Car Detection in Low Resolution Aerial Images *

Tao Zhao Ram Nevatia
University of Southern California
Institute for Robotics and Intelligent Systems
Los Angeles CA 90089-0273
{taozhao|nevatia}@iris.usc.edu

Abstract

We present a system to detect passenger cars in aerial images along the road directions where cars appear as small objects. We pose this as a 3D object recognition problem to account for the variation in viewpoint and the shadow. We started from psychological tests to find important features for human detection of cars. Based on these observations, we selected the boundary of the car body, the boundary of the front windshield, and the shadow as the features. Some of these features are affected by the intensity of the car and whether or not there is a shadow along it. This information is represented in the structure of the Bayesian network that we use to integrate all features. Experiments show very promising results even on some very challenging images.

Keywords

Car Detection, Object Detection, Multi-Cue Integration, Bayesian Network, Aerial Image Analysis

1 Introduction

Vehicle detection in aerial images has important civilian and military uses, such as traffic surveillance, both for traffic information system or to gather traffic statistics for urban planning. It can also produce strong evidence for road detection [11]. It also provides a good test domain for methods of object detection in difficult situation that require integration of multiple cues. The aerial images we used are grayscale images taken mostly from a vertical

[1]This research was supported in part by a subgrant from MURI grand no. F49620-95-1-0457 from the US Army Research Office awarded to Purdue University.

or slightly oblique viewpoint. The length of a typical car in our datasets ranges from 13 to 26 pixels in image. The camera calibration is known as well as the sunlight direction.

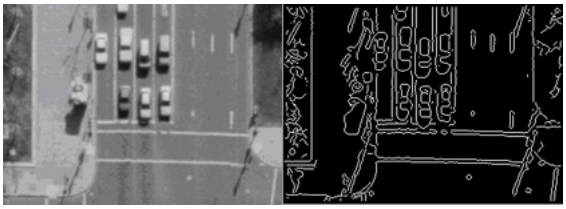
Detection from aerial image is easier than from detection from an arbitrary viewpoint in that the viewpoint is constrained. However, it is still not as easy as it may seem to be. Example images are shown in Fig.1. The main difficulties lie in the following:

- Although the viewpoint is constrained, there are still variations that make the cars have different appearance.
- The image resolution is low so not many details are visible.
- Some cars are heavily obscured by the environment in the images, mostly tree branches. (Fig.1.b)
- Cars can be of any intensity in the image, from very dark to very light. Also, some cars' intensity is very close to the road.
- The shadow cast on the ground by sunlight is more salient in an aerial view than in a ground view, which complicates the detection.
- The image quality varies. The brightness, contrast and sharpness of the images change due to factors including illumination, focusing and atmospheric turbulence.
- The expected features of a car differ with its intensity and the existence of shadow. For a simple example, whether or not the boundary of a gray car can be detected depends heavily on its shadow. (See 4.1 for more detail.)

We need to account for all these difficulties to get a reasonable good system.

1.1 Related work

The detection of vehicles has been receiving attention in the computer vision community because vehicles are such a significant part of our life. Papageorgiou and Poggio [9] presents a general method for object detection applied to car detection in front and rear view. An over-complete set of Haar wavelet coefficients of certain scales are computed and a SVM (Support Vector Machine) is trained to classify car and non-car. Rajagopalan, Brlina and Chellappa [12] models the distribution of car images by learning higher order statistics (HOS). The training samples are clustered according to HOS. Online background learning is performed and the HOS-based closeness of a testing image and each of the clusters of car distribution and background distribution is computed. Then it is classified into car or background. Schneiderman and Kanade [13] took a view-based approach and one



(a)

(b)

Figure 1: Two patches of images and their Canny results. (a) is clean but some edges are not all like our perception. (b) is heavily interfered with the tree branches and the cars are hardly visible in the edge map.

detector is built for each of the coarsely quantized viewpoints. It used the histograms of some empirically chosen wavelet features and their relative locations to model the car and non-car distribution assuming the histograms are statistically independent.

Vehicle detection in aerial images is relatively constrained by the viewpoint and the resolution. Generally, the camera calibration and sun angle are known for aerial images. In the work of Burlina, Parameswaran and Chellappa [1] and Moon, Chellappa and Rosenfeld [8], a vehicle is modeled as a rectangle of a range of sizes. Canny like edge detector is applied and GHT (Generalized Hough Transform) [1] or convolution with edge masks [8] are used to extract the four sides of the rectangular boundary. Liu, Gong and Haralick [6] uses average gray-level and average gradient level of the inside/outside/along the sides of the vehicle as features and learning the feature distribution for recognition. All of them treat vehicles as 2D objects and their primary evidence is the boundary of the car. This approach may be sufficient for their data (Fort Hood, a military site) where the vehicles are mostly of dark color, but may have problems when applied on urban scenes where the cars are of more variety.

There is also increasing interest in car detection from video streams, in which the motion cue can be utilized. In a static camera configuration, moving objects can be detected by background subtraction, while for a moving camera, an image stablizer is needed first. In Lipton, Fujiyoshi and Patil [5], the detected moving blobs are classified into human, vehicle and background clutter according to some simple shape measurements. In the situations of dense moving objects, a moving blob may not correspond to one single object, therefore, a more

detailed analysis similar to the technique in static image car detection should be employed. Although not for car detection, Koller, Daniilidis and Nagel [7] used a model-based approach to align the car model with the line segments of the image for tracking and pose estimation.

Most of the previous work either regards a car as a 2D pattern or takes a view-based approach where in each coarsely quantized viewpoint, a car is taken as a 2D pattern. This makes the performance degenerate when the viewpoint changes. Some efforts use statistical learning to resolve the variance of appearance, but the complex relationship of the different appearance would be difficult to learn.

1.2 Our approach

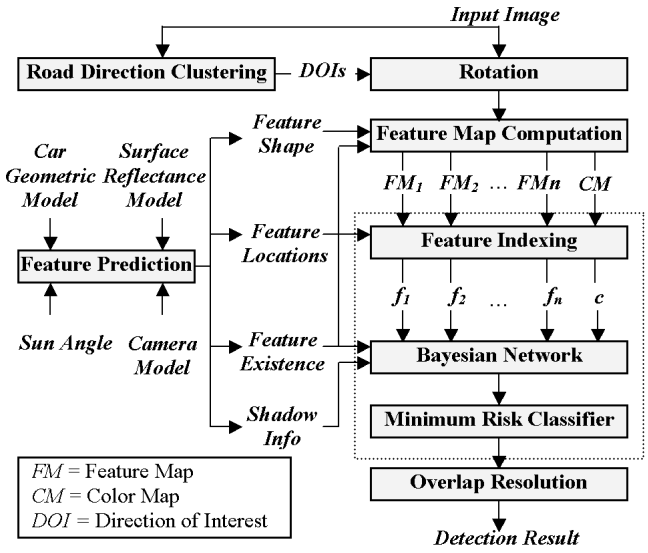


Figure 2: The diagram of the detection system. (Dotted rectangle means operations are carried out to every pixel of the image.)

We formulate the problem as a 3D object recognition problem to accommodate the change of viewpoint and make use of the camera calibration and known illumination to predict the shadow cues. The car geometry and the camera projection are modeled explicitly without learning while the distributions of the features are learnt for better adaptation and performance. We believe this is a good balance between using human knowledge and statistical learning. The diagram of our system if shown in Fig. 2 and a sketch of our approach is described below.

In this work, we only address detection of cars aligned with road direction. First the directions of the roads are estimated by clustering straight lines in an image considering the fact that most lines in urban images are aligned with the direction of roads. From a psychophysical test we performed, we decided to use four sides of the boundary of the car; four sides of the front windshield; two sides of outer boundary of the shadow and the intensity of the

shadow area (when exist) as features (Fig.5.b). With a generic model of a car, the expected features are predicted. The image features are computed at each pixel and verified with the expected ones. We observe that some of the feature distributions are affected by the intensity of the car and whether there is shadow along it or not. We embody this knowledge in the structure of the Bayesian network which is used to combine all features. The parameters of the Bayesian network are learnt from examples. Finally a decision of a car's existence is made using a Bayesian minimum risk classifier.

This paper is organized as follows. Section 2 outlines the psychological test we carried out mainly to discover how human recognizes cars in aerial images. Section 3 describes how the features are predicted and computed. Section 4 covers how the multiple features are combined with a Bayesians network. Section 5 presents the detection and post-processing. We show some results in Section 6, and finally reach the conclusion in Section 7.

2 A Psychophysical Test

The fact that human are very good at recognizing cars motivates us to do this informal psychological test to gain some insight on how humans achieve this capability. First we gathered some small image patches which contains one or more cars from large aerial images. We used a diverse dataset, including cars in different illumination condition, with different density, in different environment, etc. Some of the patches in the dataset is shown in Fig. 3. Then we asked a number of subjects to see these images and to retrospect about the factors that help them make the decision of the presence of a car. The factors most people mentioned are:

Figure 3: Some examples of the dataset for the psychophysical test.

- The rectangular shape and size of the car relative to other objects in the image.

- The layout of the visible windshields.
- The visible sides of the car when viewed obliquely.
- The shadow the car cast on the ground.
- The road, parking lot, and other environmental evidences.

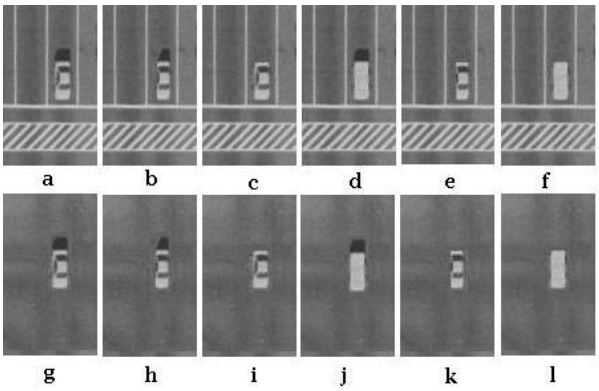


Figure 4: An example of the data used in psychological test. (a) original image of a car; (b) visible sides removed; (c) shadow removed; (d) windsheids removed; (e) both shadow and sides removed; (f) both shadow and windshields removed; (g)−(l) repeating (a)−(f) with car put in a neutral surrounding.

Then we chose some of the images in the dataset and modified some of the factors manually. The modification includes removing the windshield, removing the side of the car, removing the shadow cast on ground, putting the car in a neutral environment. In some cases, we also put the car with features modified into a street with other unmodified cars. An example is shown in Fig.4. We asked the subjects to classify what they saw and gave a confidence score for their decision. Besides, the promptness of reaction was also considered. The results from multiple subjects are summarized as the following:

- The rectangular or almost rectangular shape is the most important cue of a car.
- The layout of the windshields (frontal and rear ones) is an important factor for human detection.
- The car shadow, when it exists, can make the detection easier (reaction faster or with higher confidence) but it generally does not affect the decision.
- The environment also affects the detection. The presence of a parking lot, road or an assembly of cars is a strong supportive evidence that a rectangular object of appropriate size is a car although its other features are not salient.

3 Feature Extraction

3.1 Clustering of road directions

The vertical view aerial images of an urban area generally exhibit a few major directions. These directions are made by the parallel roads and the buildings and other structures aligned with them. These directions of interest (DOI) can be estimated from the images since a large number of straight lines in the image are aligned with these directions. We obtain the DOI of a local part of a city (e.g., 8 blocks square) by computing the histogram of the directions of the straight lines weighted by their length. The straight lines are obtained by an LMS fitting to the results of Canny edge detector. The histograms have sharp peaks at the major directions. Peaks above some threshold are declared as DOIs of this local patch of image.

When the viewpoint of an image is oblique, the image is re-projected onto a plane parallel to the imaging plane to remove the perspective effect on parallel lines before the DOI estimation. The image patch is rotated to make the DOI vertical in image. If one image has more than one DOI, we rotate it to form multiple images and then combine the results later. In fact, we rotate the image into twice the number of the DOIs to handle two-way traffic for we assume that the cars always headed down.

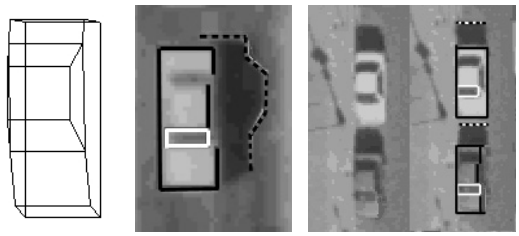
3.2 Features used for detection

Motivated by the psychophysical test, we decided to use the following features (Fig.5.b).

- The boundary of the car. The boundary of the car is mostly rectangular, but the two long sides may turn into curves in some viewpoints (see the difference of the two cars in Fig.5.c).
- The boundary of front windshield. We use only the front windshield because its shape, size and location in the car are relatively fixed. It is always assumed to be rectangular.
- The outer boundary of the shadow area when shadow exists. The shadow is an important evidence to differentiate cars from other planar rectangular structures. The information of the inner shadow boundary is utilized in the car body boundary feature. In the case of very oblique sun angle, this cue will not be used since the shadow boundaries are far from the car and less reliable.
- (optional) The intensity of the shadow area when shadow exists. It is optional because it is expensive to compute when the area is large and its contribution to detection is not as significant as other features.

As can be seen, the features we use are mostly gradient features with linear or almost linear shape, except the intensity of the shadow area. To extract this information, two methods can be used - symbolic edges (e.g. extracted by Canny edge detector) [1] or responses of gradient filters of certain shapes [8]. We chose the latter due to the following reasons. Firstly, due to the small size of a car and possible occlusions in the image, edges of the car may be lost or become fragmented and difficult to be extracted robustly (see Fig.1). Secondly, some features (two sides of the car boundary) are not always linear. Thirdly, the response of gradient filter has better resolution in amplitude than the binary valued edge detector.

Therefore the features are represented as thin gradient (horizontal or vertical) masks. These masks are convolved with the images to get the value of corresponding features. The convolution masks are generated by the feature prediction module described below.



(a) (b) (c)

Figure 5: (a) Wire-frame car model. (b) All features used for detection, including boundary of the car body (in dark line), boundary of the front windshield (in white line), outer boundary of shadow area (in dotted line) and the intensity of shadow area (dark area). (c) Predicted features overlaid on the image. (Note the shape of the right sides of the two cars is different.)

3.3 Model-based feature prediction

To detect cars with the above features, we need to know the shape (for left/right sides of car body boundary only), the location relative to the center, and the expected strength of the different features. Therefore, we use a coarse generic car model to predict the above information to verify in the input image. The generic model includes a wire-frame geometrical model and a surface reflectance model.

We use a generic wire-frame car model as shown in Fig.5.a to predict the 2D location of edges of the car. This is done by projection to the ground plane. The cars are far away from the camera and the relative depth of the cars in the view direction is small, so we can use a scaled orthographic camera model. The wire-frame model is parameterized to represent four types of cars: *mini*, *compact*, *full-size*, and *luxury*. These four car types should be able to give a reasonable approximation of the geometry of most passenger cars. For computational efficiency, the

shapes of the features of different car types are assumed the same and only the relative locations are different.

Only a subset of the edges in the wire-frame model is used as features. Front windshield and shadow are approximately planar so there is no ambiguity about their projected boundary. But for the boundary of the car body itself, for each of the four sides, there are two edges (upper and lower) that may appear in the image. We assume that only one significant edge per side will appear in the image. For the case where the lower edge is occluded, the upper one is used. But for the case where both edges are visible, we choose the edge along which the two intensities have greater difference. For example, if the intensity difference of the hood and the front of the car is less than the intensity difference of the front and the ground (or shadow if there is shadow along it), the lower front edge will be chosen. For this, we need a reflectance model of the car and the road to determine the intensity of the planes of the car body in the image under the known illumination. We use the following modified Lambertian model.

$$
\begin{aligned}
&I = I_{amb} + I_{spec} \\
&I_{amb} = \begin{cases} k_a(a_{others} + a_{ground}) & \text{if } \hat{D} \cdot \hat{N} < 0; \\ k_a(a_{others} + a_{ground}\hat{D} \cdot \hat{N}) & \text{if } \hat{D} \cdot \hat{N} >= 0. \end{cases} \\
&I_{spec} = -k_d d\hat{D} \cdot \hat{N}
\end{aligned}
$$

where $\hat{N}$ is the surface unit normal, $\hat{D}$ is the sun direction unit normal, d is sun light intensity, a_{ground},a_{others} are ambient light intensity from the ground and other places respectively, k_d is reflectance to directional light, k_a is reflectance to ambient light.

The empirical modification is intended to take into account the light reflected by the road onto the sides of the car. The parameters k_a, k_d are obtained directly by measuring the images for each car intensity, and d, a_{ground} and a_{others} are estimated from examples. Since we don't need very accurate values, the simple model works quite well. We do not model highlights on the car because they are sensitive to the subtle curvature of the surface.

4 Multi-feature Integration

4.1 Parameterization of the features

We observed that the values of some of the features are influenced by the intensity of the car and the shadow along them, which we denote as i and s respectively, as can be seen in Fig.6. More specifically, the values of four boundary edges of the car body are affected by both i and s; the two short vertical edges of the front windshield are affected by both i and s since they are close to the boundary; the two horizontal edges of the front windshield are

affected only by i; and the shadow features are not affected by either s or i. The factors are quantized into discreet values for simplicity. s is quantized into 3 values: *no shadow*, *thin shadow*, and *wide shadow*; i is quantized into 3 values: *dark-colored*, *gray-colored* and *light-colored*.

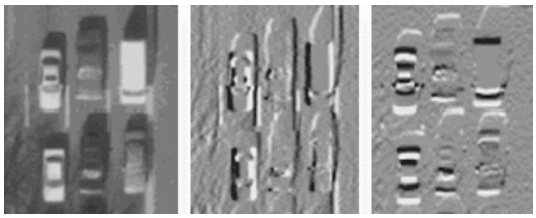


Figure 6: An image patch and its X/Y gradient maps. (in gradient maps, mid gray - 0, white - positive, black - negative) Expected feature values vary with the intensity of the car and shadow. Comparing the bottom three cars, all the features of the white car are clear; the top and right boundaries of the black car are invisible; the left boundary of the gray car is faint; the windshields of the black car and white car exhibit different sign in gradient maps.

4.2 Integration through a Bayesian network

Considering all the available evidence, we may have as many as 11 features to use. We need to combine these features to get a final decision (probability) of a car's existence. Bayesian networks (BN) provide an optimal way to integrate multiple cues for a decision if the conditional distributions are known [10]. They have been used in various applications in computer vision research and shown promising performance [3]. We use a BN to integrate all available evidences with the structure shown in Fig.7. The factors i and s are made parent nodes of the affected evidence nodes according to the parameterization above. The posterior probability $P(car|F, i, s)$ of the node *"car"* (where $F = [f_1, f_2, f_n]$ is all available features) is the one to be evaluated. The values of the evidence nodes are measured from the image at the locations computed by the feature prediction module. The parameter node s is computed from the model, view angle and sun angle. The parameter node i is obtained by getting the median of the intensities of a small region around the pixel as a robust estimation of the car's intensity. Each of the evidence nodes has a conditional probability table (CPT) associated with it that is indexed by the value combination of its parents and itself. For example, the CPT of the node BU is in the form of $P(f_{BU}|car, i, s)$.

It should be noted that the BN assumes conditional independency, i.e., the values of the feature nodes are assumed independent when all their parents nodes are fixed. We introduced two parameter nodes to achieve the conditional independence.

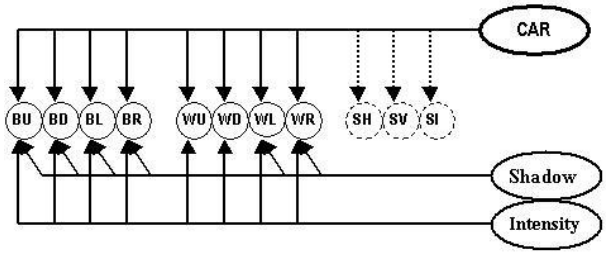


Figure 7: The Bayesian network used for detection. Dotted line shows not always available. B = body boundary / W = front windshield / S = shadow / U = rear / D = front / L = left / R = right / I = intensity. Evidences in dotted circles are used only when available.

4.3 Handcrafted BN parameters

We need to set the CPT values for the network. First, we fill these values manually based on our observation. An example is shown in Tab.1. The value of the features is quantized to binary (exists or not) using thresholds because it is hard for human to deal with continuous value directly. The parameters are designed considering the following empirical guidelines as well as measuring some examples in images:

- For *light-* and *gray-colored* cars, the probability to detect an edge along a *shadow* is higher than *no shadow* (s is quantized into *shadow* and *no shadow* in handcrafted BN);
- For *dark-colored* cars, the probability to detect an edge along a $shadow$ is much lower than $no\ shadow$;
- For *dark-colored* cars, the probability to detect the windshield boundary edges is low;
- *Light-colored* cars has higher probability to have a body boundary / windshield boundary edge detected than *gray-colored* cars;
- Short edges are more likely to be made noisy than long edges;
- The left/right edges are more likely to be made noisy than front/rear edges.

Performing detection with the handcrafted parameters showed reasonable performance. However, they are limited in the aspects discussed below. We estimate the parameters by learning from examples, which lead to better performance in testing.

4.4 Learning the parameters

Handcrafting the parameters is limited in the following aspects.

car	i	s	$P(f_{BU}\|car, i, s)$	$P(\overline{f_{BU}}\|car, i, s)$
$\overline{car}$	-	-	0.05	0.95
car	dark	no	0.8	0.2
car	dark	thin	0.4	0.6
car	dark	wide	0.15	0.85
car	gray	no	0.6	0.4
car	gray	thin	0.7	0.3
car	gray	wide	0.8	0.2
car	light	no	0.8	0.2
car	light	thin	0.9	0.1
car	light	wide	0.95	0.05

Table 1: A hand crafted CPT for front boundary of car body.

- Humans' qualitative experience may not cover every part of the problem domain.
- It is not easy to transform humans' subjective qualitative experience into objective numerical quantities for computation.
- Since humans are not good at dealing with complex numerical relationships, such as distributions, the quantization from continuous distribution to binary value loses information that could have been utilized.
- Studying a large number of examples is a tedious job requiring the familiarity with vision algorithms.

Therefore, learning the parameters of the BN from examples by the computer will be ideal. In this way, we can model the sensory data with much finer resolution (we are using 64 quantization levels). A common way to represent a distribution is to fit it with some known parametric distribution forms, among which Gaussian density is the most commonly used. But here, some of the distributions display multiple modes, thus non-Gaussian. It can be modeled by a mixture of Gaussians, but for simplicity, we use non-parametric technique instead. We tried KNN (k-nearest neighbor) and Parzen Window with Gaussian kernel. The results showed the latter outperformed the former by a small amount.

Since there are no hidden nodes in the network, learning the CPT requires just calculating the histogram. By the conditional independence assumption, the CPT of each evidence node can be learnt independently. We collected around 200 cars manually from 25 patches of images for training as positive examples and other parts of the images are served as negative examples. Only the car boundary is specified on the images and the locations

of other evidences are computed by the feature prediction module. Fig.6 shows some of the learnt CPTs; the distribution contains the guidelines of human experience as well as other aspects not noticed by human observer.

We use it as human knowledge at first that the value of the features are affected by the factors i and s. With the learnt distributions, we proved that the parameterization is efficient by showing Kullback-Leibler divergence of $P(f_j|car)$ and $P(f_i|factor, car)$ is large, where $factor = i, s$.

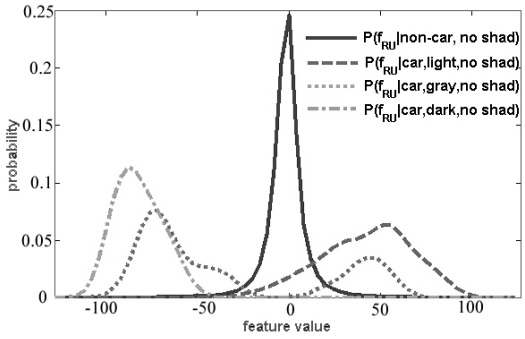


Figure 8: Learnt CPT of the front boundary of car body when there is no shadow along it ($P(f_{RU}|car, i, s = noshadow)$). For $light - colored$ car, the distribution peaks on position side; for non-car, the distribution highly peaks at zero; for $dark - colored$ car, the distribution peaks on the negative side; and for the $gray - colored$ cars, the distribution peaks in two modes, both positive and negative.

5 Detection and Post-processing

5.1 Detection

After the BN is constructed and learnt, we can use it to compute the probability of the presence of car at a given point. First the feature maps $FM_1, FM_2, ...FM_n$ are computed. Suppose we want to evaluate $P(car|F, i, s)$ at image location (x, y). The offset of a feature j ($j = 1, ..., N$) in location from (x, y) is computed by the feature prediction module as (dx_j, dy_j). And the features are retrieved from the corresponding feature maps so that $f_j = FM_j(x + dx_j, y + dy_j)$.

$$P(car|F, i, s) = \alpha(\Pi_{j=1}^{n} P(f_j|car, i, s))P(car)$$

$$P(\overline{car}|F, i, s) = 1 - P(car|F, i, s)$$

$$\alpha = \frac{1}{(\Pi_{j=1}^{n} P(f_j|car,i,s))P(car)+\Pi_{j=1}^{n} P(f_j|\overline{car},i,s))P(\overline{car}))}$$

For each feature f_j, we perform a local search in the neighborhood (+/-1 pixel along the direction orthogonal to the feature in our implementation) of the expected location to find the maximum value of $P(f_j|car, i, s)$ in order to account for small variation of the feature locations of car model. The probability of car existence at (x, y) is taken to be the maximum of the probability of all sizes (*mini*, *compact*, *full-size*, and *luxury*).

A minimum risk classifier instead of a minimum error rate classifier is used to make the final decision. It is more suitable here because a user can conveniently specify their preference on false alarms or mis-detections by a risk matrix C:

$$C = \begin{pmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{pmatrix}$$

where 0 means $\overline{car}$ and 1 means car. c_{01} means the cost to misclassify a car to be a $\overline{car}$, and so on.

The decision rule for classifying a car in minimum error rate classifier is:

$$P(car|F, i, s) > P(\overline{car}|F, i, s)$$

$$\Longleftrightarrow$$

$$P(F|i, s, car)P(car) > P(F|i, s, \overline{car})P(\overline{car}) \tag{1}$$

In a Bayesian minimum risk classifier, the decision rule is replaced by:

$$R(car|F, i, s) < R(\overline{car}|F, i, s)$$

where R is the expected risk. *i* and *s* are omitted in the derivation below.

$$R(car|F) < R(\overline{car}|F)$$

$$\Longleftrightarrow$$

$$P(F|car)P(car)(c_{01} - c_{11}) > P(F|\overline{car})P(\overline{car})(c_{10} - c_{00})$$

$$\Longleftrightarrow$$

$$P(F|car)P'(car) > P(F|\overline{car})P'(\overline{car}) \tag{2}$$

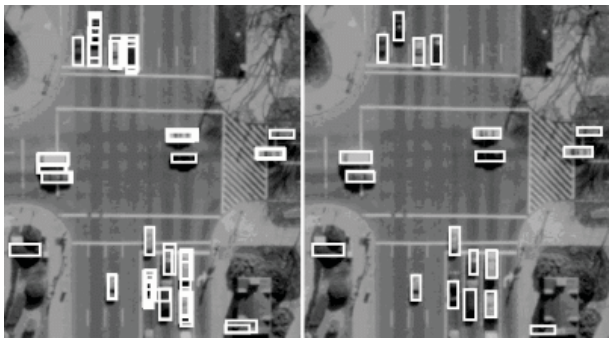
where $P'(car) = \frac{P(car)(c_{01}-c11)}{P(car)(c_{01}-c11)+P(\overline{car})(c_{10}-c_{00})}$, and $P'(\overline{car}) = 1 - P'(car)$.

Compared to rule(1), rule(2) is equivalent to adjusting the prior probability $P(car)$ and $P(\overline{car})$. We will mention $P(F|car)P'(car)$ of all pixels as the car probability map in the rest of the paper since it is the posterior probability after adjusting the prior. In our case, c_{00} and c_{11} are set to 0 and c_{01} is set to 1. c_{10} is the only free parameter for the user to specify.

5.2 Post-processing

Generally a number of pixels around the center of a car will all have a high probability value, thus many of these will also be classified as center of cars. Besides, the coincidental alignment of boundary lines and shadow lines as well as the coincidental alignment of features of adjacent cars may also create other high probability spots which are close to the true center of the car. (See Fig.9.a for example.) Overlap resolution is needed to remove the redundant results.

First, we find the connected regions in the probability map. Then for each connected component, we compute the sum of all the probabilities as a score. We assume that the true detections have higher score than false alarms around it, which is true most of the time in our experiments. The connected components are sorted by their summed probability. Valid detections are chosen from the front of the queue. For each connected region, if it does not overlap with any of the previously chosen valid detections, it is identified as a valid detection, otherwise discarded. The probability weighed centroid is used as the position of the detected car.



(a) (b)

Figure 9: Before(a) and after(b) overlap resolution.

As mentioned in [8], false alarms may appear when the sunlight creates shadow with the similar width as a car. It may also happen in our system especially when the car features are not perfect to suppress it in the previous step. If we detect a dark car very close to the expected shadow location of the car next to it and the dark car does not have good shadow evidence, we regard the dark car as shadow even if it has higher summed probability than the car next to it. One result example is shown in Fig.9.b.

6 Results and Discussion

6.1 Results

We report the results of the above described approach on two data sets: Washington DC dataset and vertical view Fort Hood dataset.

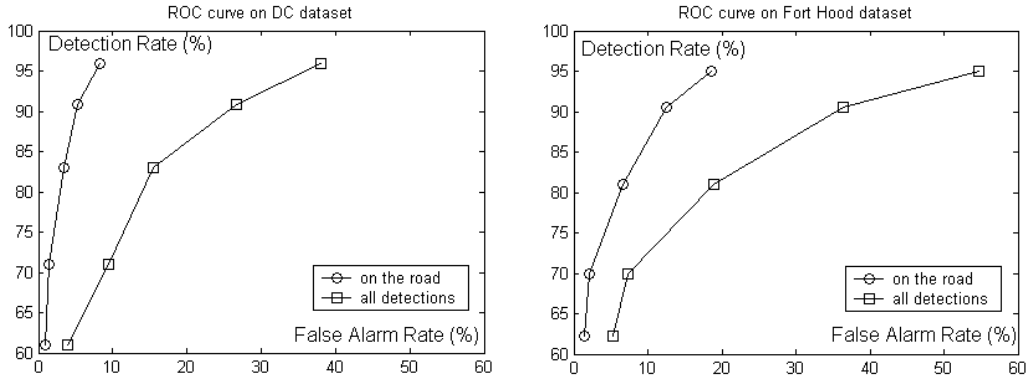
In the Washington DC dataset, a typical car has a length of around 26 pixels. We tested our system on 12 images patches of road and adjacent area which contain 320 cars in total. The image patches we selected have different sun angles, view angles, image quality and against different backgrounds and do not have overlap with the training data (Fig.11). Some of the images present very difficult conditions due to cars being hidden under tree branches (Fig.11.c,h), shadow having similar width as a car (Fig.11.a) and oblique viewpoints of some images (Fig.11.i). With all these difficulties, our detector still shows good results.

The Fort Hood dataset has half of the resolution of the Washington DC dataset, and a typical car only has a length of around 13 pixels. Furthermore, the image contrast is lower than the Washington DC dataset. We selected 12 image patches with cars on the road which contain 356 cars in total. The system used the parameters learnt on the Washington DC training data. Some of the results are shown in Fig.12. We also show the result on a parking lot in Fig.12.f.

We find that most cars with good features are detected, while some of the difficult ones also got detected with appropriate choice of the c_{10} value. Although aimed at only detecting regular passenger cars, it also detects some other vehicles (vans, SUVs) sharing similar feature placement. Both the mis-detection rate and the false alarm rate of *dark-colored* cars are higher than cars of other colors because under most situations they don't have as salient features as *light-* or *gray-colored* ones. Most false alarms result from the coincidental alignment of rectangular shape and other lines of structures in buildings, foliage of trees or road markings.

For a detector, there is always a tradeoff between false alarm rate and mis-detection rate. ROC (receiver operating characteristic) curves characterizing the performance of our system on the two datasets are given in Fig.10. The detection rate is defined as (number of correct detections)/(total number of cars) and the false alarm rate is defined as (number of false alarms)/(total number of cars). In some applications such as vehicle counting, the coarse knowledge of the road is known (e.g., by registering images with the TIGER/Line [14] database which contains the road information of US cities.). Therefore besides the regular ROC curves, ROC curves not considering the off-road part are also given. The 5 points on the curves of the two datasets are obtained with the same parameter setting. By comparing the ROC curves of the two datasets, we can find that the Fort Hood dataset has more false alarms while maintaining a slightly lower detection rate. We regard it as a consequence of the difference in image

resolution and contrast. We feel that the receiver operating characteristic of the system is sufficient for applications such as road verification, estimation of approximate traffic flow, etc.



(a) (b)

Figure 10: The ROC curves of the detector on (a) Washington DC dataset, (b) Vertical view Fort Hood dataset.

6.2 Computation time

The computation time is proportional to the number of pixels processed. Convolution of feature masks takes the greatest share of time. We use convolution decomposition to accelerate the computation by separating the gradient and the shape of the feature. Since the convolution is with binary mask, all the multiplication operations are replaced by integer logic operations. After all these techniques, a $1000 * 870$ image takes about 4 seconds for one DOI on a Pentium 4 $2.7GHz$ PC. All the operations before the overlap resolution are carried out at each pixel of the image independently, so they can be easily speeded up by highly-parallel processing.

7 Conclusion and Future Work

We have described a car detection system for aerial images that functions well with both vertical and slightly oblique viewpoints since we formulate the detection as a 3D object recognition problem. We analyze shadows explicitly to make them a useful cue for detection instead of a source of problems as has been the case in some previous work [8]. We use the response of gradient mask filters as feature to account for the low resolution and noise in aerial image, which is more robust than binary edge detection. We introduced car intensity i and shadow s as parameters and they were proven to be effective. A Bayesian network is used to combine multiple features with learning and the only hand given parameter is to make a balance over false alarms and mis-detections. Our method gives very promising result on tested examples, even in very difficult situations.

Machine learning is gaining popularity in computer vision community. In this work, we showed a good example on how human knowledge and learning can be balanced. The generic car model, camera projection and the parameterization of the features that are difficult to learn from the data are introduced as prior knowledge and the parameters of the Bayesian network are learnt from examples. We believe that similar approaches can also be useful for other object detection and recognition tasks.

This work can be improved and extended in the following aspects:

- Collect training data under different situations: this should result in better performance.
- Experiment with cars viewed from more oblique viewpoints to see if it can be extended to more general situations.
- Context information, as indicated in the psychophysical test, could be utilized for better performance.

References

[1] P. Burlina, V. Parameswaran and R. Chellappa, Sensitivity Analysis and Learning Strategies for Context-Based Vehicle Detection Algorithms, *Proc. DARPA Image Understanding Workshop*, pp. 577-584, 1997.

[2] P. Domingos and M. Pazzani, Beyond independence: Conditions for the optimality of the simple Bayesian classifier, *Proc. International Conference on Machine Learning*, pp. 105-112, 1996.

[3] Z. Kim and R. Nevatia, Uncertain Reasoning and Learning for Feature Grouping, *Computer Vision and Image Understanding*, Vol. 76, No. 3, pp. 278-288, December, 1999.

[4] C. Lin and R. Nevatia, Building Detection and Description from a Single Intensity Image, *Computer Vision and Image Understanding*, Vol. 72, No. 2, pp. 101-121, November 1998.

[5] A.J. Lipton, H. Fujiyoshi and R.R. Patil, Moving Target Classification and Tracking from Real-time Video, *Proc. DARPA Image Understanding Workshop*, pp. 8-14, 1998.

[6] G. Liu, L. Gong, and R.M. Haralick, Vehicle Detection in Aerial Imagery and Performance Evaluation, *Submitted for publication*.

[7] D. Koller, K. Daniilidis, and H. H. Nagel, Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes, *International Journal of Computer Vision*, Vol. 10, No. 3 pp. 257-281, 1993.

[8] H. Moon, R. Chellappa and A. Rosenfeld, Performance Analysis of a Simple Vehicle Detection Algorithm, *Image and Vision Computing*, Vol.20, No. 1, pp. 1-13, 2002.

[9] C. Papageorgiou, and T. Poggio, A Trainable System for Object Detection, *International Journal of Computer Vision*, Vol. 38, No. 1, pp. 15-33, 2000.

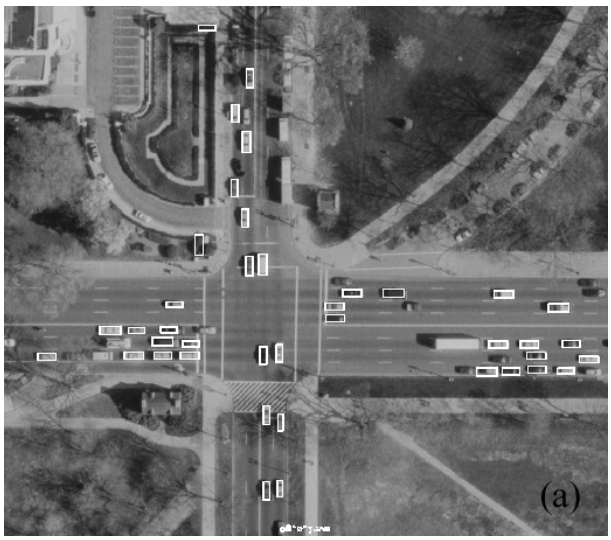
[10] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, San Francisco: Morgan Kaufmann, 1988.

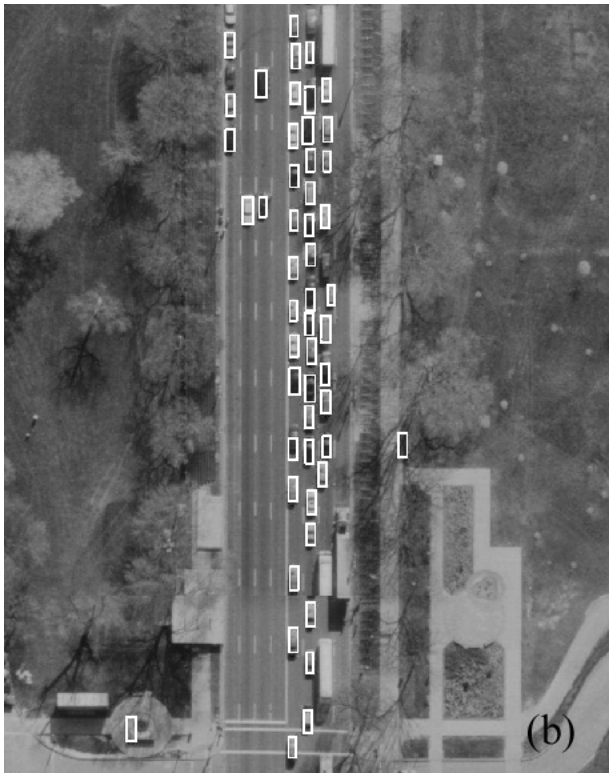
[11] K. Price, Urban Street Grid Description and Verification, *Proc. IEEE Workshop on the Application of Computer Vision*, pp. 148-154, 2000.

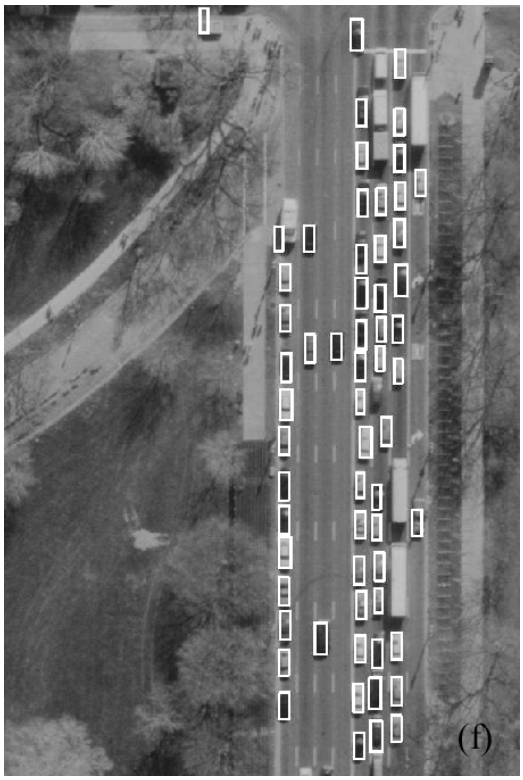
[12] A. Rajagopalan, P. Burlina and R. Chellappa, Higher Order Statistical Learning for Vehicle Detection in Images, *Proc. IEEE International Conference on Computer Vision*, pp. 1204-1209, 1999.

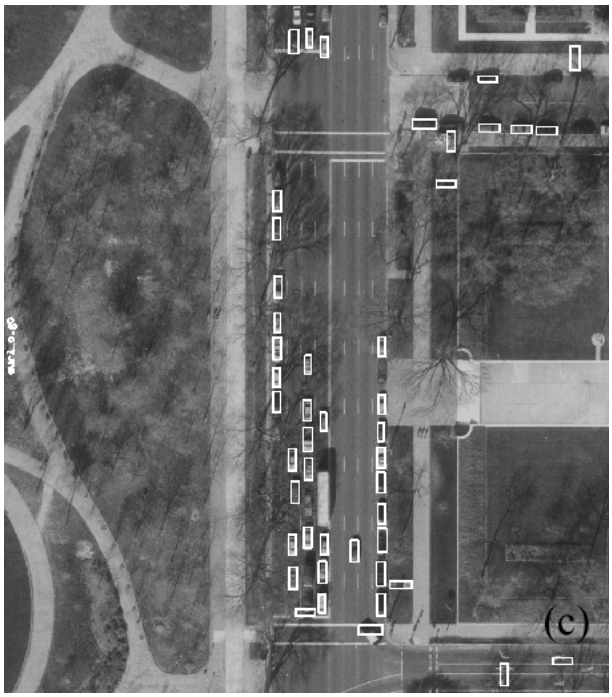
[13] H. Schneiderman and T. Kanade, A Statistical Method for 3D Object Detection Applied to Faces and Cars, *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 746-751, 2000.

[14] U.S.Census Bureau - TIGER/Lines, http://www.census.gov/geo/www/tiger/, 2000.




(b)


(f)


(c)


(g)


(d)

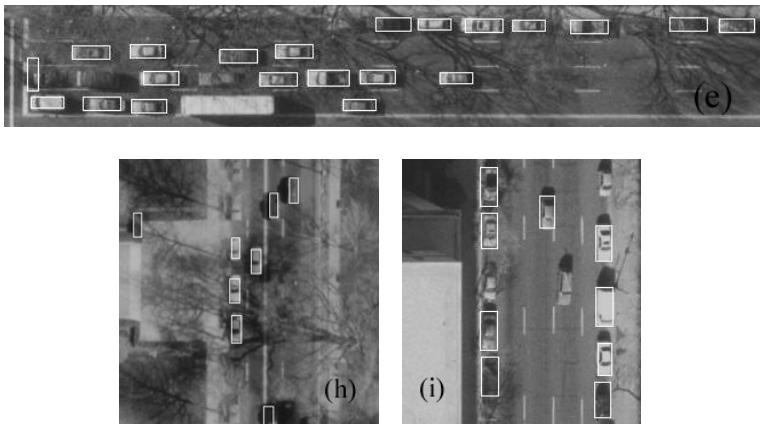


Figure 11: Some of the detection results on Washington DC dataset. (d)(e) are the close-up (rotated to fit in page) of (c); (h) is the results of Fig.1.b.

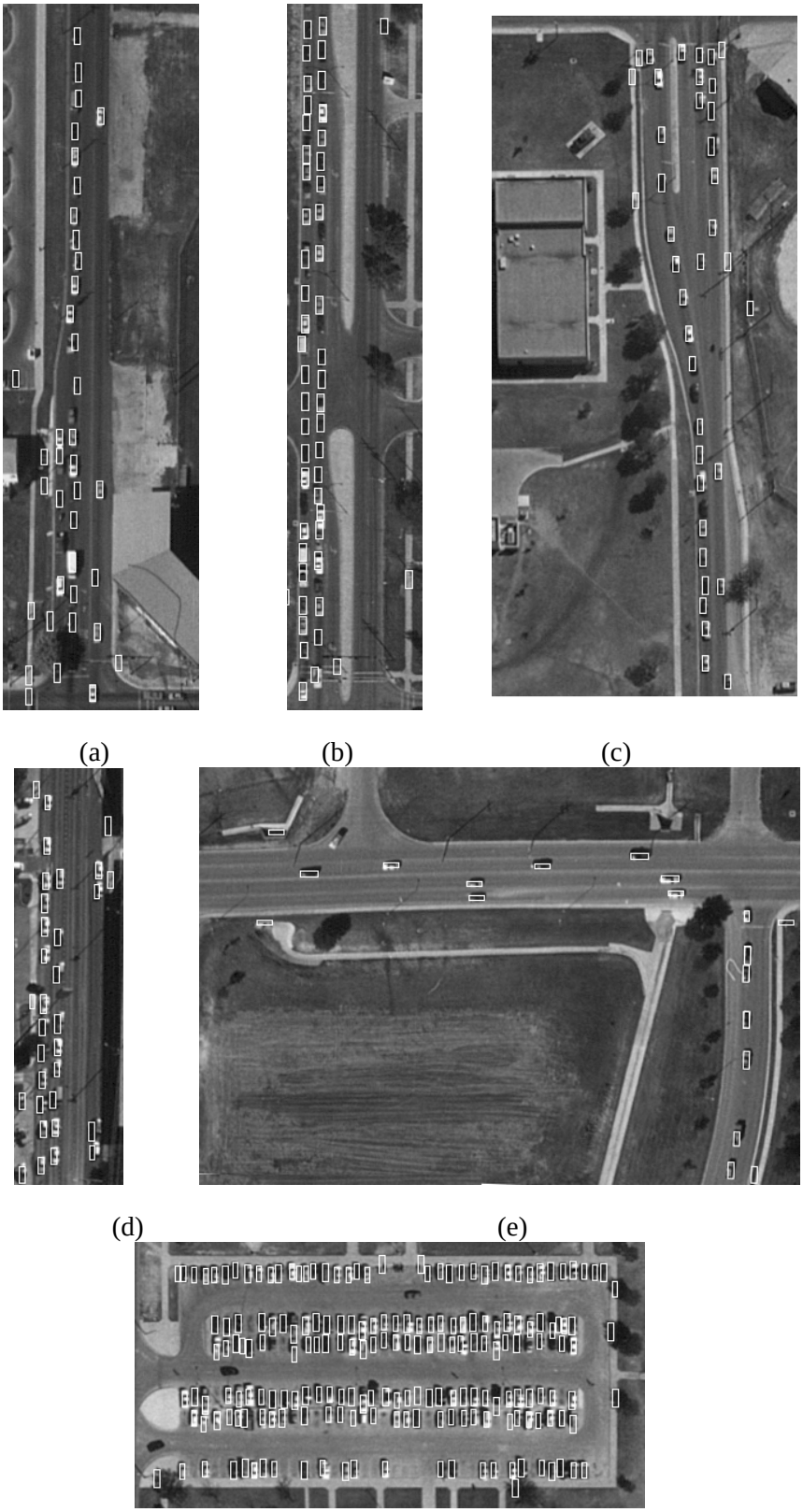


Figure 12: Some of the detection results on Fort Hood dataset.