Analysis of Vehicle Recognition methods in Traffic Scenes

Rahul Garg
Computer Science Deptt
Indian Institute of Technology, Delhi

Under supervision of
Professor Subhashis Banerjee
Computer Science Deptt.
IIT Delhi

November 18 2005

Abstract

This research project aims at analyzing the various object recognition techniques and their application to a specific field of vehicle detection. The methods to be worked on include matching using SIFT (Scale Invariant Feature Transform), PCA (Principal Components Analysis) and Edge Based Features. The SIFT method was proposed by David Lowe to build some image descriptors of an image which can further be matched in this case for the detection a particular object. PCA technique, which has been used for face recognition is useful for exact objects but still needs to be explored for a generic object detector which would be needed in this case. Another technique is the identification of Edge Based Features in images and then use them for vehicle detection.

Introduction

Image matching is a fundamental aspect of many problems in computer vision, including object or scene recognition, solving for 3D structure from multiple images, stereo correspondence, and motion tracking. A lot of research has gone into this field in past time but has failed to yield a high quality vehicle detector. This is attributed to the fact that an image of the traffic scene in a very busy city like Delhi or Bangalore will be a cluster of occluded cars, bicycles sandwiched between cars, trucks etc. All this leads to utter confusion if we try to detect whole objects. Hence the aim is to produce a robust image processor which can recognize the different kinds of vehicles on roads in all visible forms. Now, we discuss in brief the techniques to be analyzed in the research:

1. Scale Invariant Feature Transform

This method proposed by David Lowe in January, 1999 successfully build local descriptors of an image. These descriptors are such that they help to identify the potential object forming features. The mathematical details of this descriptor building are described later in the document. These features are scale and rotational invariant. Now the feature set built from a set of training images is matched with the descriptor set built on the input image to produce a set of points which successfully match over a particular thresh-hold value. The matched keypoints are now clustered so as to form cars and other kinds of vehicles. The algorithmic details of this clustering strategy shall be described later in the paper.

The main advantages which the method potentially has:

(a) Scale Invariance: Helps in reduction of the training database size

(b) Rotation Invariance: Though not of much importance in the case of vehicle detection, it can be very useful for a generic object detector.

(c) Very careful about occlusion. A part of research already done by us shows that the technique is very robust and successfully identifies cars under a high rate of occlusion (as high as 60%).

2. Principal Components Analysis

This is a statistical technique based on the identification of the patterns in the data set, and expressing the data in such a format so as to highlight their similarity and differences. In this technique a n*n image is thought of as a vector with n^2 dimensions and the intensity value a point (x,y) as the value along the (n*(y-1)+x)th direction.

By this technique ,in various training images, the linear dependence in any two dimension is observed to get the pattern in images along those directions and thus help to find new axes (fewer in number) along which the data can be represented without much loss of information. Having represented the data of the training images in the suitable format the database is formed across which the test image is tested and its eigen distance is found from each image in the database.

2. Edge Based Feature Detection and Matching

Extracting information from edge picture of the image has not been explored thoroughly for the case of Vehicle detection. We have analyzed three sub methods in this case namely :

- Fitting a minimal 2D model on the high level edge representation
- Using a 3D model of car and using RANSAC for calculation of orthographic projection matrix and fitting of projection onto a high level edge picture
- Fitting projections on binary edge picture(raw output of an edge detector) using a voting method similar to generalized hough transform.

Related Research

A lot of research is going on in the field of object recognition today. A lot of techniques have been used for face recognition (now an official part of the OpenCV Library). But the work done in the field of vehicle detection in not yet comprehensive enough to enable the building of an automated system. Haar Based Classifiers were used by Mr. Gajinder Singh of IIT, Delhi in the year 2004-2005 to build a car recognition system following the AdaBoost technique. But the results still spoke of the inaccuracies like failure in the cases of occlusion and high false recognition rate. The studies are still going on around the globe to find a suitable, reliable and robust strategy for vehicle recognition which can then, if successful, be used to build a "Generic Object Detector" system.

METHODS

The following methods will be examined:

- **Scale Invariant Feature Transform**
- **Principle Component Analysis**
- **Edge based detection**

Distinctive Image Features from Scale-Invariant Keypoints

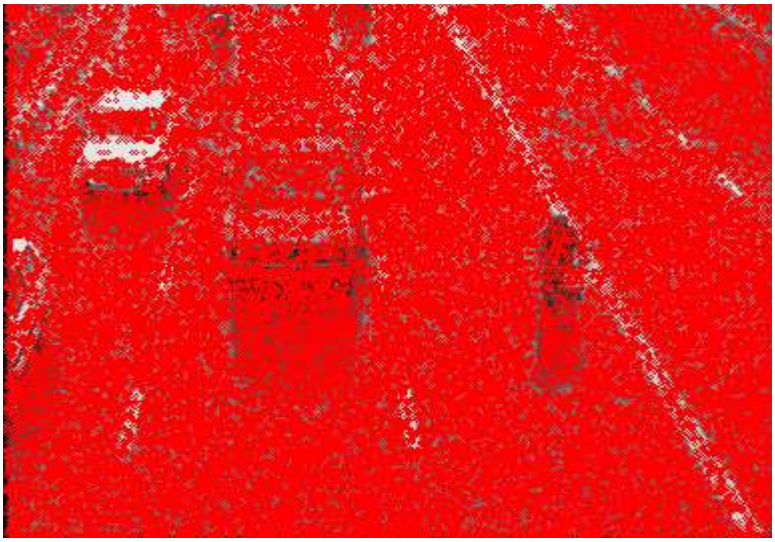
This approach presented by David Lowe in January,2004 has proved to be quite efficient and accurate in detecting cars in an image. The features detected are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise and change in illumination. These features make the technique appropriate for vehicle detection provided we keep the matching constraints sufficiently loose.

Finding the keypoints is a 4 step process described below:

1. Scale Space Extrema Detection
2. Keypoint Localization
3. Orientation Assignment
4. Descriptor Building

1.*Scale Space Extrema Detection*

A Difference of Gaussians(DoG) pyramid is built from the Gaussian pyramid. Then, an exhaustive search is performed over this pyramid to identify interest points over scale space



Scale Space Extremas Detected Initially; 152820 keypoints

2.***Keypoint Localization.***

A detailed fit is performed using the nearby data to accurately localize location and scale. The points with low contrast or poor localization are rejected. Moreover the ratios of principal curvatures around the keypoint are also observed and points with extreme values are rejected.

8358 keypoints left after rejection of keypoints with low contrast

7694 keypoints left after rejection of keypoints with extreme ratio of principal curvatures.

6191 keypoints left finally after rejection of keypoints which are not well localized.

3.***Orientation Assignment***

The orientation is also calculated at the (nearest) scale at which keypoint is detected. From the gaussian smoothed image of the appropriate scale (calculated earlier at the time of building of gaussian pyramid) gradient magnitude (m(x,y)) and orientation (theta(x,y)) is calculated as:

$$m(x,y) = \sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2}$$

$$\theta(x,y) = \arctan\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right)$$

where L is the Gaussian smoothed image. In fact, gradient magnitudes and theta are pre-calculated for efficiency.

To assign an orientation to keypoint, an orientation histogram is built from the gradient orientations of points around the keypoint. Bins of width 10 degrees are used and assignment is done after parabola fitting to the histogram peak. There can be multiple keypoints at the same location with different orientations if the histogram shows multiple peaks.

4.***Descriptor building***

First, a Gaussian window with sigma equal to one half the width of descriptor window is used to assign a weight to the gradient magnitude of each sample point. Individual histograms are then created over smaller square regions within the descriptor window. David Lowe suggests 8 bins for direction and 16 histograms in all. That leads to a feature vector of length 16x8=128.

The vector is then normalized and then to reduce the effect of illumination, the values in normalized feature vector are thresholded and the vector is renormalized. David Lowe suggests a threshold of 0.2 but values around 0.8 gave better results in vehicle detection.

For more details please refer to David Lowe's paper on SIFT.

Matching Process

This consists of the following steps:

1. Keypoint Matching
2. Clustering
3. Drawing the Bounding Box

Keypoint Matching

For matching purposes, Euclidean distance between the keypoints is compared. A keypoint is considered 'matched' using the nearest neighbor algorithm. Again, Lowe uses a distance ratio of 0.6 whereas a distance ratio of 0.8 gave better results (loose matching constraints). The loose constraints also lead to wrong matches, but they are removed during the subsequent clustering procedure (discussed below).

Clustering

Once we have matched keypoints from a scene to a training image (of a car), we need to identify the correct matches and cluster them into a car. David Lowe suggests clustering using Hough Transform. But another simpler method gave better results.

Since the cars in training images and scene are of nearly same scale, the correct matches will have a one-one correspondence between the training image and the scene i.e. if we place the two images side by side, the lines joining the correct matches will nearly be parallel and of similar length. So, by forming a r-theta

histogram of these lines for all them matches, and by subsequently thresholding the histogram allows a good identification of different cars.

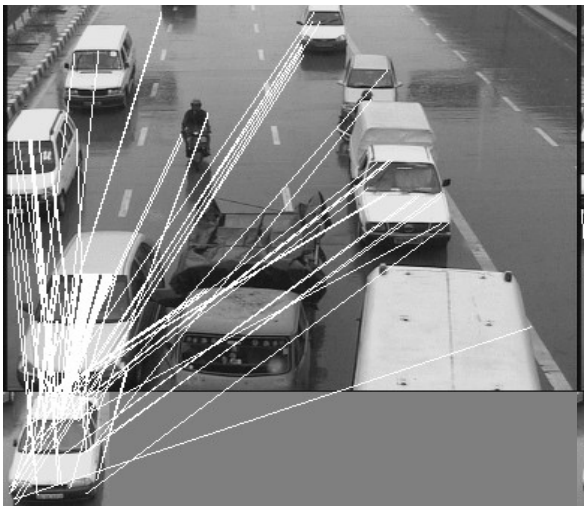


Illustration 1Initial Matching results

Illustration 2 Matches remaining after r-theta histogram based filtering

Drawing the bounding box

The bounding box is drawn by mapping back the matched keypoints to the training image and analyzing their position there. The size of the box is varied over the height of the scene as a function of y-coordinate of the left top corner of the bounding box. Note that many such boxes can occur while matching. So the filtering is done in the following manner.

The overlapping boxes over a particular thresh-hold of overlap are recursively removed in order of their keypoint matches. The final system utilizes training data from 55 training images and performs fairly well. Due to multiple training images, a car may be detected more than once leading to overlapping boxes for the same. Hence, if two boxes have a sufficient overlap, the one containing the more number of

keypoint matches is selected. Thus finally we get a set of **maximally isolated windows** which correspond to the separate vehicles with high probability.

Drawing Bounding boxes based on matches obtained above

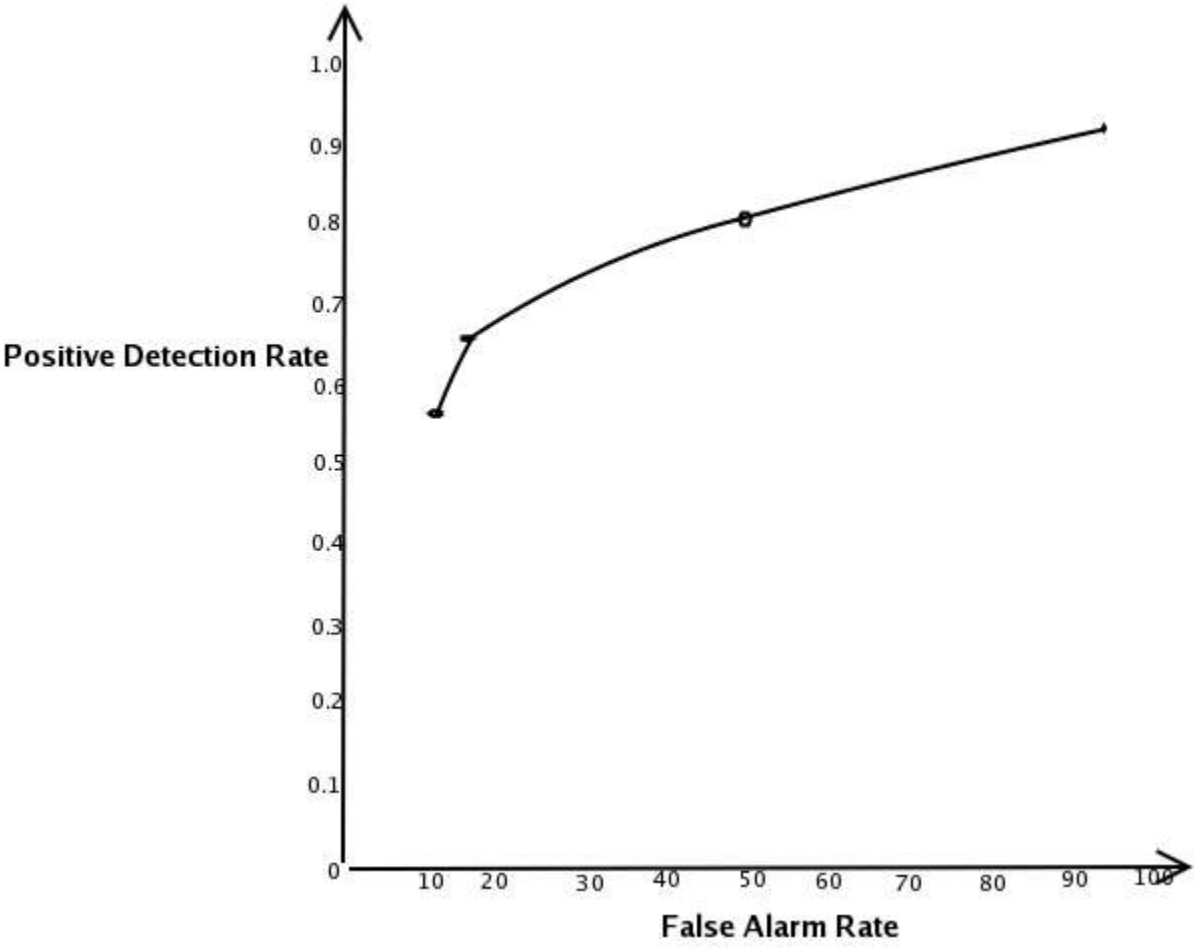
Analysis

There are two important parameters on which we analyze the ROC curve:

1. DoG Threshold:This threshold indicates the minimum contrast a valid keypoint should have.

2. Ratio of distances in the Nearest Neighbour Algorithm(While matching)

ROC curve for DoG Threshold



Total number of positives = 280

Positive Detection Rate = (True Positives / Total Number of Positives)

False Alarm Rate = False Positives

NOTE: Normally, the False Alarm Rate is the ratio of the false positives to the total number of negatives. But the total number of negatives is not defined in this case.

For the purpose of drawing ROC curve, DoG threshold was varied between 1.5 and 4.5. Generally, while drawing ROC curves of an object detector, input images normally contain one or two objects. But we considered densely populated scenes of traffic as input images for practical reasons. Hence, the detection rate may be slightly on the lower side for individual scenes. But detection results are better if the system is run on a video since a vehicle is bound to be detected in at least one frame and virtually no vehicle goes completely undetected. The results on video may be further improved by using temporal information. Moreover, the false positives were normally found to be on buses, trucks, etc.

Principal Component Analysis

This is a statistical technique based on the identification of the patterns in the data set, and expressing the data in such a format as to highlight their similarity and differences. In this technique a n*n image is thought of as a vector with n^2 dimensions and the intensity value a point (x,y) as the value along the $(n*(y-1)+x)^{th}$ direction.

By this technique ,in various training images, the linear dependence in any two dimension is observed to get the pattern in images along those directions and thus help to find new axis (fewer in number) along which the data can be represented without much loss of information. Having represented the data of the training images in the suitable format the database is formed across which the test image is tested and its eigen distance is found from each image in the database.

The individual steps of the process are described below:

1. Take a few training images (say about 60) of the cars of various shape and color and construct an average image by taking the mean of the intensity at every pixel. Subtract this average image from each individual training image to form a new database.

2. Form the covariance matrix of this new data set and find the eigenvectors and eigenvalues of this eigenfunction (covariance matrix).

3. Reject the eigenvectors with low eigenvalues since they mean that along the direction of that eigenvector there is no significant component of the vectors in the database.

4. Now the features of the training dataset are captured in the new set of axis defined by the eigenvectors. And if any test image has to be checked across the database, the average figure is subtracted from the test image and the Euclidean distance of the image from the database is found. If it is less than a

certain threshold then the image is a positive image otherwise a negative image.

Results of using this technique on the given problem

A few tests were carried out with this approach with the training database of size of about 70 and the nearest four neighbours were observed in the training database. Since some of the test images we took were a part of the training database as well, so in those cases the nearest neighbour was the image itself.

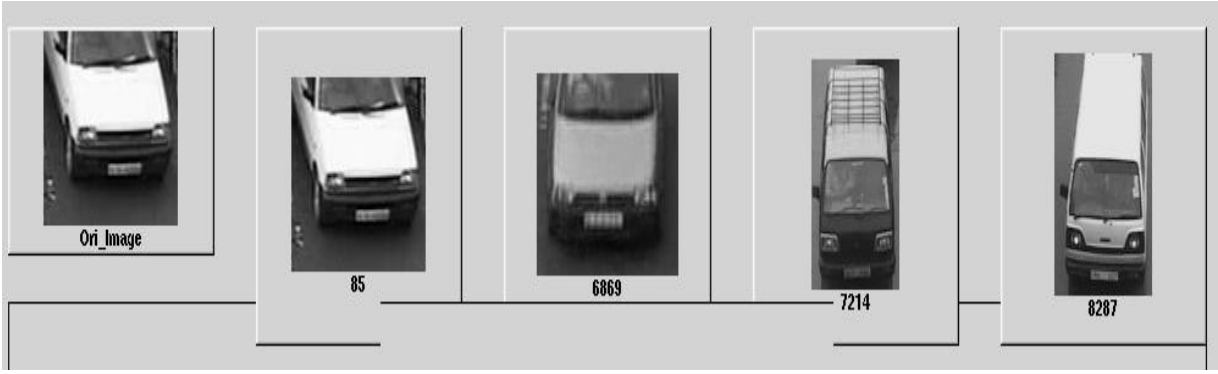


Figure 1: The numbers below the car images indicate their Eigen distance from the input (left most) image. The second image from the left is the same as the input image

The above result clearly signifies that the nearest matching image was at a distance of more than 6000 units from the test image (despite the existence of sufficient instances of white cars in the database).

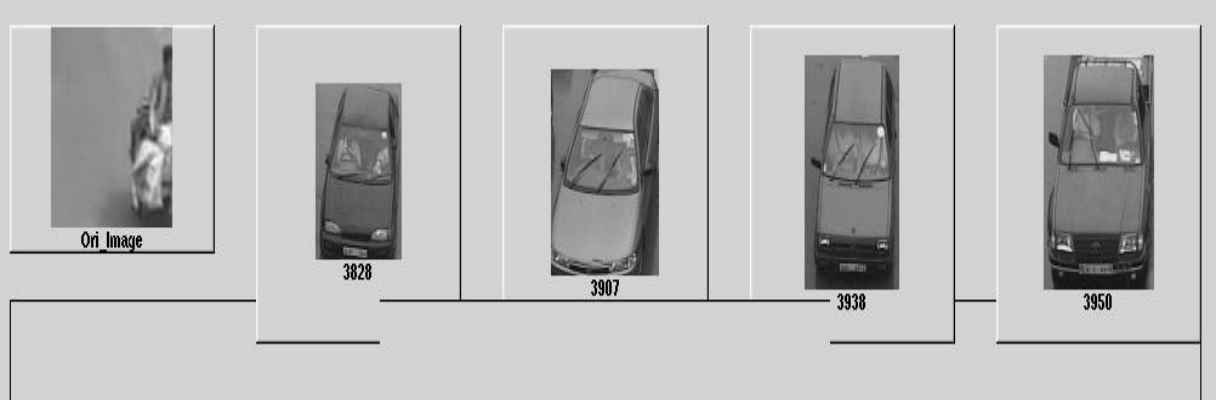


Figure 2

The image above shows that the negative images were having Eigen distance of less than 4000 units from the images in the training database.

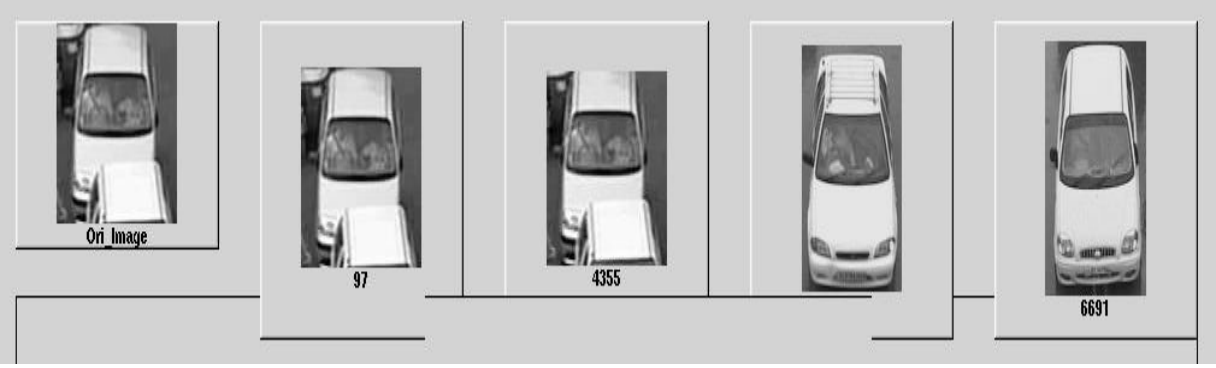


Figure 3

In the above image, the second nearest neighbor is at the distance of 4355 units though the only difference between the two being a small displacement. Hence if a system is to be built using the PCA technique, the parsing window must be moved with very low granularity over the image in order to accommodate all the large variances as seen above. This will increase the time complexity of the algorithm to quite an extent.

Limitations

- Scale variant: If an image containing various cars has to be tested to find the matching cars, then each patch of the image of size comparable to that of car has to be cropped and tested separately. It increases the computation time drastically.

- As already seen in one of the above images, for proper matching, the window needs to be very finely moved over the input images so as to capture every small variance in the patch. This drastically increases the time complexity of the method.

Future Prospects

- Different clusters of training images can be made according to their color, size, rate of occlusion and then matched with the input image. This needs to be a very large database.
- The time complexity of the method needs to be bettered.

Edge Based Detection

Another approach which we explored is edge based detection. Though not many object detectors are based on edge detection, but presence of strong edges along the windshield of the car and the generic shape of the car hint at the success of this method. As discussed before we analyze three sub methods

Fitting a minimal 2D model on the high level edge representation

1. The input gray scale image is passed through a canny edge detector

Illustration 1: Input Image to Canny edge detector

Illustration 2: Output of Canny Edge Detector

2. The output of canny edge detector is a binary image. To extract higher level information about edges, we perform a local search i.e. pixels lying approximately along a straight line are joined to get an edge. The edges are stored as pair of endpoints. Further, the set of edges is filtered to remove the edges which deviate from near horizontal and near vertical directions. This is justified because since we are detecting cars, the edges of interest will lie along either near horizontal or near vertical directions.

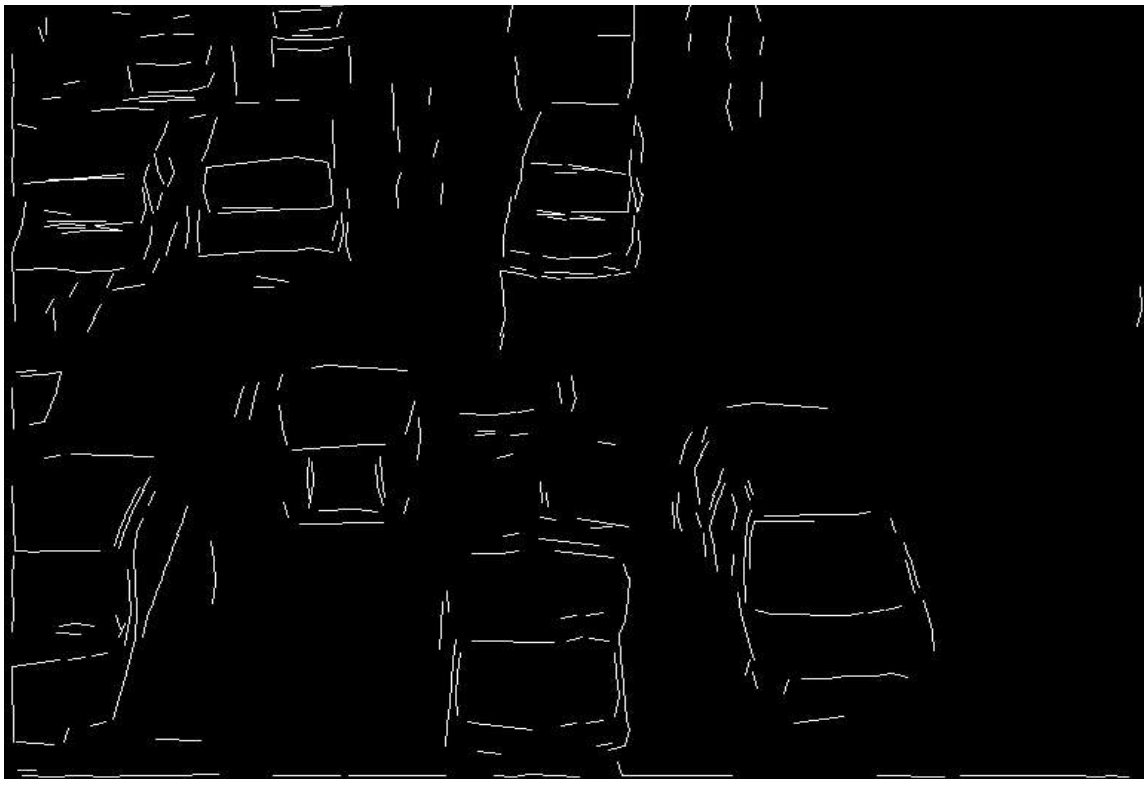


Illustration 3: Extracted edges. Note that some information is lost during the conversion of output of canny edge detector to this high level representation. Moreover, we get broken edges instead of

Though hough transform is a standard method for extraction of line segments it was observed that local search gave much better results. In fact hough transform is suited to images which have well defined and fewer no of line segments.The difference is observable in following outputs :

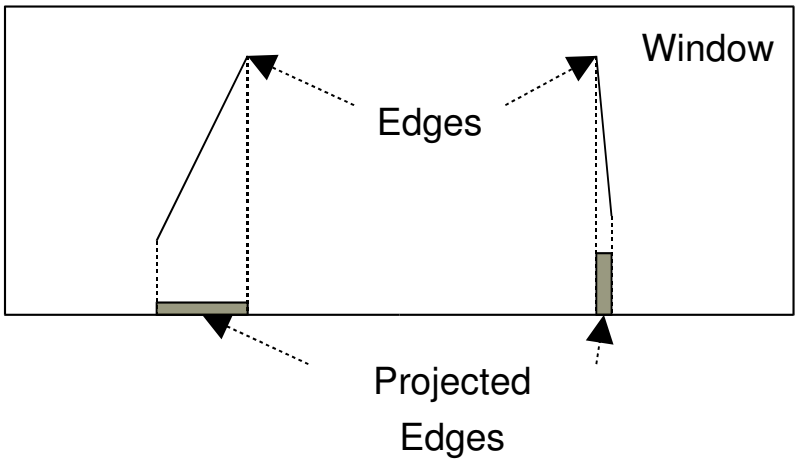
Edges extracted using Hough transform

Edges extracted using Local Search

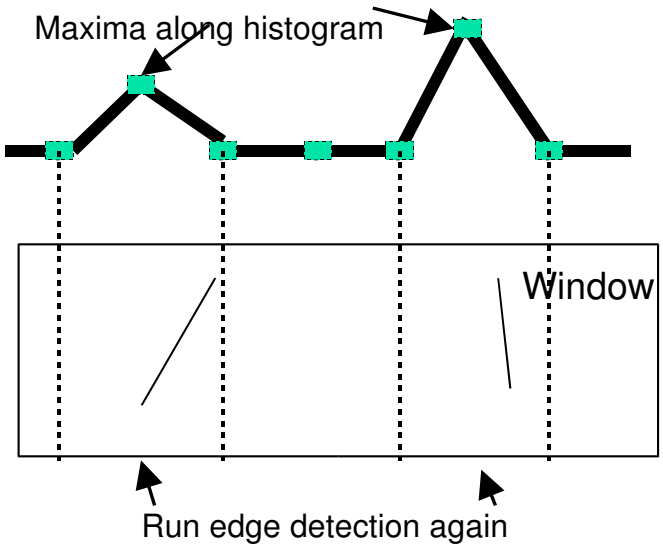
Further improving edge detection :

The edge detection can be improved by applying the following heuristic:

- Consider a window of appropriate size and consider the edges in it. Project the edges onto the X-axis as shown



- Quantize the projected points using a one dimensional histogram along the X-axis. The maxima of the histogram indicate the vertical regions in which the probability of a presence of a vertical edge is high. Hence edge detection may safely be done in that region again with relaxed thresholds as shown in the following figure. Similarly, by taking projection along Y-axis we can improve detection of horizontal edges as well.



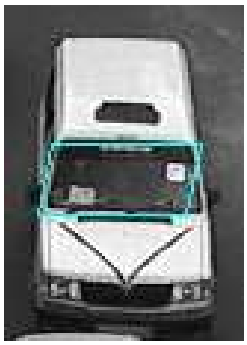
3. Next step consists of sliding a window over the whole image and in each window satisfying a threshold criterion of a minimum no of edges, a wire model of car is fit onto the edges present in that window. But as it is clear from the above illustration it is seldom possible to extract all the edges of the car. Hence, we try to fit a trapezium to the window.

Trapezium fitting

Fitting Strategy

- Out of a set of detected edges (in the current window), we select four edges (two horizontal and two vertical) and try to generate a trapezium from them assuming the horizontal edges to be parallel.
- A penalty is imposed for every other edge that does not agree with the generated trapezium. The penalty is proportional to the amount of deviation of the edge from the trapezium's structure.
- In every window the trapezium of lowest penalty is considered and if the penalty is lesser than a fixed thresh-hold value, then it is stored.
- The overlapping trapeziums are removed by selecting the trapezium with the minimum penalty if their corresponding windows overlap above a thresh-hold ratio. This can still be improved by considering the overlap of the trapeziums but this operation may be costly.

Due to complete set of edges extracted, the trapezium fitting is not consistent in the sense that there can be three sub cases of fitting as shown below :


-

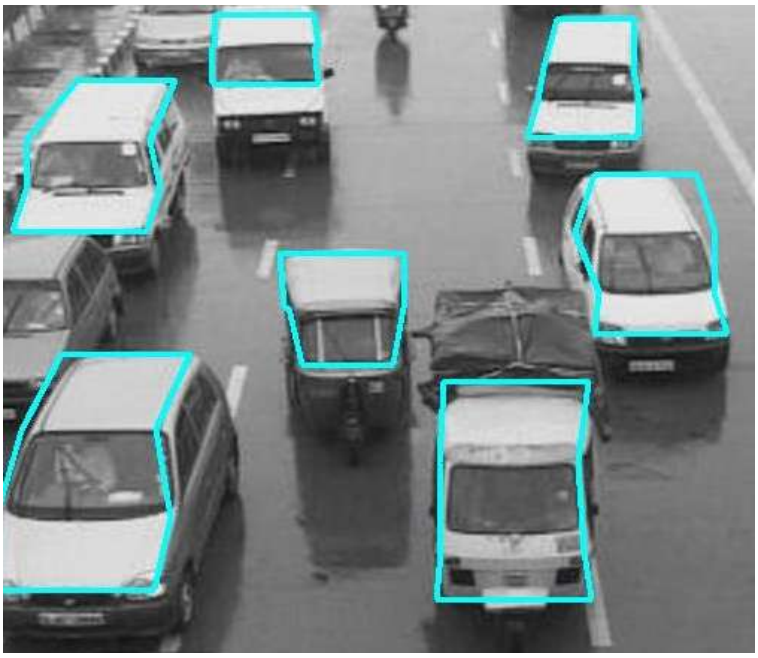
In order to localize a car, we need to distinguish between the three cases. The methodology for solving this problem is outlined below

- Consider the 4 windows as shown in the image around the detected trapezium.

Window 1
Window 2
Detected
Window 3
Window 4

- Again fit trapezium in the 4 windows but with relaxed bounds.
- Analyze the relative penalties of three sub cases according to the penlties of fit in the 4 windows.

Using this approach, it is possible to draw a tight bounding box.

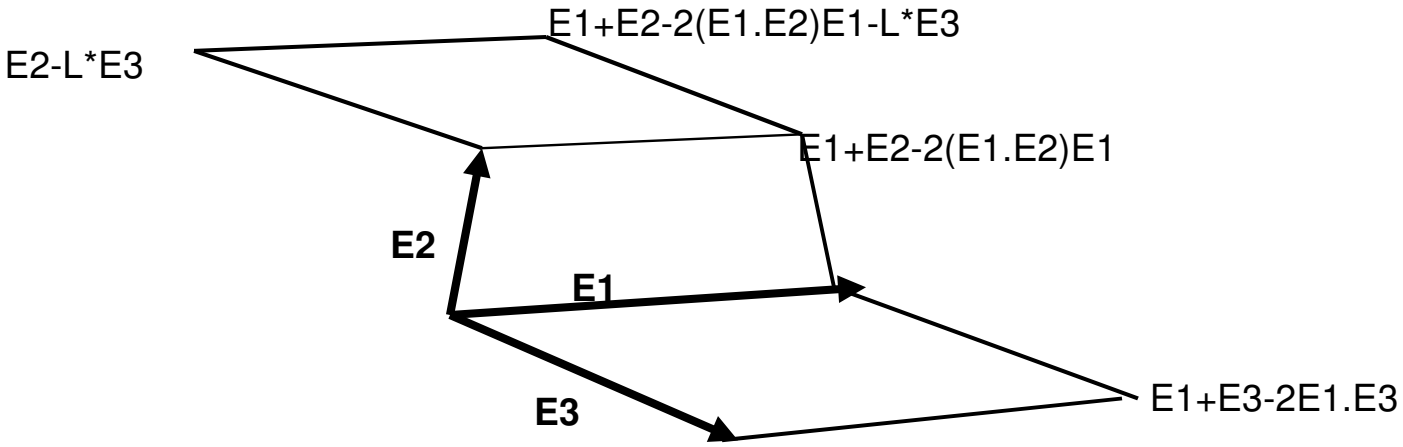


Result of detection using 2D fit

Using a 3D model of car and using RANSAC for calculation of orthographic projection matrix

The 2D fit method described above is inefficient in the sense that it utilizes a component based fitting for three major parts of a car. Moreover, the characterization of each part is similar. Hence we may end up merging parts of two different cars.

To increase robustness, a 3D fitting is proposed. Again, since the edge information is incomplete, a minimal 3D model is used as shown below



3D model used

The basis vectors E1, E2, E3 are highlighted in the figure. Two parameters are allowed to vary to account for the variability observed across cars namely
:

- **E1.E2: $0 <= E1.E2 < \theta_{max}$**
- **L: $L_{min} <= L <= L_{max}$**

The detection procedure is similar to as described in the previous approach.

- After detection and extraction of edges a window is run over the image. If a window satisfies the criteria of having a minimum no. of horizontal and vertical

edges, a RANSAC based approach is applied to select vectors as projections of the basis vectors. The procedure is $O(n^3)$ in number of edges present in the window.

- The projection matrix of the camera is approximated as an orthographic projection on this small region of the image. The approximation is justified since the size of the area concerned is relatively small as compared to the size of the image.
- From the knowledge of projection of the basis vectors, the projection matrix is calculated and the model is projected along this projection matrix.
- We now fit the projected model onto the edges present while taking into account the parameters *E1.E2* and *L*.

The time complexity of the RANSAC method can be improved by incorporating several heuristics while selecting the projected basis

- Partition the set of edges into horizontal and vertical edges beforehand. Use horizontal edges as projection of E1 and vertical edges as projection of E2 and E3.
- We can exploit the geometric arrangement of projected basis vectors i.e we'll like to select the projections e1, e2, e3 such that e1 lies in the middle of e2 and e3 since we are interested in frontal views of the model.

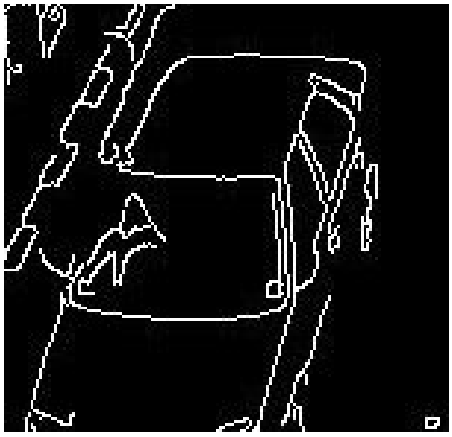
With above heuristics the detection can be done in near real time and it shows better results than 2D fitting.

3D fitting results on edge picture

Result of 3D fitting. Notice the accuracy of localization of cars

Fitting projections on binary edge picture(raw output of edge detector)

The main bottleneck of the 3D model fit is the inefficiency of the edge extraction. One can analyze the information loss by comparing the outputs of canny edge detector and extracted edges

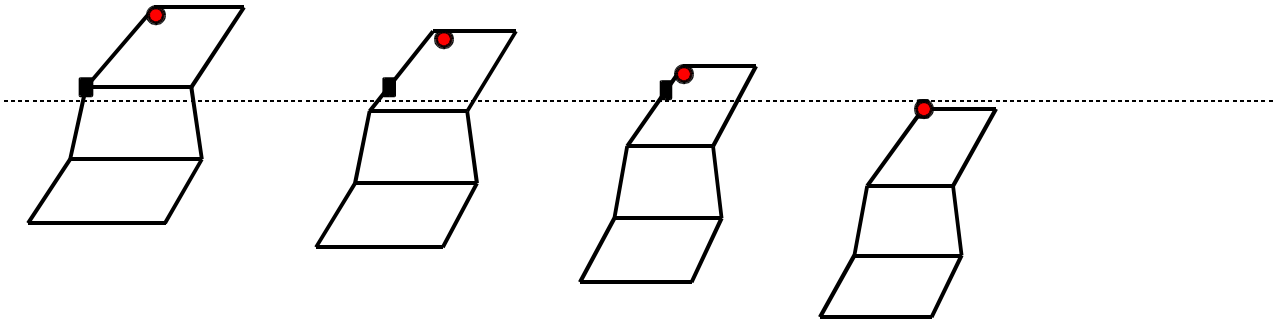


Output of canny edge detector

Output of edge extractor

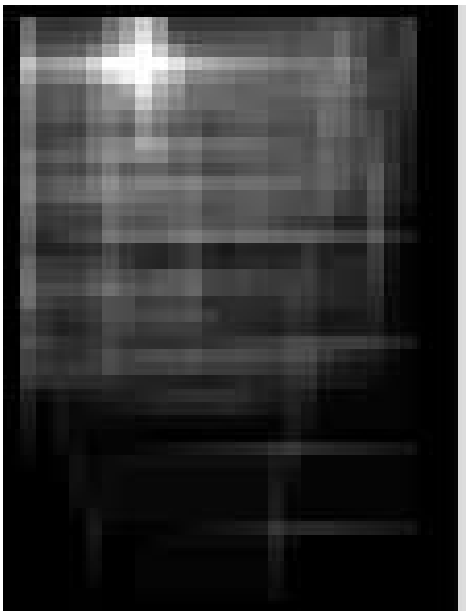
Hence we can try to directly work with binary output of the canny edge detector. Here too we use a similar 3D model but we can't incorporate the variation among different cars into a single model due to the voting method to be described later. We need to keep a dictionary of various 3D models. The step by step approach is as follows

- The initial step is same. Run window of size appropriately fixed by homography of the image over the image. If the window satisfies a threshold criterion (contains a minimum number of edge points) project the 3D model according to the projection matrix of the camera.
- For each edge point in the window cast votes as shown in the figure

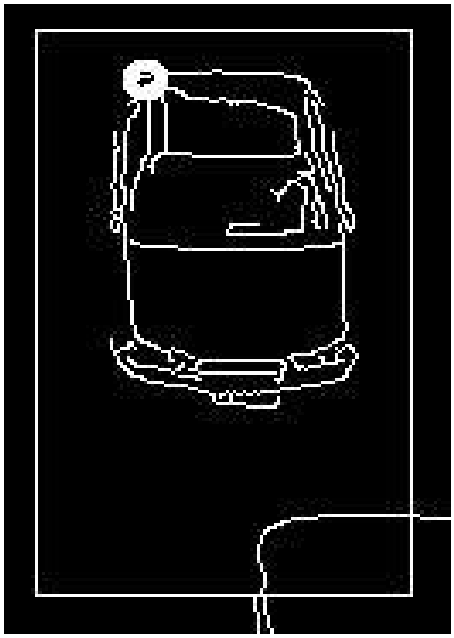


Voting Method

- Let the black square point denote an edge point. This point can lie along any edge of the projected model. Hence, for each edge move the point along the concerned edge in quantized steps. For each location of a point, it casts a vote for the subsequent location of top left corner of the model (shown by red dot). Repeat the process for every edge.
- After processing all the edge points in the window, we'll expect a histogram as shown if window contains a car.



Histogram seen as an intensity image



Corner of car as detected from maxima of histogram

- Hence we can threshold the histogram to sift out positive and negative results.

Result of detection using voting method. Again the cars are well localized

ANALYSIS

Edge based detection shows healthy results and detection rate was found to be approximately 90%. The pros and cons of this method may be weighted against other methods

PROS:

- It is considerably faster than SIFT based detection and it is possible to make a real time system based on this approach.
- The fitting algorithm is highly accurate and tight fitting bounding boxes for cars may be drawn.
- There is no need for extensive training data.
- The 3D fit method performs exceptionally well if high level extraction of edges is made accurate.
- The voting method can perform well if used with appropriately selected collection of 3D models.

CONS:

- There is a large amount of information loss while converting the output of canny edge detector into edges thus resulting in the failure of 3D fit and 2D fit method.
- The edges of black cars are not as pronounced and this makes the detection difficult..
- The voting method, which is not constrained by the accuracy of edge extraction is relatively slower.
- The system will not work as efficiently as SIFT in case of occlusion.

Combined Analysis

	SIFT	PCA	Edge Based Features Matching	HAAR
Guiding Principle	Potential keypoints detection, matching and clustering	Matching based on eigen distance between the training and the input images	Alignment and fitting of vehicle model on the set of edges detected in the image	Cascaded Haar Classifier built using AdaBoost
Detection Rate	95%	Not Applicable	90%	90%
Cases of Occlusion	works for upto 70% occlusion	difficult to include all kinds of occlusions in training database	Better than haar	Doesn't work in cases of occlusion
Detection of Diverse Cars	Dark cars detected by loose parameters which partially increase false matches	Results not feasible for complete implementation	Works for all kinds of cars. Depends on the edge detector used (edges of dark cars)	Efficiency dependent on the size of database
Pros	➢ Good detection in cases of occlusion ➢ Small size of training database required	Results not feasible for complete implementation	➢ Fast ➢ Tighter Bounding Box ➢ No training required	Detection speed is quite good
Cons	Finding keypoint descriptor is a long process	Results not feasible for complete implementation	➢ Detailed high level edge representation required. ➢ Black cars again a problem	Does not work in case of occlusion

References & Bibliography

1. Lowe,D.G.,2004 Distinctive Image Features from Scale-Invariant Keypoints

2. Viola,P. Jones,M. Robust Real Time Object Detection in Second International Workshop on Statistical and Computational Theories of Vision-Modelling, Learning, Computing and Sampling

3. Fergus,R. Perona,P. Zisserman,A. 2003 Object Class Recognition by Unsupervised Scale-Invariant Learning

4. Fergus,R. Perona,P.Zisserman,A. 2005 A sparse Object Category Model for Efficient Learning and Exhaustive Recognition

5. Brown,M. Lowe,D.G 2002 Invariant Features from Interest Point Groups

6. Lowe,D.G. 1999 Object Recognition from Local Scale-Invariant Features

7. Mikolajczyk,K. Schmid,C. A Performance Evaluation of Local Descriptors

8. Y. Ke and R. Sukthankar, 2004 Computer Vision and Pattern Recognition

9. Leung,B. 2004 Component Based Car Detection in Street Scene Images

10. Tijl De Bie, Cristianini,N. Rosipal,R. Eigenproblems in Pattern Recognition

11. Smith,L.I. 2002 A Tutorial on Principal Components Analysis