On-Road Vehicle Detection Using Evolutionary Gabor Filter Optimization

Zehang Sun, *Member, IEEE*, George Bebis, *Member, IEEE*, and Ronald Miller

***Abstract*—Robust and reliable vehicle detection from images acquired by a moving vehicle is an important problem with numerous applications including driver assistance systems and self-guided vehicles. Our focus in this paper is on improving the performance of on-road vehicle detection by employing a set of Gabor filters specifically optimized for the task of vehicle detection. This is essentially a kind of feature selection, a critical issue when designing any pattern classification system. Specifically, we propose a systematic and general evolutionary Gabor filter optimization (EGFO) approach for optimizing the parameters of a set of Gabor filters in the context of vehicle detection. The objective is to build a set of filters that are capable of responding stronger to features present in vehicles than to nonvehicles, therefore improving class discrimination. The EGFO approach unifies filter design with filter selection by integrating genetic algorithms (GAs) with an incremental clustering approach. Filter design is performed using GAs, a global optimization approach that encodes the Gabor filter parameters in a chromosome and uses genetic operators to optimize them. Filter selection is performed by grouping filters having similar characteristics in the parameter space using an incremental clustering approach. This step eliminates redundant filters, yielding a more compact optimized set of filters. The resulting filters have been evaluated using an application-oriented fitness criterion based on support vector machines. We have tested the proposed framework on real data collected in Dearborn, MI, in summer and fall 2001, using Ford's proprietary low-light camera.**

***Index Terms*—Evolutionary computing, Gabor filter optimization, support vector machines, vehicle detection.**

I. Introduction

RECOGNIZING that vehicle safety is a primary concern for motorists, many national and international companies have launched multiyear research projects to investigate new technologies for improving safety and accident prevention [1]. Vehicle accident statistics disclose that the main threats drivers are facing are from other vehicles. Consequently, onboard automotive driver assistance systems—aiming to alert a driver about driving environments, possible collision with other vehicles, or take control of the vehicle to enable collision avoidance and mitigation—have attracted more and more attention lately. In these systems, robust and reliable vehicle detection is a required critical step.

The most common approach to vehicle detection is using active sensors such as lidar, millimeter wave radars, and lasers [1]. Prototype vehicles employing active sensors have shown promising results. However, active sensors have several drawbacks, such as low spatial resolution, slow scanning speed, and high cost. Moreover, when there is a large number of vehicles moving simultaneously in the same direction, interference among sensors of the same type poses a big problem. Passive sensors on the other hand, such as optical cameras, offer a more affordable solution and can be used to track more effectively cars entering a curve or moving from one side of the road to another. Visual information can be very important in a number of related applications, such as lane detection, traffic sign recognition, or object identification (e.g., pedestrians, obstacles), without requiring any modifications to road infrastructures. Our emphasis in this paper is on improving vehicle detection using optical sensors.

Robust and reliable vehicle detection from images acquired by a moving vehicle (i.e., on-road vehicle detection) has numerous applications including driver assistance systems, self-guided vehicles, etc. In general, vehicle detection using optical sensors is very challenging due to huge within class variabilities. For example, vehicles may vary in shape [Fig. 1(a)], size, and color. Also, vehicle appearance depends on its pose [Fig. 1(b)] and is affected by nearby objects. Complex outdoor environments, e.g., illumination conditions [Fig. 1(c)], cluttered background, and unpredictable interactions between traffic participants [Fig. 1(d)], are difficult to control.

A. Vehicle Detection Overview

Optical-sensor-based vehicle detection systems follow two basic steps: 1) hypothesis generation (HG) where the locations of possible vehicles in an image are hypothesized, and 2) hypothesis verification (HV) where tests are performed to verify the presence of vehicles in an image (see Fig. 2). The objective of HG step is to provide some candidate locations quickly for further exploration. Methods reported in the literature fall in one of the following three basic categories: 1) knowledge based, 2) stereo vision based, and 3) motion based. Knowledge-based methods employ a priori knowledge to hypothesize vehicle locations in an image such as: a) symmetry [2]–[4],

Manuscript received May 20, 2004; revised September 17, 2004. This work was supported by the Ford Motor Company under grant no. 2001332R, the University of Nevada, Reno, under an Applied Research Initiative grant, and in part by NSF under CRCD grant no. 0088086. The Associate Editor for this paper was N. Zheng.

Z. Sun was with the Computer Science and Engineering Department, University of Nevada, Reno NV 89557-0042 USA. He is now with eTreppid Technologies LLC, Reno, NV 89521 USA (e-mail: zsun@ieee.org).

G. Bebis is with the Computer Science and Engineering Department, University of Nevada, Reno, NV 89557-0042 USA (e-mail: bebis@cs.unr.edu).

R. Miller is with the Vehicle Design R&A Department, Ford Motor Company, Dearborn, MI 48126 USA (e-mail: rmille47@ford.com).

Digital Object Identifier 10.1109/TITS.2005.848363

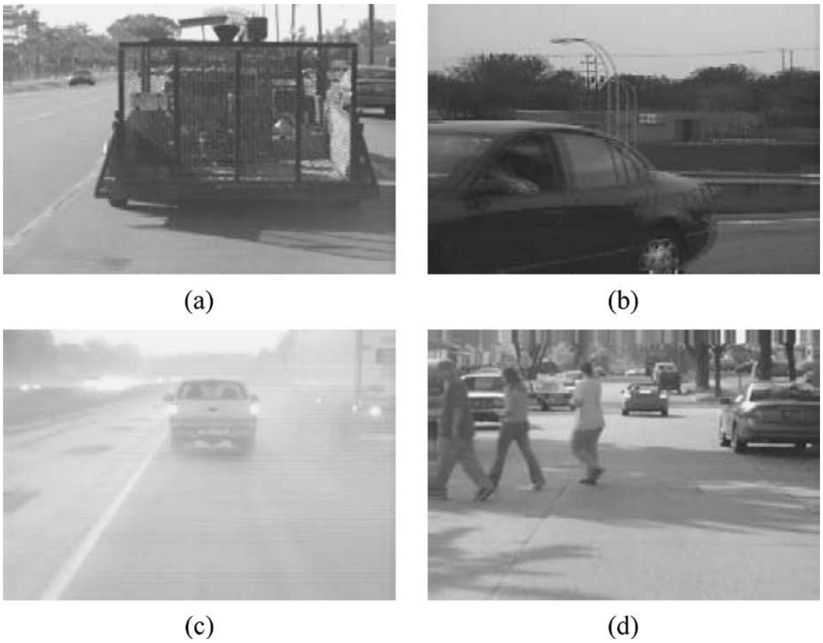


Fig. 1. A variety of vehicle appearances pose a big challenge for vehicle detection.

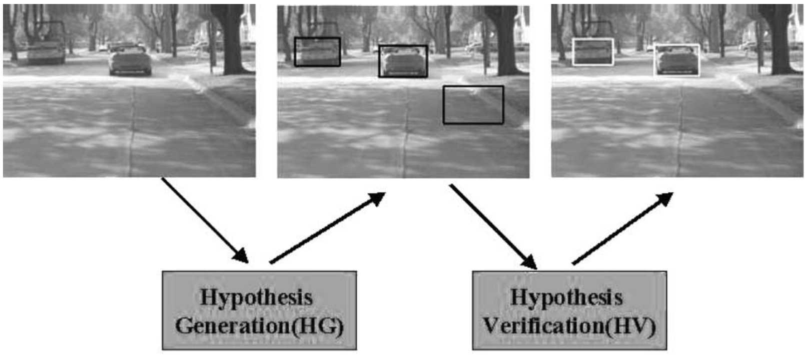


Fig. 2. Illustration of the two-step vehicle detection strategy.

b) shadow [5]–[7], c) texture [8], d) horizontal/vertical edges [9]–[12], and e) color [13]–[15]. Stereovision-based approaches take advantage of the inverse perspective mapping [16]–[19] to estimate the locations of vehicles and obstacles in images. Motion-based methods detect vehicles and obstacles using dense optical flow [20], [21] or "sparse optical flow" based on image features, such as corners [22] or local minima and maxima [23].

In the phase of HV, tests are performed to verify the correctness of each hypothesis. HV approaches can be classified into two main categories: 1) template based and 2) appearance based. Template-based methods use predefined patterns of vehicle class and perform correlation between an input image and the template. Betke *et al.* [24] proposed a multiple-vehicle detection approach using deformable gray-scale template matching. In [25], a deformable model was built from manually sampled data using principal component analysis (PCA). Both the structure and the pose of a vehicle were recovered by fitting the PCA model to the image.

Appearance-based methods learn the characteristics of the vehicle class from a set of training images, which capture the variability in vehicle appearance. Usually, the variability of the nonvehicle class is also modeled to improve performance. First, each training image is represented by a set of local or global features. Then, the decision boundary between vehicle and nonvehicle classes is learned either by training a classifier [e.g., neural network (NN)] or by modeling the probability distribution of the features in each class (e.g., using the Bayes rule assuming Gaussian distributions). In [9], PCA was used for feature extraction and NNs for classification. Goerick *et al.* [26] used a method called local orientation coding (LOC) to extract edge information. The histogram of LOC within the area of interest was then provided to a NN for classification. A statistical model for vehicle detection was investigated by Schneiderman *et al.* [27], [28]. A view-based approach based on multiple detectors was used to cope with viewpoint variations. The statistics of both object and "nonobject" appearances were represented using the product of two histograms with each histogram representing the joint statistics of a subset of PCA features in [27] or Haar wavelet features in [28] and their position on the object. A different statistical model was investigated by Weber *et al.* [29]. They represented each vehicle image as a constellation of local features and used the expectation–maximization algorithm to learn the parameters of the probability distribution of the constellations. An interest operator, followed by clustering, was used to identify important local features in vehicle images. Papageorgiou and Poggio [30] have proposed using the Haar wavelet transform for feature extraction and support vector machines (SVMs) for classification.

B. *Feature Selection for Vehicle Detection*

On-road vehicle detection involves "concepts," rather than a specific vehicle, that is, we need to detect any vehicle regardless of its maker, model, color, etc. This "conceptual vehicle" has large within class variabilities; therefore, there is no easy way to come up with an analytical decision boundary to separate vehicles from other objects. One feasible approach is to learn the decision boundary of the vehicle class from a set of training examples using supervised learning where each training instance is associated with a class label (i.e., vehicle versus nonvehicle) [9], [26], [30].

Building a vehicle detection system under the supervised learning framework involves two main steps: 1) extracting a number of features (e.g., PCA features [9], wavelet features [28], Gabor features [31], etc.) and 2) training a classifier (e.g., NNs [9], SVMs [32], modified quadratic discriminant function [33], etc.) using the extracted features to distinguish between vehicle and nonvehicle classes. A key issue with this approach is selecting a number of appropriate features. In most cases, relevant features are unknown. Often, a large number of features are extracted to better represent the target; however, without explicitly employing a feature selection strategy, many of them could be either redundant or even irrelevant to the classification task. As a result, classification performance might not be optimum.

Watanabe [34] has shown that it is possible to make two arbitrary patterns similar by encoding them with a sufficiently large number of redundant features. Ideally, we would like to use only features having high separability power while ignoring or paying less attention to the rest. In the context of vehicle detection, it would be desirable if we could exclude features encoding fine details (i.e., features that might be present in particular vehicles only). Finding out what features to use in a classification task is referred to as feature selection. A limited yet salient feature set can simplify both the pattern representation and the classifiers that are built on the selected representation.

In our recent work, we have investigated the application of Gabor features for vehicle detection, demonstrating their superiority compared to other features including PCA and wavelet features [12], [31], [32], [35]. Like others, we employed a generic Gabor filter bank for feature extraction. To improve classification performance, however, it would be critical selecting an optimum set of features and, consequently, an optimum set of Gabor filters. This raises the problem of Gabor filter optimization. Despite considerable amount of work on the application of Gabor filters in various pattern classification tasks, their design and selection have not been systematic. Existing techniques are either only suitable for a small number of filters or problem oriented.

C. *Proposed Approach*

An evolutionary Gabor filter optimization (EGFO) approach is proposed in this paper. The EGFO approach unifies filter design with filter selection by integrating genetic algorithms (GAs) with an incremental clustering approach. GAs allow for searching the space of filter parameters efficiently while clustering removes redundant filters. Specifically, filter design is performed using GAs, a global optimization approach that encodes the parameters of the Gabor filters in a chromosome and uses genetic operators to optimize them. Filter selection is performed by grouping together filters having similar characteristics (i.e., similar parameters) using incremental clustering in the parameter space. Each group of filters is represented by a single filter whose parameters correspond to the average parameters of the filters in the group. This step eliminates redundant filters, leading to a compact optimized set of filters. The average filters are evaluated using an application-oriented fitness criterion based on SVMs.

The EGFO approach is suitable for optimizing any number of filters for a given application. The search space of our method is much larger than those of the existing methods (see Section II for a review), providing a higher likelihood of getting close to the optimal solution. Moreover, we represent filter optimization as a closed-loop learning problem. The search for an optimal solution is guided by the performance of a classifier on features extracted from the responses of the Gabor filters. We use SVMs in this paper.

The rest of the paper is organized as follows. In Section II, we present a brief review of Gabor filters, their design, and optimization methods. Section III presents our EGFO approach in detail. The Gabor filter feature extraction method and the learning engine used in our experiments are described in Section IV. Experiments and results are presented in Section VI. A discussion of our experimental results is given in Section VII. Finally, Section VIII contains our conclusions and directions for future work.

II. Gabor Filter Design

Motivated by biological findings on the similarity of two-dimensional (2-D) Gabor filters and receptive fields of neurons in the visual cortex [36], there has been increased interest in deploying Gabor filters in various computer vision applications. One of their most important properties is that they have optimal joint localization in both spatial and frequency domains [36]. Gabor filters have been successfully applied to various image analysis applications including edge detection [37], image coding [36], texture analysis [38]–[40], handwritten number recognition [41], face recognition [42], vehicle detection [32], and image retrieval [43].

The general functionality of the 2-D Gabor filter family can be represented as a Gaussian function modulated by a complex sinusoidal signal. Specifically, a 2-D Gabor filter $g(x, y)$ can be formulated as

$$g(x,y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left[-\frac{1}{2}\left(\frac{\tilde{x}^2}{\sigma_x^2} + \frac{\tilde{y}^2}{\sigma_y^2}\right)\right] \exp[2\pi jW\tilde{x}] \quad (1)$$

$$\begin{cases} \tilde{x} = x\cos\theta + y\sin\theta \\ \tilde{y} = -x\sin\theta + y\cos\theta \end{cases} \quad (2)$$

where σ_x and σ_y are the scaling parameters of the filter and determine the effective size of the neighborhood of a pixel in which the weighted summation takes place. $\theta(\theta \in [0, \pi))$ specifies the orientation of the Gabor filters. W is the radial frequency of the sinusoid. A filter will respond stronger to a bar or an edge with a normal parallel to the orientation θ of the sinusoid. The Fourier transform of the Gabor function in (1) is given by

$$G(u,v) = \exp\left[-\frac{1}{2}\left(\frac{(u-W)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2}\right)\right] \quad (3)$$

where $\sigma_u = 1/2\pi\sigma_x$, $\sigma_v = 1/2\pi\sigma_y$. The Fourier domain representation in (3) specifies the amount by which the filter modifies each frequency component of the input image.

Gabor filters act as local bandpass filters. Fig. 3 shows four Gabor filters with different parameter settings in frequency domain. The light areas of the power spectrum indicate frequencies and wave orientations. It is obvious from Fig. 3 that different parameter settings will lead to quite different filter responses, an important issue in pattern classification problems. Each filter is fully determined by choosing the four parameters in $\Phi = \{\theta, W, \sigma_x, \sigma_y\}$. Therefore, choosing a filter for a particular application involves optimizing these four parameters. Assuming that N filters are needed in an application, $4N$ parameters need to be optimized. Solving this high-dimensional multivariate optimization problem is very difficult in general.

Previous efforts in designing Gabor filters follow two main directions: the "Filter design approach" and the "Filter bank approach" [38], [44]. In the "filter design approach" the filter parameters are chosen by considering the data available, that is, the parameters are appropriate for the problem at hand only. In one of the pioneering studies on the design of Gabor filters conducted by Bovik *et al.* [45], the peak detection technique was used. Okombi-Diba *et al.* [46] implemented a multi-iteration peak detection method for a texture segmentation problem. Dunn and Higgins [47] investigated an exhaustive search to find the center frequency. Due to the exhaustive search, this method is quite time consuming. A more computationally efficient method was described in [38], [44] using a segmentation error criterion similar to [47].

In the "filter bank approach," first, the filter parameters are chosen in a data independent way. Then, a subset of filters is selected for a particular application. Turner [48] used 32 filters (four frequencies × four orientations × two phase pairs) in a texture discrimination problem. Jain and Farrokhnia [39] chose the filter parameters such that the radial frequencies were one octave apart. To reduce the computational burden, a greedy filter selection method was employed. To reduce the redundancy in the Gabor feature representation, Manjunath and Ma [43] proposed a design method to ensure that the half-peak magnitude supports of the filter responses in the frequency domain touch each other. For fast image browsing, they implemented an "adaptive filter selection algorithm," where spectrum difference information was used to select filters with better performance. In the context of handwritten number recognition, Hamamoto *et al.* [41] optimized the filters by checking the error rate for all possible combinations of filter parameters and then choosing those minimizing the error rates.

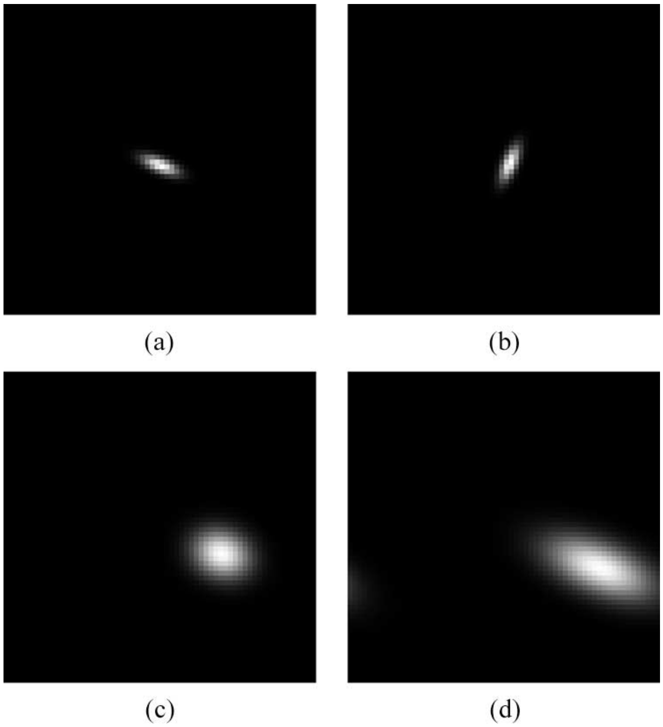

(a) (b) (c) (d)

Fig. 3. Gabor filters with different parameters $\Phi = \{\theta, W, \sigma_x, \sigma_y\}$ in the frequency domain (i.e., the Fourier transform of the Gabor functions). (a) $\Phi_a = \{0°, 0.0961, 0.0204, 0.01219\}$, (b) $\Phi_b = \{0°, 0.3129, 0.06, 0.359\}$, (c) $\Phi_b = \{90°, 0.3129, 0.06, 0.359\}$, (d) $\Phi_c = \{90°, 0.3921, 0.0503, 0.3066\}$.

Although good performances have been reported in the literature, certain limitations still exist. "Filter design approaches," for example, divide the design process into two stages: prefilter and postfilter. Several prefilter design approaches have been investigated; however, an explicit methodology for selecting an appropriate postfilter step for a given prefilter step has not been suggested. Moreover, the selection of the bandwidth parameter is done mostly heuristically. The design stage in the "filter bank approach" is mostly problem independent. Different pattern classification problems, however, might require selecting an optimum set of features and, consequently, an optimum set of Gabor filters. We would not expect, for example, that a set of Gabor filters optimized for a vehicle classification application (compact car versus truck) would work well in a vehicle detection application (vehicle versus nonvehicle), since more detailed information is required in the former case than in the latter.

Many researchers have realized that this is a serious problem and have suggested filter selection schemes to deal with it; however, filters are selected from an original small pool of filters that might not be suitable for the problem at hand (e.g., Hamamoto *et al.* [41] performed filter selection using a pool of 100 predefined filters). The main issue with this approach is that there is no guarantee that the optimum set of filters would be included in the predefined pool of filters.

III. EGFO

In this section, we describe the proposed EGFO approach. Gabor filter optimization corresponds to selecting the proper values for each of the four parameters in the parameter set $\Phi = \{\theta, W, \sigma_x, \sigma_y\}$.

A. A Brief Review of GAs

GAs are a class of optimization procedures inspired by the biological mechanisms of reproduction. In the past, they have been used to solve various problems including target recognition [49], object recognition [50], [51], face recognition [52], and face detection/verification [53]. This section contains a brief summary of the fundamentals of GAs. Goldberg [54] provides a great introduction to GAs and the reader is referred to this source as well as to the survey paper of Srinivas and Patnaik [55] for further information.

GAs operate iteratively on a population of structures, each of which represents a candidate solution to the problem at hand, properly encoded as a string of symbols (e.g., binary). A randomly generated set of such strings forms the initial population from which the GA starts its search. Three basic genetic operators guide this search: selection, crossover, and mutation. The genetic search process is iterative: evaluating, selecting, and recombining strings in the population during each iteration (generation) until reaching some termination condition.

Evaluation of each string is based on a fitness function that is problem dependent. It determines which of the candidate solutions are better. Selection of a string, which represents a point in the search space, depends on the string's fitness relative to those of other strings in the population. It probabilistically removes, from the population, those points that have relatively low fitness. Mutation, as in natural systems, is a very low probability operator and just flips a specific bit. Mutation plays the role of restoring lost genetic material. Crossover in contrast is applied with high probability. It is a randomized yet structured operator that allows information exchange between points. Its goal is to preserve the fittest individuals without introducing any new value.

GAs do not guarantee a global optimum solution. However, they have the ability to search through very large search spaces and come to nearly optimal solutions fast. Their ability for fast convergence is explained by the schema theorem (i.e., short-length bit patterns in the chromosomes with above-average fitness get exponentially growing number of trials in subsequent generations [54]).

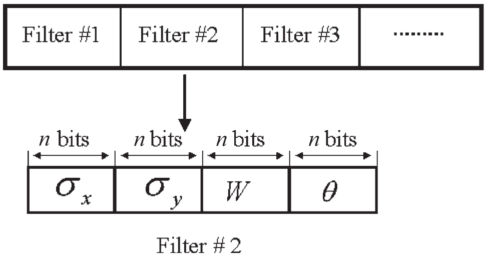


Fig. 4. Encoding scheme.

B. Parameter Encoding/Decoding

Using a binary encoding scheme, each Gabor filter is represented by M bits that encode its four parameters. To design N filters, we use a chromosome of length MN bits. Each of the four parameters in Φ is encoded using $n = M/4$ bits as illustrated in Fig. 4. It is worth mentioning that the encoding scheme is quite flexible and allows us to encode any number of filters by simply varying the length of the chromosome. The numbers of bits associated with each parameter need not be the same, we can make the search for a particular parameter finer or coarser by simply adding or removing bits for this parameter. If we need to fix certain parameter(s) using prior knowledge, we can remove the parameter(s) from the chromosome. In this case, the GA will optimize the remaining parameters. Each of the parameters in Φ has its own constraints and ranges. The encoding/decoding scheme was designed to ensure that the generated filters satisfy these requirements.

The orientation parameter θ should satisfy $\theta \in [0, \pi)$. If D_θ denotes the decimal number corresponding to the chunk of bits associated with θ (see Fig. 4), then the value of θ is computed by

$$\theta = \frac{D_\theta \pi}{2^n}. \tag{4}$$

that always satisfies the range requirement.

W is the radial frequency of the Gabor filter, which is application dependent. Using some prior knowledge, we can limit the range of W into $[W_{\min}, W_{\max}]$. Then the decoding formula is given by

$$W = W_{\min} + \frac{(W_{\max} - W_{\min})D_W}{2^n} \tag{5}$$

where D_W is the decimal number corresponding to the chunk of bits associated with W. In this study, we have used $W_{\min} = 0$ and $W_{\max} = 0.5$.

σ_x and σ_y are essentially the effective sizes of the Gaussian functions and are within the range $[\sigma_{\min}, \sigma_{\max}]$. The upper limit $\sigma_{\max}$ is determined by the mask width w [56]. A relation between $\sigma_{\max}$ and the mask size w can be obtained by imposing that w subtends most of the energy of the Gaussian. An adequate choice is $\sigma_{\max} < w/5$, which subtends 98.76% of the energy of the Gaussian filter. The lower limit can be derived using the sampling theorem. If the pixel width is taken as our unit step, we cannot reconstruct completely a signal containing frequencies higher than 0.5 pixel^{-1} from its samples, which means that any frequency component at $|\omega| > \omega_{\mathrm{c}} = 2\pi(0.5) = \pi$ is distorted. The ω_{c} is determined by the pixelization, not by the signal. The Fourier transform of the Gaussian function $g(x, \sigma) = e^{-x^2/2\sigma^2}$ is $F_x(k) = \sqrt{2\pi\sigma^2} e^{-(\pi k)^2 2\sigma^2} = g(\omega, \sigma')$, which is also a Gaussian function. Also, we have $\sigma = 1/\sigma'$. To avoid aliasing, we need to keep most of the energy of the Gaussian function $g(\omega, \sigma')$ within the interval $[-\pi, \pi]$. Applying the "98.76% of the energy" criterion, we have $5\sigma' = 5/\sigma \leq 2\pi$ or $\sigma \geq 5/2\pi = 0.796$. To meet the range constraint ($[\sigma_{\min}, \sigma_{\max}]$), our decoding scheme follows

$$\sigma_x = \sigma_{\min} + \frac{(\sigma_{\max} - \sigma_{\min}) D_{\sigma_x}}{2^n} \tag{6}$$

for σ_x and

$$\sigma_y = \sigma_{\min} + \frac{(\sigma_{\max} - \sigma_{\min}) D_{\sigma_y}}{2^n} \tag{7}$$

for σ_y. D_{σ_x} and D_{σ_y} are again the decimal numbers corresponding to the chunk of bits associated with σ_x and σ_y correspondingly.

C. Eliminating Redundant Filters Through Clustering

During parameter optimization, some of the Gabor filters encoded in a chromosome might end up being very similar to each other or even identical. These filters will result in similar/identical responses, therefore introducing great redundancy and increasing time requirements. To eliminate redundant filters, we perform filter selection, implemented through filter clustering in the parameter space. An incremental clustering algorithm [57] has been adopted in this paper for its simplicity. A high-level description of the clustering algorithm is given below.

1) Assign the first Gabor filter to a cluster.
2) Compute the distance of the next Gabor filter from the centroid of each cluster.
3) Find the smallest distance.
4) If the distance is less than a threshold, assign the filter to the corresponding cluster; otherwise, assign the filter to a new cluster.
5) Repeat steps 2–4 for each of the remaining filters.
6) Represent the filters in each cluster by a single filter whose parameters correspond to the cluster's centroid.

As it can be depicted from the above pseudocode, if a filter lies within a predefined range/threshold from a cluster, it is added to that cluster. Otherwise, it is used to create a new cluster. This incremental clustering is a one-pass approach, very simple and efficient. We have obtained satisfactory results using this method as shown in Section VI. We envision that a more sophisticated clustering approach will produce even better results at the expense of higher computational burden.

The optimized filters are evaluated using the fitness function defined in Section III-D. In our implementation, clustering is carried out in the parameter domain. Representing the parameters of a Gabor filter with $\{\theta^n, W^n, \sigma_x^n, \sigma_y^n\}$ and the centroid of the clusters with $\{\theta_{\mathrm{c}}^i, W_{\mathrm{c}}^i, \sigma_{\mathrm{c}x}^i, \sigma_{\mathrm{c}y}^i\}$ with $i \in [1\ N]$, where N is the number of currently existing clusters, we assign the filter to the ith cluster only if all of the following conditions are satisfied

$$\theta_{\mathrm{c}}^i - \frac{1}{2} \times \mathrm{Thr}_\theta \leq \theta^n \leq \theta_{\mathrm{c}}^i + \frac{1}{2} \times \mathrm{Thr}_\theta \tag{8}$$

$$W_{\mathrm{c}}^i - \frac{1}{2} \times \mathrm{Thr}_W \leq W^n \leq W_{\mathrm{c}}^i + \frac{1}{2} \times \mathrm{Thr}_W \tag{9}$$

$$\sigma_{\mathrm{c}x}^i - \frac{1}{2} \times \mathrm{Thr}_\sigma \leq \sigma_x^n \leq \sigma_{\mathrm{c}x}^i + \frac{1}{2} \times \mathrm{Thr}_\sigma \tag{10}$$

$$\sigma_{\mathrm{c}y}^i - \frac{1}{2} \times \mathrm{Thr}_\sigma \leq \sigma_y^n \leq \sigma_{\mathrm{c}y}^i + \frac{1}{2} \times \mathrm{Thr}_\sigma. \tag{11}$$

Otherwise, the filter is assigned to a new cluster. The above conditions are quite strict to make sure that filters falling in the same cluster are very similar to each other. We can always relax the criterion by increasing the predefined thresholds. The following thresholds were used in our experiments: $\mathrm{Thr}_\theta = \pi/K$, $\mathrm{Thr}_W = (W_{\max} - W_{\min})/K$, and $\mathrm{Thr}_{\sigma_x} = \mathrm{Thr}_{\sigma_y} = \mathrm{Thr}_\sigma = (\sigma_{\max} - \sigma_{\min})/K$. The value of K determines the number of "bins" we are going to divide the parameter range into. The bigger the K is, the more "bins" we have, the more compact the clusters are. Depending on different applications and the desired tradeoff between model compactness and accuracy, K can be set to different values.

D. Fitness Evaluation

Each individual's fitness will determine whether or not it will survive in subsequent generations. The fitness value used here is the performance of an SVM classifier on a validation set using features extracted from the responses of the selected Gabor filters. In this way, the Gabor filter optimization design is implemented as a closed-loop learning scheme, which is more powerful, more problem specific, and less heuristic than previous approaches.

E. Initial Population

The initial population is generated randomly (i.e., each bit in an individual is set by flipping a coin). In all of our experiments, we used a population size of 700 and 100 generations. In most cases, the GA converged in less than 100 generations.

F. Selection

Our selection strategy was cross generational. Assuming a population of size N, the offspring double the size of the population and we select the best N individuals from the combined parent–offspring population [58].

G. Crossover

There are three basic types of crossovers: one-point crossover, two-point crossover, and uniform crossover. For one-point crossover, the parent chromosomes are split at a common point chosen randomly and the resulting subchromosomes are swapped. For two-point crossover, the chromosomes are thought of as rings with the first and last gene connected (i.e., wrap-around structure). In this case, the rings are split at two common points chosen randomly and the resulting subrings are swapped. Uniform crossover is different from the above two schemes. In this case, each gene of the offspring is selected randomly from the corresponding genes of the parents. Since we do not know in general how parameters from different filters depend on each other, if dependent parameters are far apart in the chromosome, it is very likely that traditional one-point or two-point crossover will destroy the schemata. To avoid this problem, uniform crossover is used here. The crossover probability used in all of our experiments was 0.66.

H. Mutation

We use the traditional mutation operator that just flips a specific bit with a very low probability. The mutation probability used in all of our experiments was 0.03.

IV. Gabor Feature Extraction and Classification

Designing an optimal set of Gabor filters is the first step in building a pattern classification algorithm. Then, we need to extract features using the responses of the selected filters and train a classifier using those features. To demonstrate the proposed filter design approach, redundant statistical Gabor features and SVMs are utilized.

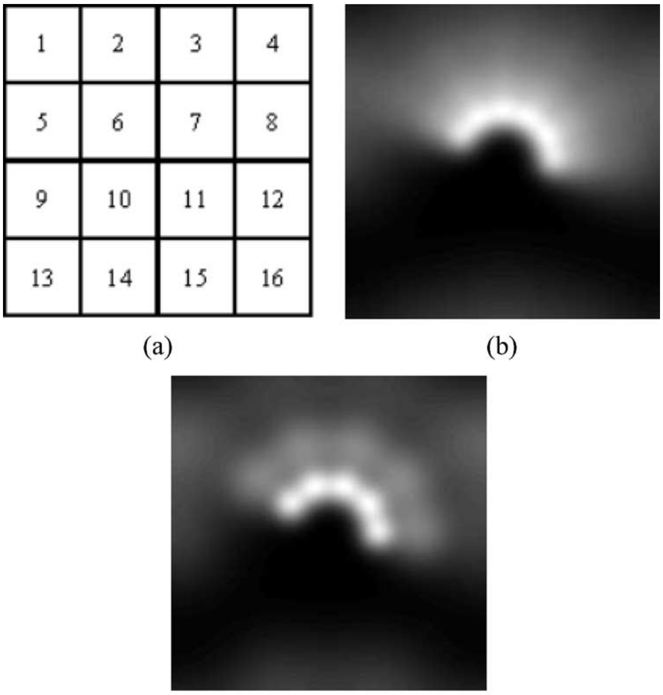
A. Gabor Filter Features

Given an input image $I(x,y)$, Gabor feature extraction is performed by convolving $I(x,y)$ with a set of Gabor filters

$$r(x,y) = \iint I(\xi,\eta)g(x-\xi, y-\eta)\,\mathrm{d}\xi\,\mathrm{d}\eta. \qquad (12)$$

Although the raw responses of the Gabor filters could be used directly as features, some kind of postprocessing is usually applied (e.g., Gabor energy features, thresholded Gabor features, and moments based on Gabor features [59]). In this study, we use moments derived from Gabor filter outputs on subwindows defined on subimages extracted from the whole input image.

First, each subimage is scaled to a fixed size of 32 × 32. Then, it is divided into nine overlapping 16 × 16 subwindows. Each subimage consists of sixteen 8 × 8 patches as shown in Fig. 5(a), patches 1, 2, 5, and 6 comprise the first 16 × 16 subwindow, 2, 3, 6, and 7 the second, 5, 6, 9, and 10 the fourth, and so forth. The Gabor filters are then applied on each subwindow separately. The motivation for extracting—possibly redundant—Gabor features from several overlapping subwindows is to compensate for the error due to the subwindow extraction step (e.g., subimages containing partially extracted objects or background information), making feature extraction more robust.



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

(a) (b)

(c)

Fig. 5. (a) Feature extraction patches. (b) Gabor filter bank with four scales and six orientations. (c) Gabor filter bank with three scales and five orientations.

The magnitudes of the Gabor filter responses are collected from each subwindow and represented by three moments: the mean μ_{ij}, the standard deviation σ_{ij}, and the skewness κ_{ij}, where i corresponds to the ith filter and j corresponds to the jth subwindow. We have investigated different combinations of various moments in our past work; however, we have found that the triplet (μ, σ, κ) gives the best performance [31], [32]. Using moments implies that only the statistical properties of a group of pixels are taken into consideration, while position information is discarded. This is particularly useful to compensate for errors in the extraction of the subimages. Suppose we are using $N = 6$ filters. Applying the filter bank on each of the nine subwindows yields a feature vector of size 162, having the following form

$$[\mu_{11}\sigma_{11}\kappa_{11}, \mu_{12}\sigma_{12}\kappa_{12} \ldots \mu_{69}\sigma_{69}\kappa_{69}] \qquad (13)$$

B. SVM Classifier

SVMs are primarily two-class classifiers that have been shown to be an attractive and more systematic approach to learning linear or nonlinear decision boundaries [60], [61].

Given a set of points that belong to either one of the two classes, SVM finds the hyperplane leaving the largest possible fraction of points of the same class on the same side while maximizing the distance of either class from the hyperplane. This is equivalent to performing structural risk minimization to achieve good generalization [60], [61]. Given l examples from two classes

$$(x_1, y_1)(x_2, y_2)\ldots(x_l, y_l), \qquad x_i \in R^N \quad y_i \in \{-1, +1\} \quad (14)$$

finding the optimal hyperplane implies solving a constrained optimization problem using quadratic programming. The optimization criterion is the width of the margin between classes. The discriminating hyperplane is defined as

$$f(x) = \sum_{i=1}^{l} y_i a_i k(x, x_i) + b \qquad (15)$$

where $k(x, x_i)$ is a kernel function and the sign of $f(x)$ indicates the membership of x. Constructing the optimal hyperplane is equivalent to finding all the nonzero a_i. Any data point x_i corresponding to a nonzero a_i is a support vector of the optimal hyperplane.

Kernel functions, which satisfy Mercer's condition, can be expressed as a dot product in some space [60]. By using different kernels, SVMs implement a variety of learning machines (e.g., a sigmoidal kernel corresponds to a two-layer sigmoidal NN while a Gaussian kernel corresponds to a radial basis function NN). The Gaussian radial basis kernel, which is used in this study, is given by

$$k(x, x_i) = \exp\left(-\frac{\|x - x_i\|^2}{2\delta^2}\right) \qquad (16)$$

Our experiments with different kernels have shown that the Gaussian kernel outperforms the others in the context of our application.

Given the real-time constraints for a vehicle detection system, a natural question might be why we decided to use the costly SVM classifier instead of some simple classifier, for instance, a nearest neighbor classifier. In our previous work [31], [32], [35], [62], [63], we have investigated various feature extraction/selection methods along with different classifiers (e.g., NNs, Bayes classifier, Fischer discriminant function, and SVM). Comparisons across different classifiers have shown that the SVMs achieve the best performance. It should be mentioned that we have already implemented a real-time vehicle detection system using a traditional Gabor filter bank and SVMs [11] (see also Section VII).

V. Vehicle Detection Using Optimized Gabor Filters

In this section, we consider the problem of vehicle detection from gray-scale images. The first step in vehicle detection is usually to hypothesize the vehicle locations in an image. Then, verification is applied to test the hypotheses, as we have discussed in Section I. Our emphasis in this paper is on improving the performance of the verification step by optimizing the Gabor filters. Therefore, we assume that the hypothesized candidate windows are already available. For completeness, we discuss briefly below the HG step.

A. Hypothesizing Vehicle Locations

An edge-based HG method has been proposed in our previous work [11]. It is a multiscale approach that combines subsampling with smoothing to hypothesize possible vehicle locations more robustly. Assuming that the input image is f, let $f^{(K)} = f$. The representation of $f^{(K)}$ at a coarser level $f^{(K-1)}$ is defined by a reduction operator. The size of the input images was 360 × 248. We used three levels of detail: $f^K(360 \times 248)$, $f^{K-1}(180 \times 124)$, and $f^{K-2}(90 \times 62)$. At each level, we process the image by applying the following operations: i) low-pass filtering; ii) vertical edge detection, vertical profile computation of the edge image, and profile filtering using a low-pass filter; iii) horizontal edge detection, horizontal profile computation of the edge image, and profile filtering using a low-pass filter; and iv) local maxima and minima detection (e.g., peaks and valleys) of the two profiles. The peaks and valleys of the profiles provide strong information about the presence of a vehicle in the image.

Starting from the coarsest level of detail (f^{K-2}), first we find all the local maxima at that level. Although the resulting low-resolution images have lost fine details, important vertical and horizontal structures are mostly preserved. Once we have found the maxima at the coarsest level, we trace them down to the next finer level f^{K-1}. The results from f^{K-1} are finally traced down to level f^K where the final hypotheses are generated. It should be mentioned that due to the complexity of the scenes, some false peaks are expected. We used several heuristics to get rid of them, for example, the ratio of successive maxima and minima, the absolute value of a maximum, and perspective projection constraints under the assumption of flat surface (i.e., road). These rules were applied at each level of detail.

B. Vehicle Data

The images used in our experiments were collected in Dearborn, MI, in two different sessions, one in the summer of 2001 and one in the fall of 2001. To ensure a good variety of data in each session, the images were captured at different times of different days and on five different highways. The training set contains subimages of rear vehicle views and nonvehicles, which were extracted manually from the fall 2001 data set. A total of 1051 vehicle and 1051 nonvehicle subimages were extracted manually (see Fig. 6). In [30], the subimages were aligned by warping the bumpers to approximately the same position. However, we have not attempted to align the data since alignment requires detecting certain features on the vehicle accurately. Moreover, we believe that some variability in the extraction of the subimages can actually improve performance. Each subimage in the training and test sets was scaled to a size

Fig. 6. Examples of vehicle and nonvehicle images used for training.

TABLE I
VEHICLE DETECTION ERROR RATES USING DIFFERENT FILTERS. THE NUMBERS IN PARENTHESES INDICATE THE NUMBER OF OPTIMIZED FILTERS

	3×5	4×6	EGFO
Data Set1	10.82%	9.09%	6.93%(21)
Data Set2	11.69%	11.26%	7.79%(18)
Data Set3	8.66%	6.93%	4.33%(19)
Average	10.38%	9.09%	6.36%(19.3)

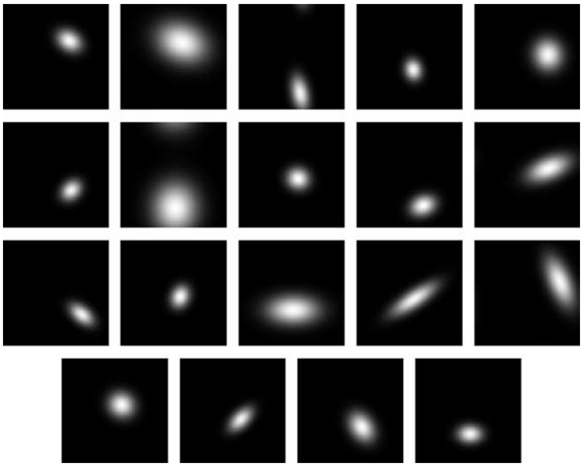


Fig. 7. Nineteen optimized Gabor filters using $K = 3$.

of 32×32 and preprocessed to account for different lighting conditions and contrast using the method suggested in [53].

In this study, we used a threefold cross validation strategy to estimate the true performance of the proposed system. Theoretically, the true performance of a learning system is statistically defined as the performance of the classifier on an asymptotically large number of unseen data that converge in the limit to the actual population distribution of a certain class. In practice, however, the number of samples is always finite, and typically relatively small, making it always impossible to reach the true performance. Several evaluation methods are often used to estimate the true performance, including cross validation, leaving one out, and bootstrap. A convenient rule of thumb is to choose the method according to the number of available samples [64]: if the number of training data is more than 100, use cross validation; less than 100, use leaving one out; and less than 50, use bootstrap.

Given that we have more than 100 training data, the error rates (ER) were recorded using a threefold cross validation procedure. Specifically, we sample the training data set randomly three times (set 1, set 2, and set 3) by keeping 280 of the vehicle subimages and 280 of the nonvehicle subimages for training. Three hundred subimages (150 vehicle subimages and 150 nonvehicle subimages) were used for validation during the filter optimization. For testing, we used a fixed set of 231 vehicle and nonvehicle subimages that were extracted from the summer 2001 data set.

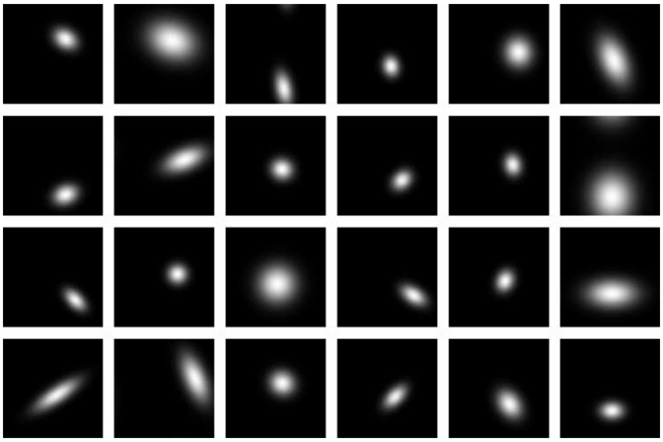


Fig. 8. Twenty-four optimized Gabor filters without clustering.

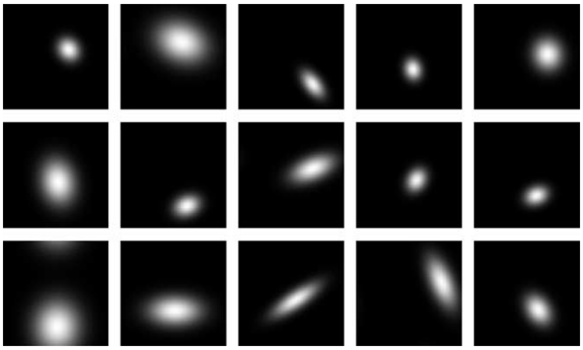


Fig. 9. Fifteen optimized Gabor filters with $K = 2$.

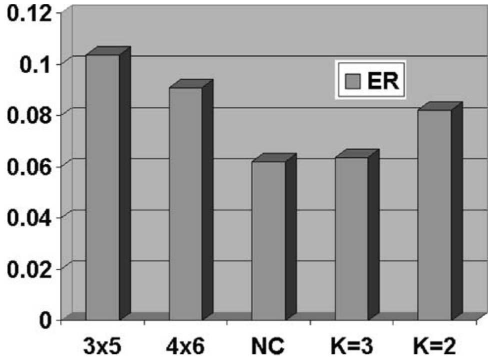


Fig. 10. Vehicle detection error rate. 3×5: Gabor filter bank with three scales and five orientations; 4×6: Gabor filter bank with four scales and six orientations; NC: EGFO method without clustering; $K = 3$: EGFO method using clustering with $K = 3$; and $K = 2$: EGFO method using clustering with $K = 2$.

VI. EXPERIMENTAL RESULTS

For comparison purposes, we also report the detection error rates using two different Gabor filter banks without optimization: one with four scales and six orientations [Fig. 5(b)], the

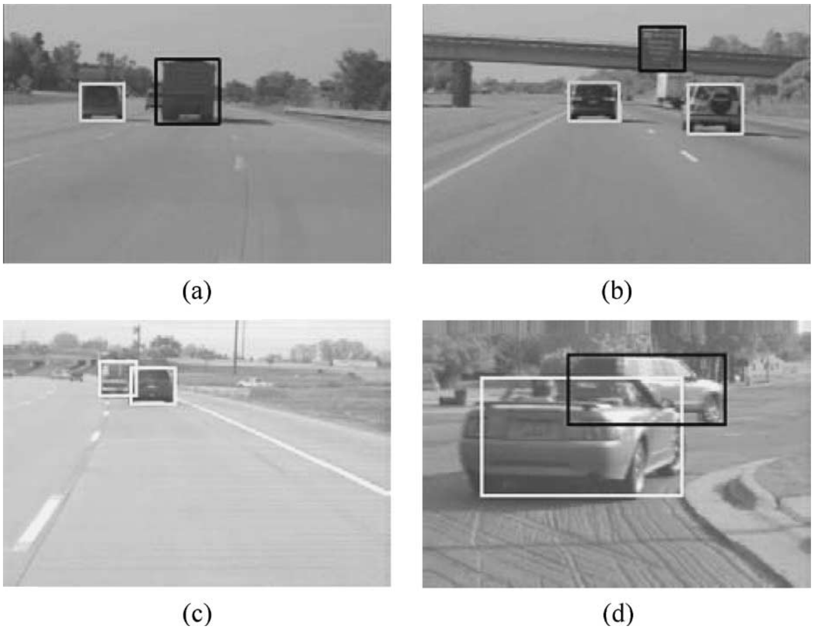


Fig. 11. Some vehicle detection results. The white box indicates correct classifications, while the black box indicates incorrect classifications (i.e., vehicle classified as nonvehicle or nonvehicle classified as vehicle).

other with three scales and five orientations [Fig. 5(c)]. These filter banks were designed by following the method proposed in [43].

We have carried out a number of experiments and comparisons to demonstrate the proposed Gabor filter optimization approach in the context of vehicle detection. First, a Gabor filter bank with three scales and five orientations was tested using SVMs for classification. Using the feature extraction method described in Section IV-A, the size of each Gabor feature vector was 405 in this experiment. The average error rate was found to be 10.38% (see Table I). Then, we tested a Gabor filter bank with four scales and six orientations that yielded feature vectors of size 648. The error rate in this case was 9.09% that is slightly better.

Second, we used the EGFO approach to customize a group of filters, up to 24, for the vehicle detection problem. We limited the number of filters to 24 to make the comparison with the traditional filter bank design methods fair. Each parameter in $\Phi = \{\theta, W, \sigma_x, \sigma_y\}$ was encoded using 4 b. The total length of the chromosome was $384(4 \times 4 \times 24)$, which corresponds to a large search space (i.e., 2^{384}). The threshold factor K for the clustering was set to 3 in our experiments. The average error rate in this case was 6.36%, and the average number of customized filters was 19.3. The optimized 19 filters generated for set 3 are shown in Fig. 7. The individual results from the three data sets are shown in Table I. Fig. 10 shows the average detection error rates for all methods. The 19 filters are displayed in Fig. 7 in the frequency domain.

For comparison purposes, we also ran the filter optimization method without clustering on the same data sets using the same parameters. Fig. 8 shows the final 24 filters obtained. The average error rate was 6.36% with clustering and 6.19% without clustering using a threefold cross validation. The difference (0.17%) is not statistically significant and the performances using filters with and without clustering can be considered the same. The main advantage of using clustering is that it produces a more compact set of filters that is critical in a real-time system.

To get an idea regarding the effectiveness of the clustering subcomponent, we performed more experiments using different threshold settings for the factor $K = 2$. The average error rate was 8.23% and the average number of customized filter was 14.7. The 15 filters generated for set 3 are shown in Fig. 9.

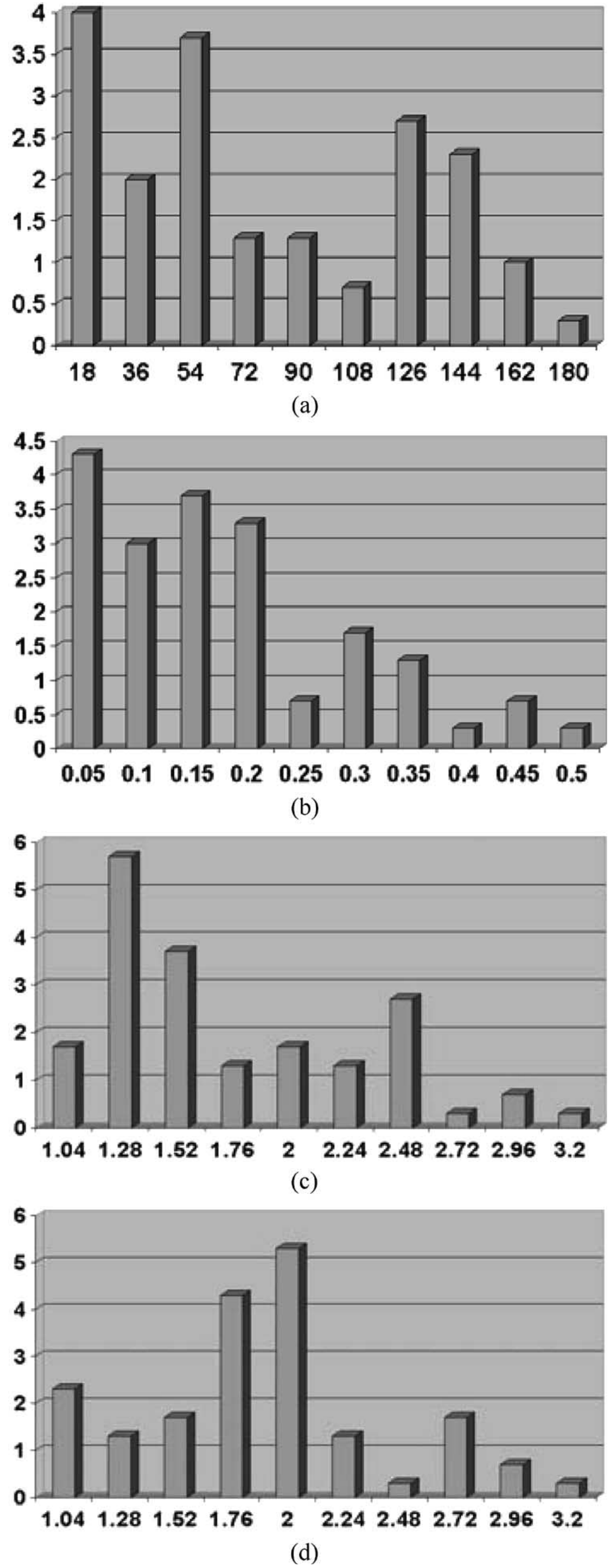
Although the focus of this work is on improving the performance of the verification stage, we would also like to draw attention to several other important issues related to real-time vehicle detection. Fig. 11 shows some representative successful and unsuccessful detection results, including a false negative [Fig. 11(a)], a false positive [Fig. 11(b)], correct detection under minor occlusion [Fig. 11(c)], as well as incorrect detection under severe occlusion [Fig. 11(c)]. The main reason for the false negatives of our system is the lack of sufficient examples, covering all possible vehicle types. On the other hand, the main reason for the false positives is the lack of sufficient nonvehicle examples. It should be mentioned that an effective method for reducing the number of false positives in object detection problems where the nonobject class is much larger than the object class is bootstrapping [65]. In terms of occlusion, the method demonstrates some tolerance since Gabor features are local features. In general, occlusion is not a big issue in the context of on-road vehicle detection since most of the time we are interested in detecting/tracking the nearest vehicle(s) to the host vehicle.

VII. Discussion

To get a better idea of the filter parameters chosen by the EGFO approach, we computed a histogram for each of the parameters (Fig. 12), showing the average distribution of their values over the three data sets. In each graph, the x-axis corresponds to a parameter from $\Phi = \{\theta, W, \sigma_x, \sigma_y\}$ and has been divided into ten bins to compute the histogram. The y-axis corresponds to the average number of Gabor filters whose parameters are within a given interval. For example, Fig. 12(a) shows the average distribution of θ, where the width of each bin is 18°, given $\theta \in [0\ 180°)$. The bar associated with the first bin indicates that there were four filters (average number over the three training data sets) in the optimized Gabor filter set whose orientation parameter satisfies $\theta \in [0\ 18°)$. The only difference for the rest of the parameters is the bin size, for instance, the ith bin in Fig. 12(b) corresponds to the interval $[(i-1)\mathrm{STEP}_W\ \ i\mathrm{STEP}_W)$, where $\mathrm{STEP}_W = (W_{\max} - W_{\min}/10)$.

Several interesting comments can be made based on the experimental results presented in Section VI, the filters shown in Figs. 7–9, and the parameter distributions shown in Fig. 12.

1) The Gabor filters customized using the proposed approach yielded better results in vehicle detection. The most important reason for this improvement is probably that the Gabor filters were designed specifically for the pattern classification problems at hand (i.e., the proposed method is more application specific than existing filter design methods).
2) The orientation parameters of the filters optimized by the GA were tuned to exploit the implicit information available in vehicle data. A Gabor filter is essentially a bar, edge, or grating detector, and will respond most strongly if the filter's orientation is consistent with the orientation of specific features in an image (i.e., bar, edge, etc.). We can see that horizontal, 45°, and 135° structures appear more often in a rear view of a vehicle image, which explains why most of the filter orientations chosen were close to 0°, 45°, and 135° [see Fig. 12(a)].
3) The radial frequency parameters (W) of the filters found by the GA approach were also tuned to encode the implicit information present in vehicle images. Generally speaking, we have more filters with lower radial frequencies than with higher radial frequencies [see Fig. 12(b)]. This is reasonable given that vehicle images contain large structures (windows, bumper, etc.), requiring filters with lower radial frequencies.
4) The parameters σ_x, σ_y were also tuned to respond to the basic structures of a vehicle. Figs. 12(c) and (d) show that the σ_y parameter has bigger values than the σ_x parameter. Bigger σ_y values imply a wider Gaussian mask in the y direction. This is consistent with the observation that horizontal structures in vehicle images spread more widely than structures in vertical direction.
5) The EGFO approach provides a good base for compromising between model compactness and performance accuracy. By setting the threshold factor to 2, we ended up with 14.7 filters on average. The error rate went up to 8.23 from 6.36%, which is still better than using the traditional Gabor filter bank with three scales and five orientations.



(a)

(b)

(c)

(d)

Fig. 12. Distributions of the Gabor filter parameters: (a) θ; (b) W; (c) σ_x; (d) σ_y.

> When we build a pattern classification system, among many factors, we need to find the best balance point between model compactness and performance accuracy. Under some scenarios, we prefer the best performance, no matter what the cost might be. Under different situations, we might favor speed over accuracy, as long as the accuracy is within a satisfactory range.

The EGFO approach is quite computationally demanding. Fortunately, this algorithm is designed to run offline. After the optimization is over, we get a more compact optimized set of Gabor filter, for example, 19 in our experiments. It is this set of optimized filters that will be used to build a real-time vehicle detection system. It should be mentioned that we have already built a real-time vehicle detection system running on Ford Motor Company's concept vehicle [11]. The system runs in an embedded system in the concept vehicle with an average speed of 10 Hz using a traditional Gabor filter bank (24 filters) and SVMs for classification. Therefore, it is safe to say that the optimized set of filters will lead to a real-time vehicle detection system running faster than 10 Hz.

VIII. Conclusion

We have considered the problem of vehicle detection using Gabor filter optimization. In particular, we presented a systematic EGFO approach that yields a more optimal problem-specific set of filters. The EGFO approach unifies filter design with filter selection by integrating GAs with an incremental clustering approach. The resulting filters were evaluated using an application-oriented fitness criterion based on SVMs. Our experimental results show that the set of Gabor filters, specifically optimized for the problem of vehicle detection, yields better performance than using traditional filter banks. The proposed EGFO framework is general and can be applied in other areas requiring filter customization such as face detection. For future work, we plan to evaluate this framework using different data sets and different types of filters. We also plan to test different filter selection schemes by encoding selection in the chromosome explicitly.

References

[1] W. Jones, "Building safer cars," *IEEE Spectrum*, vol. 39, no. 1, pp. 82–85, Jan. 2002.

[2] G. Marola, "Using symmetry for detecting and locating objects in a picture," *Comput. Vis. Graph. Image Process.*, vol. 46, no. 2, pp. 179–195, May 1989.

[3] A. Kuehnle, "Symmetry-based recognition for vehicle rears," *Pattern Recogn. Lett.*, vol. 12, no. 4, pp. 249–258, Apr. 1991.

[4] T. Zielke, M. Brauckmann and W. V. Seelen, "Intensity and edge-based symmetry detection with an application to car-following," *CVGIP, Image Underst.*, vol. 58, no. 2, pp. 177–190, Sep. 1993.

[5] H. Mori and N. Charkai, "Shadow and rhythm as sign patterns of obstacle detection," in *Int. Symp. Industrial Electronics*, Budapest, Hungary, Jun. 1993, pp. 271–277.

[6] E. Dickmanns *et al.*, "The seeing passenger car 'vamors-p'," in *Int. Symp. Intelligent Vehicles*, Paris, France, Oct. 1994, pp. 24–26.

[7] C. Tzomakas and W. Seelen, "Vehicle detection in traffic scenes using shadows," Institut fur neuroinformatik, Ruht-universitat, Bochum, Germany, Tech. Rep. 98-06, 1998.

[8] T. Kalinke, C. Tzomakas and W. V. Seelen, "A texture-based object detection and an adaptive model-based classification," in *IEEE Int. Conf. Intelligent Vehicles*, Stuttgart, Germany, Oct. 1998, pp. 143–148.

[9] N. Matthews, P. An, D. Charnley and C. Harris, "Vehicle detection and recognition in greyscale imagery," *Control Eng. Pract.*, vol. 4, no. 4, pp. 473–479, 1996.

[10] P. Parodi and G. Piccioli, "A feature-based recognition scheme for traffic scenes," in *Proc. IEEE Intelligent Vehicles Symp.*, Detroit, MI, Sep. 1995, pp. 229–234.

[11] Z. Sun, R. Miller, G. Bebis and D. DiMeo, "A real-time precrash vehicle detection system," in *IEEE Int. Workshop Application Computer Vision*, Dec. 2002, pp. 171–176.

[12] Z. Sun, G. Bebis and R. Miller, "Monocular pre-crash vehicle detection: Features and classifiers," *IEEE Trans. Intell. Transp. Syst.*, 2004. (under review).

[13] J. Crisman and C. Thorpe, "Color vision for road following," in *SPIE Conf. Mobile Robots III*, Boston, MA, Nov. 1988, pp. 246–249.

[14] S. D. Buluswar and B. A. Draper, "Color machine vision for autonomous vehicles," *Int. J. Eng. Appl. Artif. Intell.*, vol. 1, no. 2, pp. 245–256, 1998.

[15] D. Guo, T. Fraichard, M. Xie and C. Laugier, "Color modeling by spherical influence field in sensing driving environment," in *IEEE Intelligent Vehicle Symp.*, Dearborn, MI, Oct. 2000, pp. 249–254.

[16] H. Mallot, H. Bulthoff, J. Little and S. Bohrer, "Inverse perspective mapping simplifies optical flow computation and obstacle detection," *Biol. Cybern.*, vol. 64, no. 3, pp. 177–185, 1991.

[17] G. Zhao and S. Yuta, "Obstacle detection by vision system for autonomous vehicle," in *Proc. Intell. Vehicles*, Budapest, Hungary, Jun. 1993, pp. 31–36.

[18] M. Bertozzi and A. Broggi, "Gold: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Trans. Image Process.*, vol. 7, pp. 62–81, Jan. 1998.

[19] A. Broggi, M. Bertozzi, A. Fascioli, C. Guarino Lo Bianco and A. Piazzi, "Visual perception of obstacles and vehicles for platooning," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, pp. 164–176, Sep. 2000.

[20] A. Giachetti, M. Campani and V. Torre, "The use of optical flow for road navigation," *IEEE Trans. Robot. Autom.*, vol. 14, pp. 34–48, Feb. 1998.

[21] W. Kruger, W. Enkelmann and S. Rossle, "Real-time estimation and tracking of optical flow vectors for obstacle detection," in *Proc. IEEE Intelligent Vehicle Symp.*, Detroit, MI, Sep. 1995, pp. 304–309.

[22] J. Weng, N. Ahuja and T. Huang, "Matching two perspective views," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, pp. 806–825, Aug. 1992.

[23] D. Koller, N. Heinze and H. Nagel, "Algorithmic characterization of vehicle trajectories from image sequence by motion verbs," in *IEEE Int. Conf. Computer Vision Pattern Recognition*, Maui, HI, Jun. 1991, pp. 90–95.

[24] M. Betke, E. Haritaglu and L. Davis, "Multiple vehicle detection and tracking in hard real time," in *Proc. IEEE Intelligent Vehicles Symp.*, Tokyo, Japan, Sep. 1996, pp. 351–356.

[25] J. Ferryman, A. Worrall, G. Sullivan and K. Baker, "A generic deformable model for vehicle recognition," in *Proc. British Machine Vision Conf.*, Birmingham, U.K., Sep. 1995, pp. 127–136.

[26] C. Goerick, N. Detlev and M. Werner, "Artificial neural networks in real-time car detection and tracking applications," *Pattern Recogn. Lett.*, vol. 17, no. 4, pp. 335–343, 1996.

[27] H. Schneiderman and T. Kanade, "Probabilistic modeling of local appearance and spatial relationships for object recognition," in *Proc. IEEE Int. Conf. Computer Vision Pattern Recognition*, Santa Barbara, CA, Jun. 1998, pp. 45–51.

[28] H. Schneiderman, *A Statistical Approach to 3D Object Detection Applied to Faces and Cars*, 2000. CMU-RI-TR-00-06.

[29] M. Weber, M. Welling and P. Perona, "Unsupervised learning of models for recognition," in *Proc. European Conf. Computer Vision*, Dublin, Ireland, 2000, vol. 1, pp. 18–32.

[30] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *Int. J. Comput. Vis.*, vol. 38, no. 1, pp. 15–33, 2000.

[31] Z. Sun, G. Bebis and R. Miller, "On-road vehicle detection using Gabor filters and support vector machines," in *IEEE 14th Int. Conf. Digital Signal Processing*, Greece, Jul. 2002, pp. 1019–1022.

[32] ——, "Improving the performance of on-road vehicle detection by combining Gabor and wavelet features," in *IEEE 5th Int. Conf. Intelligent Transportation Systems*, Singapore, Sep. 2002, pp. 130–135.

[33] T. Kato, Y. Ninomiya and I. Masaki, "Preceding vehicle recognition based on learning from sample images," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, pp. 252–260, Dec. 2002.

[34] S. Watanabe, *Pattern Recognition: Human and Mechanical*. New York: Wiley, 1985.

[35] Z. Sun, G. Bebis and R. Miller, "Quantized wavelet features and support vector machines for on-road vehicle detection," in *7th Int. Conf. Control, Automation, Robotics Vision*, Singapore, Dec. 2002, pp. 1641–1646.

[36] J. Daugman, "Complete discrete 2-D Gabor transforms by neural network for image analysis and compression," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 36, pp. 1169–1179, Jul. 1988.

[37] R. Mehrotra, K. Namuduri and N. Ranganathan, "Gabor filter-based edge detection," *Pattern Recogn.*, vol. 25, no. 12, pp. 1479–1493, 1992.

[38] T. Weldon, W. Higgins and D. Dunn, "Efficient Gabor filter design for texture segmentation," *Pattern Recogn.*, vol. 29, no. 12, pp. 2005–2015, 1996.

[39] A. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recogn.*, vol. 23, no. 12, pp. 1167–1186, 1991.

[40] T. Hofmann, J. Puzicha and J. Buhmann, "Unsupervised texture segmentation in a deterministic annealing framework," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, pp. 803–818, May 1998.

[41] Y. Hamamoto, S. Uchimura, M. Watanabe, T. Yasuda, Y. Mitani and S. Tomota, "A Gabor filter-based method for recognizing handwritten numerals," *Pattern Recogn.*, vol. 31, no. 4, pp. 395–400, 1998.

[42] K. Chung, S. Kee and S. Kim, "Face recognition using independent component analysis of Gabor filter responses," in *IAPR Workshop Machine Vision Applications*, Tokyo, Japan, Nov. 2000, pp. 331–334.

[43] B. Manjunath and W. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, pp. 837–842, Aug. 1996.

[44] T. Weldon, W. Higgins and D. Dunn, "Gabor filter design for multiple texture segmentation," *Opt. Eng.*, vol. 35, no. 10, pp. 2852–2863, Oct. 1996.

[45] A. Bovik, M. Clark and W. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, pp. 55–73, Jan. 1990.

[46] B. Okombi-Diba, J. Miyamichi and K. Shoji, "Edge-based segmentation of textured images using optimally selected Gabor filters," in *IAPR Workshop Machine Vision Applications*, Tokyo, Japan, Nov. 2000, pp. 267–270.

[47] D. Dunn and W. Higgins, "Optimal Gabor filters for texture segmentation," *IEEE Trans. Image Process.*, vol. 4, pp. 947–964, Jul. 1995.

[48] M. Turner, "Texture discrimination by Gabor functions," *Biol. Cybern.*, vol. 55, no. 2–3, pp. 71–82, Nov. 1986.

[49] A. Katz and P. Thrift, "Generating image filters for target recognition by genetic learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, pp. 906–910, Sep. 1994.

[50] G. Bebis, S. Louis, Y. Varol and A. Yfantis, "Genetic object recognition using combinations of views," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 132–146, Apr. 2002.

[51] D. Swets, B. Punch and J. Weng, "Genetic algorithms for object recognition in a complex scene," in *IEEE Int. Conf. Image Processing*, Washington, DC, Oct. 1995, pp. 595–598.

[52] C. Liu and H. Wechsler, "Evolutionary pursuit and its application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 570–582, Jun. 2000.

[53] G. Bebis, S. Uthiram and M. Georgiopoulos, "Face detection and verification using genetic search," *Int. J. Artif. Intell. Tools*, vol. 9, no. 2, pp. 225–246, 2000.

[54] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[55] M. Srinivas and L. Patnaik, "Genetic algorithms: A survey," *IEEE Comput.*, vol. 27, no. 6, pp. 17–26, Jul. 1994.

[56] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Upper Saddle River, NJ: Prentice-Hall, 1998.

[57] A. Jain, M. Murty and P. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 265–323, 1999.

[58] L. Eshelman, "The chc adaptive search algorithm: How to have safe search when engaging in non-traditional genetic recombination," in *Foundation Genetic Algorithms*, G. J. E. Rawlings, Ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 265–283.

[59] P. Kuizinga, N. Petkov and S. Grigorescu, "Comparison of texture features based on Gabor filters," in *Proc. 10th Int. Conf. Image Analysis Processing*, Venice, Italy, Sep. 1999, pp. 142–147.

[60] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[61] C. Burges, "Tutorial on support vector machines for pattern recognition," *Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 955–974, 1998.

[62] Z. Sun, X. Yuan, G. Bebis and S. Louis, "Neural-network-based gender classification using genetic eigen-feature extraction," in *IEEE Int. Joint Conf. Neural Networks*, Honolulu, HI, May 2002, vol. 3, pp. 2433–2438.

[63] Z. Sun, G. Bebis, X. Yuan and S. Louis, "Genetic feature subset selection for gender classification: A comparison study," in *IEEE Int. Workshop Application Computer Vision*, Orlando, FL, Dec. 2002, pp. 165–170.

[64] S. Weiss and C. Kulikowski, *Computer Systems That Learn*. San Mateo, CA: Morgan Kaufmann, 1991.

[65] H. Rowley, S. Baluja and T. Kanade, "Neural network-based face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, pp. 22–38, Jan. 1998.

Zehang Sun received the B.Eng. degree in telecommunication and the M.Eng. degree in digital signal processing from Northern Jiaotong University, Beijing, China, in 1994 and 1997, respectively, the M.Eng. degree in electrical and electronic engineering from Nanyang Technological University, Singapore, in 1999, and the Ph.D. degree in computer science and engineering from the University of Nevada, Reno, in 2003.

He joined eTreppid Technologies, LLC, Reno, NV, immediately upon his graduation. His expertise is in the area of real-time computer vision systems, statistical pattern recognition, artificial intelligence, digital signal processing, and embedded systems.

George Bebis received the B.S. degree in mathematics and the M.S. degree in computer science from the University of Crete, Greece, in 1987 and 1991, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Central Florida, Orlando, in 1996.

He is currently an Associate Professor at the Department of Computer Science and Engineering, University of Nevada, Reno (UNR), and the Director of the UNR Computer Vision Laboratory (CVL). He is an Associate Editor of the *Machine Vision and Applications Journal*, and serves on the Editorial Board of the *Pattern Recognition Journal* and the *International Journal on Artificial Intelligence Tools*. He has served on the program committees of various national and international conferences and has organized and chaired several conference sessions. His research interests include computer vision, image processing, pattern recognition, machine learning, and evolutionary computing. His research is currently funded by NSF, NASA, ONR, and Ford Motor Company.

He is a Member of the IEEE and the IAPR Educational Committee. In 2002, he received the Lemelson Award for Innovation and Entrepreneurship.

Ronald Miller received the B.S. degree in physics from the University of Massachusetts, Boston, in 1983 and the Ph.D. degree in physics from the Massachusetts Institute of Technology, Cambridge, in 1988.

He heads a research program at Ford Motor Company, Dearborn, MI, in intelligent vehicle technologies focusing on advanced RF communication, radar, and optical sensing systems for accident avoidance and telematics. His research has ranged from computational modeling of plasma and ionospheric instabilities to automotive safety applications.