Front Vehicle Recognition Based on Automotive Vision

Fei Huang

School of Mechanical and Electronic Engineering, Suzhou University, Suzhou 23400, Anhui, China

Abstract
This paper proposes an algorithm based on Haar-like features and improved Adaboost classifier for vehicle recognition. The theoretical contribution of this study includes two aspects: (1) combine Haar-like features with the corresponding class labels while training a weak classifier, which increases the efficiency of training. (2) introduce an adaptive threshold algorithm for classification, which improves the recognition capability. We apply the method to a road vehicle recognition system based on automotive vision. The experimental results demonstrate that the method we propose performs better than traditional methods in the time consumed on training and the recognition capability.

Key words: Vehicle Recognition, Haar-like Features, Adaboost Algorithm.

1. INTRODUCTION

According to the statistics of the traffic accidents, vehicle collisions make up a big proportion of the causes of traffic accidents, especially the vehicle in front is a main threat to a driver. In order to decrease the accident rate and personal property losses, many automobile manufacture and research institutes dedicate to the driver assistance system or collision avoidance system based on intelligent vehicle. The driver assistance system could alert driver the driving environment information about the front or dangerous situation, reducing the probability of vehicle collision. To achieve this, the primary task is to recognize and locate the front vehicle using sensor. Now the commonly used sensors include laser, sonar, infrared and vision. Compared to others, vision has the advantage of acquiring more information.

Vehicle recognition based vision falls into several categories: feature-based, based on optical flow field, model-based and based on machine learning. Feature-based recognition method takes advantage of vehicle features to recognize vehicle e.g. shadow, edges and symmetry. For more accurate, usually use fusion features containing above-mentioned features. Machine learning uses the rules or patterns which extract from the training samples to translate the data into information, helping recognizing vehicle. Till now, machine learning combined with features has attracted more and more attention, for the reason of good performance of robustness and recognition capability. Nospide proposed a vehicle recognition method based on SVM classification extracting features with PCA (Nospide and Salgado, 2012). Sun introduced a kind of SVM classification combined with Gabor features or Haar features to recognize vehicle (Sun and Bebis, 2002). Both these machine learning methods require computationally expensive features extraction process, and have the defect of the inadequate description capacity of features. Meanwhile the parameters selection process of SVM based on radial-basis function is complicated and easily getting into local minimum. Another machine learning method based on Adaboost have been successfully used in face recognition application, and have being introduced into vehicle recognition field (Viola and Jones, 2001; Viola and Jones, 2004; Lienhart and Maydt, 2002). STOJMENOVIC proposed a vehicle recognition method based on Adaboost classifier combined with edge and color features (Stojmenovic, 2005). Chang introduced a method of training cascade classifier for recognition (Wen and Chi-Wei, 2010). Experiment results indicate that Adaboost method can satisfy the requirement of high-level real-time with high recognition rate and low false rate. But when the dimension of training sample is extremely high, the time consumed on training takes too long.

Using for reference from the application of Haar-like features and Adaboost classifier on face recognition, this paper propose a vehicle recognition algorithm based on improved Adaboost algorithm combined with Haar-like features. Experiment results demonstrate that compared to other proposed methods, our approach has better recognition capability, less false recognition rate and less training time.

2. ALGORITHM STRUCTURE

The algorithm mainly consists of two steps called training and recognition (Freund and Schapire, 1997). In training part, collect a large number of vehicle images as positive samples and non-vehicle images easily mistaken for vehicles as negative samples. Then, extract Haar-like features which are suitable to describe vehicle appearance for training Adaboost classifier (Wang and Shen, 2011). The features extraction and
selection process is critical for that the selected features should match with vehicle appearance as possible as it can, which determines the accuracy of the whole algorithm.

In recognition part, firstly locate the area where vehicle may exist, i.e., ROI (region of interest). Secondly traversal the ROI with multi-scale sliding windows to extract the Haar-like features, then input the feature vectors from ROI into the trained classifier. The classifier evaluates a result for the input vectors and feature vectors stored in classifier, if positive, merge searching windows and precisely locate the vehicle position. Figure 1 shows algorithm flowchart of training and recognition.

3. OFFLINE TRAINING

3.1. Image Pre-processing

The training samples include positive samples and negative samples. We select rear vehicle images under several conditions as positive samples, e.g. different vehicle models, different views, different distance, etc. Negative samples are arbitrary non-vehicle images. This paper selects 10000 positive samples and 5000 negative samples. To avoid distinction brought by different pixel sizes of images, before using the images, gray and normalize the sample images to size 30×30, finally form the training set with these normalized gray image. Figure 2 and 3 show the training samples.
3.2. Haar-like Features

Haar-like features is introduced by Viola firstly applied to face recognition, it is a kind of rectangle feature usually used to describe local gray scale variation. On the basis of horizontal rectangle Haar-like features and vertical rectangle Haar-like features, Lienhart introduced a rectangle feature at a 45 degree angle and a fast calculation method of this feature (Lienhart and Kuranov, 2003). This algorithm improves the recognition capability of system without affecting speed of recognition (Lienhart and Maydt, 2002). This paper selects 12 kinds of Haar - like rectangle feature templates to describe edge and structure features of vehicle, including 11 kinds from Lienhart proposed extension libraries, and a single rectangle feature proposed by this paper to represent shadow at the bottom of vehicle. Figure 4 shows 12 kinds of Haar-like features.

![Figure 4. The 12 kinds Haar-like features](image)

Under normal light conditions, relative to the driving environment, vehicles exist apparent shadow features and density differences, e.g. shadow at the bottom of vehicle, density differences between the top of vehicle and the sky, etc. These form primary rectangle features which could represent vehicle well. Figure 5 shows Haar-like features represent vehicle.

![Figure 5. Haar-like features represent vehicle](image)

3.3. Integral Image

In the image pre-processing part, we gray and normalize all the training samples including positive and negative ones. Then traverse these gray images of size 30×30 and calculate the integral images. It contains a huge quantity of Haar-like rectangle features in a gray images of size 30×30, for a real-time system, extracting and calculating these features takes too much time while evaluating feature value. So integral image is usually...
applied to calculating the features values associated with every area in gray image. In recognition process, integral image is also used to calculate input images for the purpose of improving real-time.

3.4. Classifier Based on Improved Adaboost Algorithm

The number of Haar-like features extracted from a single image of size 30x30 is more than 50000, it is a huge computation to apply all the features to vehicle recognition, even these features can be calculated fast. Selecting critical features from 50000 features for recognition is significant, and Adaboost algorithm is an effective method of selecting key features for vehicle classifier (Collins and Schapire, 2002).

Freund and Schapire proposed Adaboost (adaptive boosting) algorithm that does not need prior knowledge, which is significant for practical problem-solving(Freund and Schapire, 1996). We first start with a review of Adaboost algorithm for classification. Given labeled training sample set \( \{ (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n) \} \), with \( y_i \in \{-1, +1\}, \) \( x_i \) denotes the training samples, \( y_i \) denotes the class of samples. \( y_i = -1 \) indicates non-vehicle sample image, \( y_i = +1 \) indicates vehicle sample image.

1. Initialize the weight of sample \( x_i \)
\[
w_{i,t} = 1/n
\]

Where \( w_{i,t} \) denotes the weight associated with sample \( x_i \) in Round \( t \).

2. For each feature, train a weak classifier \( f_{i,j} \)
\[
f_{i,j} = \begin{cases} 1, & p_j h_j(x) < p_j \theta_j \\ -1, & \text{otherwise} \end{cases}
\]

Where \( h_j(x) \) denotes the value of the \( j \)’th feature in sample \( x \), \( p_j \) denotes classes, \( \theta_j \) denotes the threshold of the weak classifier. Calculate the weighted classification error for each weak classifier
\[
\epsilon_{i,j} = \sum_{i=1}^{n} w_{i,t} = |f_{i,j} - y_i|
\]

3. Find the weak classifier contributes the least classification error this round
\[
\epsilon_t = \min(\epsilon_{i,j})
\]

4. Update the weight associated with each sample. The samples which are classified incorrectly have their weight increased, otherwise decreased.

The aim of Adaboost algorithm is to find the classifier contributes the least to the overall classification error(Li and Shen, 2008), i.e. find the best parameters \( \{h, p_j, \theta_j\} \). Traditional Adaboost algorithm uses exhaustion method on all the Haar-like features extracted from training samples set, to generate corresponding classifier, then select the classifier contributes the least classification error as weak classifier (Schwenk and Bengio, 1998). Exhaustion method leads to huge computation and long training time, for improving the disadvantage, we propose an efficient Adaboost algorithm to generate weak classifier, combining the Haar-like features with the corresponding class label, and propose a new method of setting classifier threshold (Ge and Luo, 2009; Masnadi-Shirazi and Vasconcelos, 2011).

Given Matrix A with all the Harr-like features extracted from training samples set.
\[
A = \begin{pmatrix}
\alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\
\alpha_{21} & \alpha_{22} & \cdots & \alpha_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mn}
\end{pmatrix}
\]

Where \( n \) denotes the number of samples in the training sample set, \( m \) denotes the number of Harr-like features extracted from a normalized gray image of size 30x30, \( \alpha_{ij} (i \in \{1, 2, \ldots, m\}, j \in \{1, 2, \ldots, n\}) \) denotes the value of the \( i \)’th Haar-like feature in the \( j \)’th sample image.
E.g. training a weak classifier using the $i$’th Haar-like feature in the $i$’th iteration (Lee and Yuille, 2011). Given Vector $\text{Vec}$ consisted of the corresponding feature values, the weight associated with sample $x_i$ is $w_i$, the type of class is $y_i \in \{-1, +1\}$. $\text{Vec}[i]$ indicates the value of Haar-like feature in sample $x_i$.

The process of the improved algorithm generating a weak classifier is as follows:

Begin
(1) For $i = 1$ To $m$
   a. Sort the features vector: let $\text{VectSort}$ denote the sorted values of $\text{Vec}$ in increasing order with corresponding labels as $\text{Lab}$.
   b. Generate the candidate classification positions set: scan $\text{Lab}$ to find the positions of a pair which are different, check whether their corresponding feature values are same. If not, put the first of the pair into the candidate classification positions set. If same, from right to left find the first feature value that is different from the previous ones, check whether this one has been included in the candidate classification positions set, if not, put it into the set; Then from left to right follow the process as above-mentioned, repeat this step until the end of $\text{Lab}$. Finally we get the candidate classification positions set $L = \{l_1, l_2, \cdots, l_k\}$.
   c. Generate the classification position of the $i$’th Haar-like feature: select a classifier position contributes the least to the overall classification error from the classification positions set $L = \{l_1, l_2, \cdots, l_k\}$.
   d. Set classification threshold based on the classification position, acquire the classification rule of the $i$’th Haar-like feature.
(2) Select the rule contributes the least to the overall classification error from $m$ kinds of rules to be the weak classifier as this iteration.
(3) Output the weak classifier.
End

For further interpretation, the process of generating the classification position of the $i$’th Haar-like feature is as follows: for describing conveniently, $\forall l_i \in L$. The prediction for the samples corresponding to $l_i$ and $l_j$ are same, i.e. $\lambda \in \{-1, +1\}$, and the prediction for the samples corresponding to $l_{i+1}, \cdots, l_k$ is $-\lambda$. We rewrite the calculation formula of classification error:

$$\varepsilon_i = \frac{1}{4} \sum_{j=1}^{n} w_j \left( f_j - y_j \right)^2$$

Where $j$ indicates the index of element in feature vector $\text{VectSort}$, $n$ indicates the number of samples, $w_j$ denotes the weight associated with the $j$’th sample, $f_j \in \{-1, +1\}$ denotes the prediction class of the $j$’th sample, $y_i \in \{-1, +1\}$ denotes the class label of the $j$’th sample. We infer:

$$\varepsilon_i = \frac{1}{4} \left( \sum_{j=1}^{n} w_{j} \left( f_j - y_j \right)^2 \right)$$

$$= \frac{1}{4} \left( \sum_{j=1}^{l_i} w_{j} \left( \lambda - y_j \right)^2 + \sum_{j=l_i+1}^{n} w_{j} \left( -\lambda - y_j \right)^2 \right)$$

$$= \frac{1}{4} \sum_{j=1}^{n} w_{j} \left( \lambda^2 + y_j^2 \right) + \frac{1}{2} \lambda \left( \sum_{j=1}^{l_i} w_{j} y_j - \sum_{j=l_i+1}^{n} w_{j} y_j \right)$$

As $w_j$ and $y_j$ are known, also $\sum w_j y_j$ is known. Cause

$$\sum_{j=1}^{n} w_{j} y_j = \sum_{j=1}^{l_i} w_{j} y_j + \sum_{j=l_i+1}^{n} w_{j} y_j$$

and $\lambda^2 = y_j^2 = 1$, $\sum_{i=1}^{n} w_j = 1$, we can see:
\[
\varepsilon_i = \frac{1}{4} \sum_{j=1}^{n} w_j (\lambda^2 + y_j^2) + \frac{1}{2} \lambda \left[ \sum_{j=1}^{n} w_j y_j - 2 \sum_{j=1}^{l} w_j y_j \right]
\]

\[
= \frac{1}{2} + \frac{1}{2} \lambda \left[ \sum_{j=1}^{n} w_j y_j - 2 \sum_{j=1}^{l} w_j y_j \right]
\]

Next we discuss the cases when the value of \( \lambda \) differs:

(a) When \( \lambda = 1 \), equation 9 changes

\[
\varepsilon_i = \frac{1}{2} + \frac{1}{2} \lambda \left[ \sum_{j=1}^{n} w_j y_j - 2 \sum_{j=1}^{l} w_j y_j \right]
\]

\[
\min(\varepsilon_i) \text{ means } \max \left( \sum_{j=1}^{l} w_j y_j \right), \text{ with } w_j > 0, \text{ so only when } y_i = 1 \text{ and } y_{i,u} = -1. \sum_{j=1}^{l} w_j y_j \text{ maximum.}
\]

Find some classification positions conform with above conditions from the classification positions set

\[
L = \{l_1, l_2, \cdots, l_k\}.
\]

Supposing that there are \( p \) positions, \( \tau \in \{s_1, s_2, \cdots, s_p\} \subseteq L \), we define

\[
e_i = \sum_{j=1}^{n} w_j y_j
\]

\[
e_r = e_{r-1} + \sum_{j=r-1}^{l} w_j y_j, r = 2, 3, \cdots, p \max \left( \sum_{j=1}^{l} w_j y_j \right) = \max \left\{ e_1, e_2, \cdots, e_p \right\}
\]

(b) When \( \lambda = -1 \), equation 9 changes

\[
\varepsilon_i = \frac{1}{2} - \frac{1}{2} \lambda \left[ \sum_{j=1}^{n} w_j y_j - 2 \sum_{j=1}^{l} w_j y_j \right]
\]

\[
\min(\varepsilon_i) \text{ means } \min \left( \sum_{j=1}^{l} w_j y_j \right), \text{ with } w_j > 0, \text{ so only when } y_i = -1 \text{ and } y_{i,u} = 1. \sum_{j=1}^{l} w_j y_j \text{ minimum.}
\]

Find some classification positions conform with above conditions from the classification positions set

\[
L = \{l_1, l_2, \cdots, l_k\}.
\]

Supposing that there are \( p \) positions, \( \delta \in \{t_1, t_2, \cdots, t_p\} \subseteq L \), the next process is similar to step 11 and step 12, we finally get

\[
\min \left( \sum_{j=1}^{l} w_j y_j \right) = \min \left\{ t_1, t_2, \cdots, t_p \right\}
\]

Given the minimum error \( \varepsilon_i \) with \( \lambda = 1 \), \( e_{i-1} \) with \( \lambda = -1 \), the classification symbol \( p_i \) and the classification error \( \Omega_i \) corresponding to the \( i \)’th Haar-like feature for the Round \( t \) iteration are as follows:

\[
\Omega_i = \min (\varepsilon_i, e_{i-1})
\]

\[
p_i = \begin{cases} 
1, & \lambda = 1 \\
-1, & \lambda = -1 
\end{cases}
\]

Where we define the classification position contributes the least to the overall classification error as \( \eta \in L \), next we calculate the classification threshold using traditional mean algorithm:

\[
\theta_i = \frac{\text{SortVector}[\eta] + \text{SortVector}[\min\{(\eta+1), n\}]}{2}
\]
However, this method does not reflect the distribution rule of the training sample. Here we propose an adaptive classification threshold algorithm: when the value of a Haar-like feature is equal or lesser than \(SortVec[\eta]\), the output class is \(Label(\text{Label} \in \{+1,-1\})\), when the value is equal or greater than \(SortVec[\eta]\), the output is \(-Label\). The prior probability \(P_i\) for that the class label with the feature value equal or lesser than \(SortVec[\eta]\) is \(Label\):

\[
P_i[\text{Label} | \text{SortVec}[k] \leq \text{SortVec}[\eta]] = \frac{\sum_{y_k = \text{Label}} w_k}{\sum_{k \leq q} w_k}
\]

The prior probability \(P_2\) for that the class label with the feature value equal or greater than \(SortVec[\eta + 1]\) is \(-Label\):

\[
P_2[-\text{Label} | \text{SortVec}[k] \geq \text{SortVec}[\eta + 1]] = \frac{\sum_{y_k = -\text{Label}} w_k}{\sum_{k > q + 1} w_k}
\]

Where \(w_k\) denotes the weight associated with the training sample indexed \(k\), \(y_k\) denotes the true classification label of the training sample indexed \(k\).

The above two sorts of probability values reflect the distribution rule of two different training samples. Based on the distribution rule, we set the classification threshold as:

\[
\theta_t = \text{SortVec}[\eta] + \frac{P_1 * \text{SortVec}[\eta] - \text{SortVec}[\eta + 1]}{P_1 + P_2}
\]

Through the above-mentioned algorithm, we can acquire the classification direction, error and threshold for all the Haar-like features in Round \(t\) iteration, further acquiring the minimum classification error, the corresponding classification direction, the classification threshold and the corresponding Harr-like feature. In the end, follow the definition of Adaboost classifier algorithm, combining the weak classifiers generated in every iteration round for a strong classifier, then apply the strong classifier to the subsequent vehicle recognition.

### 4. EXPERIMENT RESULTS

On-line recognition process is applied to detecting the existence of vehicle in some area, including image pre-processing, Haar-like features extraction, integral image calculating and Adaboost classifier recognition (Shen and Hao, 2011). The classifier evaluates a result for the feature vectors extracted from test images and feature vectors stored in classifier (Warmuth and Glocer, 2007). Based on the classifier result, we can detect whether do vehicles exist in the test image and locate the vehicles.

For testing the effectiveness and feasibility of our method, we apply the method to a road vehicle recognition system based on monocular vision, using Visual Studio 2008 with OpenCV2.4.2 library as an experiment platform.

We test 1713 relevant vehicle objects from videos recorded in driving cars, 1665 vehicles successfully recognized, the recognition rate is 97.20%, 81 non-vehicle objects misrecognized, the false positive rate is 4.72%. Table 1 shows the specific statistical result.

**Table 1. Vehicle recognition result**

| Vehicle Model | Vehicle Number | Recognition Number | Recognition Rate | Misrecognition Number | False Positive Rate |
|---------------|----------------|--------------------|------------------|-----------------------|---------------------|
| car           | 1156           | 1135               | 98.18%           | -                     | -                   |
| minibus       | 177            | 172                | 97.17%           | -                     | -                   |
| bus           | 245            | 228                | 93.06%           | -                     | -                   |
| truck         | 135            | 130                | 96.30%           | -                     | -                   |
| Total         | 1713           | 1665               | 97.20%           | 81                    | 4.72%               |

We compare our method with the commonly used vehicle recognition methods. Table 2 shows the result of recognition rate comparison. And Compared with traditional methods, the adaptive classification threshold algorithm we propose saves 10% of the training time.
Table 2. Recognition algorithm comparison

| Recognition Algorithm       | Recognition Rate |
|----------------------------|------------------|
| PCA+SVM                    | 95.67%           |
| Gabor+SVM                  | 95.04%           |
| Wavelet+SVM                | 94.78%           |
| Haar-like Feature +Cascade Adaboost | 97.10%     |
| **Our algorithm**          | **97.20%**       |

Figure 6 shows that our algorithm has a good performance on recognizing the front vehicles with different views and distance.

**Figure 6.** Real-time recognition result

5. CONCLUSION

In this paper, we show vehicle recognition method based on Haar-like features and an improved Adaboost algorithm. The theoretical contribution to Adaboost algorithm is two improved method: (1) combine the Haar-like features with the corresponding class labels while training a weak classifier, it increases the efficiency of training. (2) propose an adaptive classification threshold algorithm. The experimental results demonstrate that the method we propose performs better than traditional methods in the time consumed on training and the recognition capability.

Acknowledgements

This work was supported by 2014YKF37. (Scientific Research Platform Project of Suzhou University)

REFERENCE

Collins M, Schapire R E, Singer Y. (2002) “Logistic regression, AdaBoost and Bregman distances”, *Machine Learning*, 48(1-3), pp. 253-285.

Freund Y, Schapire R E. (1997) “A decision-theoretic generalization of on-line learning and an application to Boosting”, *Journal of Computer and System Sciences*, 55(1), pp. 119-139.

Freund Y, Schapire R E. (1996) “Experiments with a New Boosting Algorithm”, *Proceedings of the 13th Conference on Machine Learning*, pp. 148-156.

Ge J F, Luo Y P. (2009) “A comprehensive study for asymmetric AdaBoost and its application in object detection”, *Acta Automatica Sinica*, 35(11), pp. 1403-1409.

Lee J J, Lee P H, Lee S W, Yuille A, Koch C. (2011) “AdaBoost for text detection in natural scene”, *Proceedings of the 11th International Conference on Document Analysis and Recognition*, pp. 429-434.

Lienhart R, Kuranov A, Pisarevsky V. (2003) “Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection”, *Proc.25th German Pattern Recognition Symposium. Magdeburg*, pp. 297-304.

Lienhart R, Maydt J. (2002) “An Extended Set of Haar-like Features for Rapid Object Detection”, *Proc.of the IEEE Conf.on Image Processing*, pp. 155 -162.

Li H X, Shen C H. (2008) “Boosting the minimum margin: LPBoost vs AdaBoost”, *In: Proceedings of the 2008 Digital Image Computing: Techniques and Applications*, pp. 533-539.
Masnadi-Shirazi H, Vasconcelos N. (2011) “Cost-sensitive Boosting”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2), pp. 294-309.

Nospide J, Salgado L. (2012) “Region-dependent vehicle classification using PCA features”, *IEEE International Conference on Image Processing*, pp. 453-456.

Schwenk H, Bengio Y. (1998) “Training methods for adaptive Boosting of neural networks for character recognition”, *Advances in Neural Information Processing Systems*, 46(13), pp. 647-653.

Shen C H, Hao Z H. (2011) “A direct formulation for totally corrective multi-class Boosting”, *Proceedings of the 2011 IEEE Conference on Pattern Recognition*, pp. 647-653.

Schwenk H, Bengio Y. (1998) “Training methods for adaptive Boosting of neural networks for character recognition”, *Advances in Neural Information Processing Systems*, 46(13), pp. 647-653.

Shen C H, Hao Z H. (2011) “A direct formulation for totally corrective multi-class Boosting”, *Proceedings of the 2011 IEEE Conference on Pattern Recognition*, pp. 647-653.

Stojmenovic M. (2005) “Real time object detection in images based on an AdaBoost machine learning approach and a small training set”, *Proceedings of 12th Int Workshop on Systems, Signals & Image Processing*, pp. 117-122.

Sun Z, Bebis G, Miller R. (2002) “On road vehicle detection using Gabor filters and support vector machines”, *IEEE 14th International Conference on Digital Signal Processing*, pp. 1019-1022.

Viola P, Jones M. (2001) “Rapid object detection using a boosted cascade of simple features”, *In Proceeding of International Conference on Computer Vision and Pattern Recognition*, pp. 511-518.

Viola P, Jones M. (2004) “Robust real-time face detection”, *International Journal of Computer Vision*, 57(2), pp. 137-154.

Wang P, Shen C H, Barnes N, Zheng H, Ren Z. (2011) “Asymmetric totally-corrective Boosting for real-time object detection”, *In: Proceedings of the 10th Asian conference on Computer Vision*, pp. 176-188.

Warmuth M K, Glocer K A. (2007) “Boosting algorithms for maximizing the soft margin”, *Advances in Neural Information Processing Systems*, pp. 1585-1592.

Wen Chung Chang, Chi-Wei Cho. (2010) “Online boosting for vehicle detection”, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 40(3), pp. 892-902.