection of vehicles, but the same camera can be used for detection of lanes, road curvature, road condition, signs, and many other data types.

Previous works have attempted to use PCA for front view classification of vehicles [5]. Z. Sun et al. use support vector machines [6] after using a gabor filterbank, template matching on rear view of the vehicles. M. Bertozzi et al. uses stereo correspondence matching to find vehicles [7]. All of these do not use a side view, and most do not single out a single feature that they are trying to detect. Our approach of using a difference of Gaussian filterbank alongside principle component analysis (PCA) is unique in that the view used is novel and the training is focusing on one particular vehicle feature, the wheels.

The idea of the difference of gaussian filters comes from how the brain interprets line segments [8]. Responses from the visual cortex have been recorded as similar filters. The idea is that these primitive filters combine to represent all that we see and can interpret.

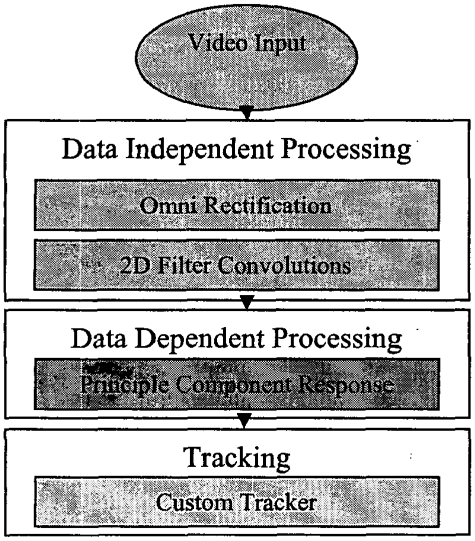
Detection of wheels is not much different than detection of the textures [9]. Using a combination of these primitive filters yields a response that is representative of edges and angles of edges. Since wheel lighting is not the

0-7803-8310-9/04/$20.00 © 2004 IEEE

primary goal of detection, using filters is a good idea. PCA directly on a scene has to automatically weed out all information not pertaining to wheels including lighting and information about parts of the vehicle far away from the wheel. It has a harder task than if we give a filterbank output that already weeds out some of the redundant and superfluous data.

2. Object Detection with Filterbanks

The vehicle detector has three stages. First is the data independent stage, then the data dependant stage, and finally, the tracking stage.


Video Input
Data Independent Processing
Omni Rectification
2D Filter Convolutions
Data Dependent Processing
Principle Component Response
Tracking
Custom Tracker

2.1 Data Dependant Processing

The omnidirectional camera is a useful tool to capture a full 360 degree view for processing. Using a single omnidirectional camera, one can see both passenger and drivers side of the vehicle. A problem arises from the way the image is formed. Distortion is generated by the mirror used to create this view and must be corrected for if a generalized template not dependant on location in the frame is to be used. The first stage in the algorithm is to correct for omnidirectional camera distortion. An appropriate camera view is chosen (parallel to passenger and drivers side and perpendicular to the ground). The second stage is to convolve the image with difference of Gaussian two-dimensional filters.

2.1.1 Omnidirectional Virtual Pan and Tilt

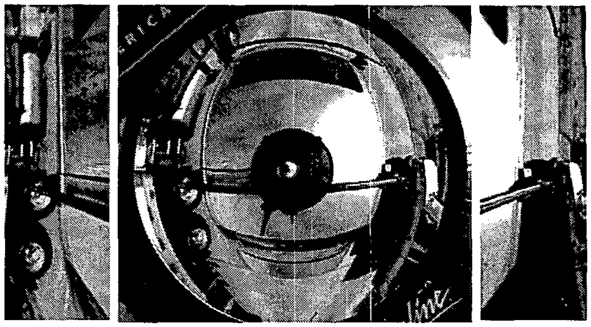


Figure 1 – Right and left images are selected views from the center omni image.

The omnidirectional camera sees all sides of the car simultaneously. Two views are selected parallel to the length of the car on either side. These yield two low-resolution views where wheels the same distance from the vehicle are all round (Figure 1).

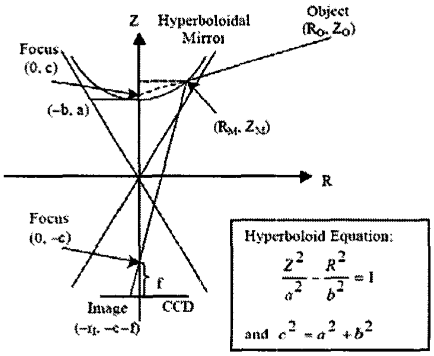


Figure 2 – Model of omnidirectional mirror in relation to the CCD

The mirror used in these experiments is a V-Stone hyperboloid mirror (Figure 2). Extracting a side view image from the omni-image is a matter of projecting the appropriate view (Figure 3) onto the ccd and picking out the pixels.

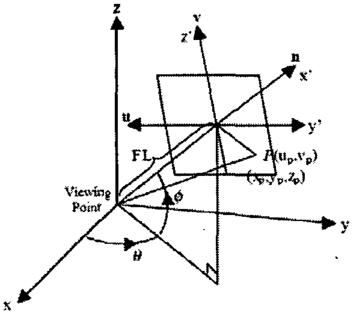


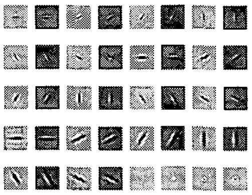
Figure 3 – Model of perspective view in relation to origin (where the omni mirror is located)

$$r_i = \frac{f\left(a^2 - c^2\right)}{\left(a^2 + c^2\right)\left(Z_0 - c\right) - 2ac\sqrt{R_0^2 + \left(Z_0 - c\right)^2}} R_0 \quad (1)$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi & -\sin\theta & -\cos\theta\sin\phi \\ \sin\theta\cos\phi & \cos\theta & -\sin\theta\sin\phi \\ \sin\phi & 0 & \cos\phi \end{bmatrix} \begin{bmatrix} FL \\ -u \\ v \end{bmatrix} \quad (2)$$

From equation (1) and equation (2), we know the location of each projected point possible in Figure 3. Using this knowledge, and recognizing the two side views are at $\theta = -90$, $\phi = 0$ and $\theta = 90$, $\phi = 0$ yield the results in Figure 1. This formulation of omni directional rectification comes from K. Huang and M. M. Trivedi [2].

2.1.2 Difference of Gaussian processing



The next step in data independent processing is to down-sample and convolve the image with orthogonal two dimensional filters.

Difference of Gaussians are chosen for ease of processing. Any orthogonal filter set (like gabor wavelets and steerable filters) can be used, difference of Gaussians are chosen for convenience.

The side images are down-sampled and convolved with each filter separately. The convolution with the filter bank yields a forty dimensional feature vector for each pixel. The feature vectors are then normalized for robustness against lighting changes. Dimensionality is reduced using data dependant processing.

2.2 Data Dependant Processing

Comparing forty dimensions to a single wheel as a model is not entirely effective. A way to create a generic wheel and road is required. Principle Component Analysis solves this problem. Dimensionality reduction and probabilistic models define a generic road and wheel model.

The principle components algorithm takes advantage of the variance of different groups. It diagonalizes the covariance matrix to find feature vectors that align to the highest variance within the data set. Assuming that the largest variance is between the wheel samples and road samples, it becomes easy to compare a wheel vector set rather than a single wheel vector to the data.

$$S = \left[\vec{s}_1 \ \cdots \ \vec{s}_N\right]$$
$$\vec{s}_i = \begin{bmatrix} f_1 \\ \vdots \\ f_P \end{bmatrix} \quad (3)$$

Training data is comprised of hand selected wheel center points along with road samples. A training matrix comprised of sample filter responses (P filters) is assembled (equation (3)). A covariance matrix is found and diagnolized (4) and (5). PCA is performed and the first L principle components corresponding to maximum variance between the road data and wheel data is found (6).

$$C = SS^T \quad (4)$$
$$C = UDU^T \quad (5)$$
$$U = \left[\vec{u}_1 \ \cdots \ \vec{u}_L \ \cdots \ \vec{u}_N\right] \quad (6)$$

It is reasonable to assume that the wheel data collected is a Gaussian mixture model. The means of the collected data classes are all different as are their variances. They all form Gaussian-looking groups that overlap in a 3d representation of the 6D space. The wheel samples and road samples are run through the principle components found, a Gaussian mixture model is applied to the wheel samples, and a Gaussian model is applied to the road samples. These probability density functions are used for comparison to each pixel signature in the incoming frame. A wheel candidate is tagged if it is more likely than the road and above a threshold.

Even if the true wheel distribution is not a Gaussian mixture, any probability distribution can be approximated by a Gaussian mixture model. To develop a good estimate of the means and variances for each of the classes in the mixture model, the Expectation Maximization algorithm is used [10].

$$p(y_k \mid \Theta_j) = \frac{1}{\sqrt{2\pi}|\sigma|^{\frac{N}{2}}} e^{-\frac{1}{2}(\bar{y}_k-\bar{\mu})^T \sigma^{-1}(\bar{y}_k-\bar{\mu})} \qquad (7)$$

$$\hat{x}_{kl} = \frac{p(y_k \mid \Theta_l)\hat{\alpha}_l}{\sum_{j=0}^{M} p(y_k \mid \Theta_j)\hat{\alpha}_j} \qquad (8)$$

$$\max_{\Theta_l}(E(L)) = \sum_{k=0}^{N-1} \hat{x}_{kl}(\ln(p(y_k \mid \Theta_l) + \ln(\hat{\alpha}_l)) \qquad (9)$$

$$\hat{\alpha}_l = \frac{1}{N}\sum_{k=0}^{N-1} \hat{x}_{kl} \qquad (10)$$

$$\hat{\bar{\mu}} = \frac{1}{N}\sum_{k=0}^{N-1} \hat{x}_{kl}\bar{y}_k \qquad (11)$$

$$\hat{\bar{\sigma}} = \frac{1}{N}\sum_{k=0}^{N-1} \hat{x}_{kl}(\bar{y}_k - \hat{\bar{\mu}})(\bar{y}_k - \hat{\bar{\mu}})^T \qquad (12)$$

The Expectation Maximization algorithm starts with expecting over each class. An estimate of the current sample being in each particular class is needed. Equation (8) is the expectation equation that needs to be calculated each iteration.

The maximization step requires an estimation of a sample being in a particular class. Using equation (8) allows us to solve equation (9). Equations (10), (11), and (12) are the solution for estimating the parameter of our probability distribution (equation (7)).

Because some of the wheel classes overlap, the EM algorithm is initialized with too many classes. The solution is to iterate and reduce nonsense classes the EM finds. A threshold is applied to the variance of each class. If the variance eigen values become too low, the class is removed and the training continues. Using this technique, 16 heuristic classes reduce to 13 converged classes.

2.3 Tracking

Once all the wheel candidates are found, a check on the history of wheels found occurs. This history check is used to validate the wheel candidates and classify them as wheels and not as noise detections. Also, tracking facilitates estimation of future vehicle position.

First, the probabilities of each pixel being a wheel are thresholded. Then, blobs that are too big or too small are removed as are blobs that are not round enough (eccentricity is too low). Centroids are calculated for each candidate blob and tracking continues.

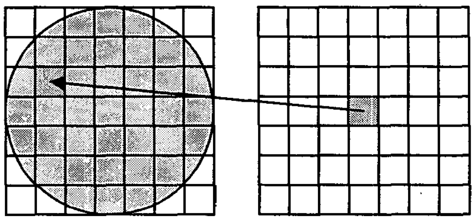


Figure 4 – green pixel is a wheel candidate

A small radius around a candidate wheel is scanned in the past frame. If a wheel is found in the previous frame, then the current wheel candidate is tagged as a wheel. This algorithm is applied to every wheel candidate. If multiple candidates are found, the highest probability of the found blob being a wheel is counted as the wheel. Since the blob centroid location has high variance, and the low relative speed of the vehicle tracked, a circle search window is needed as an ellipsoid search window would cause a loss of track of a wheel.

3. Initial Results

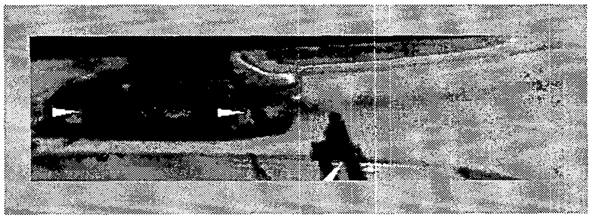


Figure 5 – Example of well detected wheels

These results are from the algorithm trained on 750 samples, of which 200 are road and 500 are wheels. The two test sequences include 3 pickup trucks, 2 sedans, and one big rig... The test sequence includes a pickup truck, two sedans, and a minivan.

In these initial results, wheels are detected with various degrees of success. Relative speed of vehicle can be attained. Using the CAN-BUS (Controller Area Network) data, absolute vehicle speed can also be attained.

Table 1: Sedan Sequence Results

	Detected Wheels	False hits on vehicle	Wheel Misses	Multiple Hits per wheel	False Positives
Non harsh conditions:	50	31	14	5	1
Harsh conditions:	9	14	28	0	3
Combined Conditions:	59	45	42	5	4

Table 1 corresponds to the sequence in Figure 5. The results show that wheels are detected 78% of the time in optimal conditions. Also, most false positives are on vehicle, so false positives were split into two classes. Another interesting phenomenon is that sometimes there are two hits on wheels. These appear due to the wheel blob being split and rejoining later.

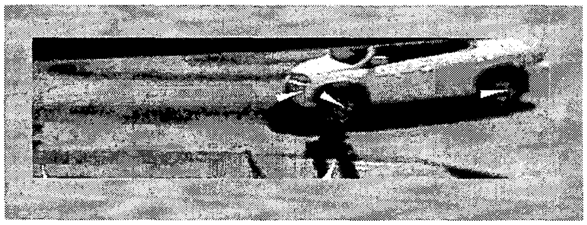


Figure 6 – Example of front wheel-like false positive

False positives on vehicles tend to be vehicle parts near road or at the edge of the frame (Figure 6, Figure 7). These spots do not always last throughout a video sequence for two reasons. They are not strong candidates, so it's not generally likely for four consecutive false wheel candidates to lie in the same region, they disappear and reappear which causes them to be discarded in the tracking phase. In the sequence Figure 5 came from, false hits were relatively rare due to the relative texture-less car and high contrast between wheel and vehicle.

Table 2: SUV Sequence Results

Detected Wheels	False hits on vehicle	Wheel Misses	Multiple Hits per wheel	False Positives
54	72	33	2	0

For the SUV sequence, there were no false positives that were not on the car or on the borders. The false hits on the vehicle were on consistent locations. One such location is the bumper as seen in Figure 6.

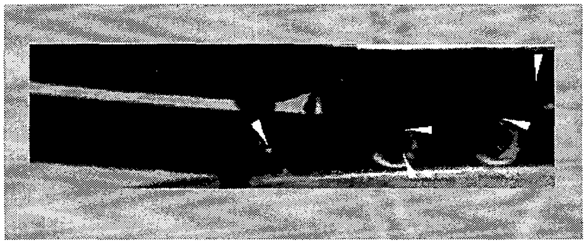


Figure 7 – Example of very textured scene causing false positives

Big rigs result in problems due to having so much texture, very harsh lighting changes, and very large wheels. Visually, the rear wheels are barely recognizable due to a strong overshadowing by the wheel itself and the large trailer as shown in Figure 8. The darkness coupled with texture yield ample opportunity for normalized signatures to be similarly to normalized wheel signatures. Therefore, a relatively large number of false hits per frame are observed.

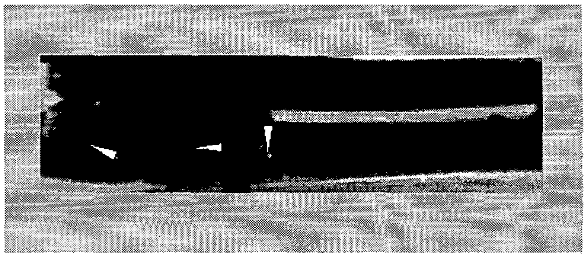


Figure 8 – Example of very dark wheels

The front wheel of the big rig is very rarely detected (Figure 9). This is because the wheel is large and the filters concentrate energy around the center. When the wheel is large, the center is missing a lot of important information. The signature becomes less representative of the actual wheel. Also, in these scenes, the wheels are close enough to the car to be partially occluded.

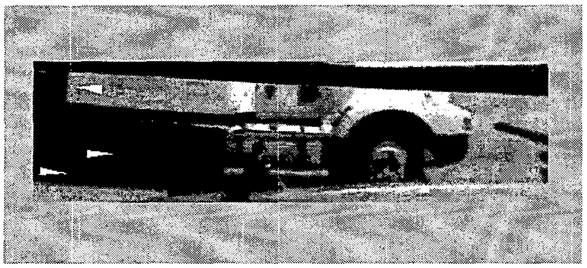


Figure 9 – False detection; Missed wheel

The big rig sequence has a high number of false hits that are detected on the vehicle. The false positives are generally on the shadows near the truck.

From the preliminary evaluation of the algorithm, the approach shows promise. Wheels can be detected in a variety of conditions and on a variety of vehicle types. Some conditions are problematic but more training and a better non-road model will improve them.

4. Conclusion

Driving safety is important to all of us. Computers can aid driver safety by knowing the situation around the car at all times. In this paper, a technique for detecting vehicles from a moving vehicle is proposed. Utilizing an omnidirectional camera, passenger and driver views are generated. A difference of gaussian filterbank is used to create feature vectors. These vectors are dimensionally reduced using PCA and a search for probabilistically best matching wheels occurs. The wheel candidates are then tracked.

Preliminary results are promising, future work includes applying a Gaussian mixture model to the non-wheel signatures, feeding false hits into the non-wheel training data, training on more data and classifying wheels by vehicle type. Future tracking will use a Kalman filter to track the wheels. A Kalman filter will then also filter wheel tracks together to find correlation between them. If there is a lot of correlation between wheels, that will be another indication that the system is tracking a vehicle.

Acknowledgments

The authors of this paper would like to acknowledge Nissan Motor Co. LTD and the UC Discovery grant for funding this research. We would also like to give special appreciation to our colleagues at the people in the Computer Vision and Robotics Research Laboratory, especially Joel McCall.

References

[1] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis, "A Survey of Video Processing Techniques for Traffic Applications", *Image and Vision Computing*, 21 (2003) 359-381.

[2] K. Huang, M. M. Trivedi, "Video arrays for real-time tracking of person, head, and face in an intelligent room," *Machine Vision and Applications, vol. 14, no. 2,* pp. 103-111, June 2003

[3] Romuald Aufrer, Jay Gowdy, Christoph Mertz, Chick Thorpe, Chieh-Chih Wang, Teruko Yata, "Perception for Collision Avoidance and Autonomous Driving", *Mechatronics*, Vol. 13, No. 10, December, 2003, pp. 1149-1161.

[4] K. Huang, M. M. Trivedi, "Driver view generation in real-time using a single omni video stream," *IEEE Transactions on Vehicular Technology, Special Issue on In-Vehicle Vision Systems*, Submitted August 2003

[5] N. D. Matthews, P. E. An, and C. J. Harris, "Speech and Intelligent Systems Vehicle Detection and Recognition for Autonomous Intelligent Cruise Control", 1995/6 *Research Journal Image.*

[6] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection using gabor filters and support vector machines," Intern*ational Conference on Digital Signal Processing*, July, 2002, Greece.

[7] M. Bertozzi and A. Broggi, "Real-Time Lane and Obstacle Detection on the GOLD System", Proceedings of *Conference on Intelligent Vehicles*, pages 213--218, Tokyo, Japan;, 19-20 September 1996. IEEE.

[8] R. A. Young, "The Gaussian Derivative Theory of Spatial Vision: Analysis of Cortical Cell Receptive Field Line-weighting Profiles", *Technical Report GMR-4920*, General Motors Research, 1985.

[9] J. Malik, S. Belongie, T. Lueng, J. Shi, "Contour and Texture Analysis for Image Segmentation", *International Conference in Computer Vision*, Vol. 17, pp. 1-100, 1987.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm", *Journal of the Royal Statistical Society.* Series B, vol.39, pp. 1-38, 1977