2012 15th International IEEE Conference on Intelligent Transportation Systems
Anchorage, Alaska, USA, September 16-19, 2012

Vehicle Detection, Tracking and Classification in Urban Traffic

Zezhi Chen, Tim Ellis, Sergio A Velastin SMIEEE

Abstract— **This paper presents a system for vehicle detection, tracking and classification from roadside CCTV. The system counts vehicles and separates them into four categories: car, van, bus and motorcycle (including bicycles). A new background Gaussian Mixture Model (GMM) and shadow removal method have been used to deal with sudden illumination changes and camera vibration. A Kalman filter tracks a vehicle to enable classification by majority voting over several consecutive frames, and a level set method has been used to refine the foreground blob. Extensive experiments with real world data have been undertaken to evaluate system performance. The best performance results from training a SVM (Support Vector Machine) using a combination of a vehicle silhouette and intensity-based pyramid HOG features extracted following background subtraction, classifying foreground blobs with majority voting. The evaluation results from the videos are encouraging: for a detection rate of 96.39%, the false positive rate is only 1.36% and false negative rate 4.97%. Even including challenging weather conditions, classification accuracy is 94.69%.**

I. Introduction

Applying image processing technologies to vehicle detection and classification has been a hot focus of research in Intelligent Transportation Systems (ITS) over the last decade. Urban traffic flow analysis is important for traffic management, but is a challenging problem under high vehicle densities which can result in frequent occlusion. Several problems have to be solved, ranging from low and middle level vision tasks, such as the detection and tracking of multiple moving objects in a scene, to high level analyses, like vehicle classification [1]. Vehicle classification is particularly useful for re-identification in multi-sensor networks and anomalous event detection, as well as the more standard applications of traffic flow analysis and unobtrusive path tracing [2-5]. In [6], we presented a comparison of methods for categorising vehicle types into a set of classes using two different approaches, based on features derived from their image silhouettes. In those experiments, manual segmentation was used to delineate vehicles in the images and a set of scaled features were extracted from each binary silhouette. Results were presented for a 10-fold cross-validation study involving over 2000 manually labeled silhouettes. A peak classification performance of 96.26% was observed for SVM. This paper presents an Automatic Vehicle Detection and Classification System (AutoVDCS) which includes (in addition to classification) automatic detection of vehicles. The approach has been tested on real video footage taken during daylight hours from cameras mounted on road-side poles in the town of Kingston upon Thames, UK. The AutoVDCS is able to detect vehicles as they move through the detection area of the camera's field of view, track them and classify each individual object into five main categories: motorcycle (motorcycle and bicycle), car (car and taxi), van (van, minivan, minibus and limousine), bus (single and double decked) and unknown. Counts are then produced for each category. The aim is to collect traffic census data for statistical analysis. This paper demonstrates the effectiveness of combining tracking with classification for significantly improved classification results. The paper is organized following the system's structure, as described in more detail in the next section. Figure 1 shows the output of the system applied to analysis of a three-lane road, with the associated vehicle counts for each lane.

II. System Overview

The system is constructed from four modules: background learning, foreground extraction, vehicle detection and vehicle classification. Figure 2 illustrates the flow chart of the AutoVDCS. The system can take digitized video images or a live video signal as input. In the vehicle detection module, the user specifies a vehicle census zone and the three lines as a virtual loop detector: StartLine (SL), MiddleLine (ML) and EndLine (EL) as shown in Fig.3. If the camera is fully calibrated, we can train the system using synthetic data. The details will be described in Section VII. However, if the camera is uncalibrated, the system must be trained using manually annotated data, as described in [6], or by annually adding vehicle class annotations to the automatically detected foreground blobs, as described in Section VIII. Section IV gives details of the vehicle detection and the vehicle classification modules.

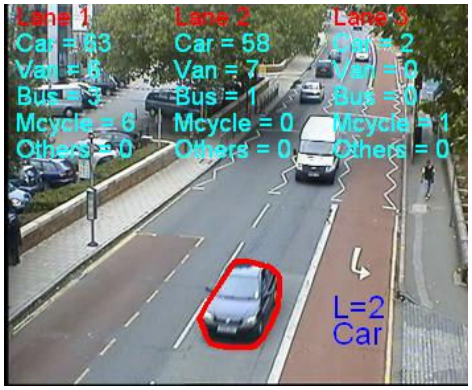


Fig. 1. Automatic annotated video output from AutoVDCS.

Digital Imaging Research Centre, School of Computing and Information Systems, Kingston University, Kingston upon Thames, London, UK. (e-mail: {z.chen, t.ellis@kingston.ac.uk}, sergio.velastin@ieee.org).

978-1-4673-3063-3/12/$31.00 ©2012 IEEE

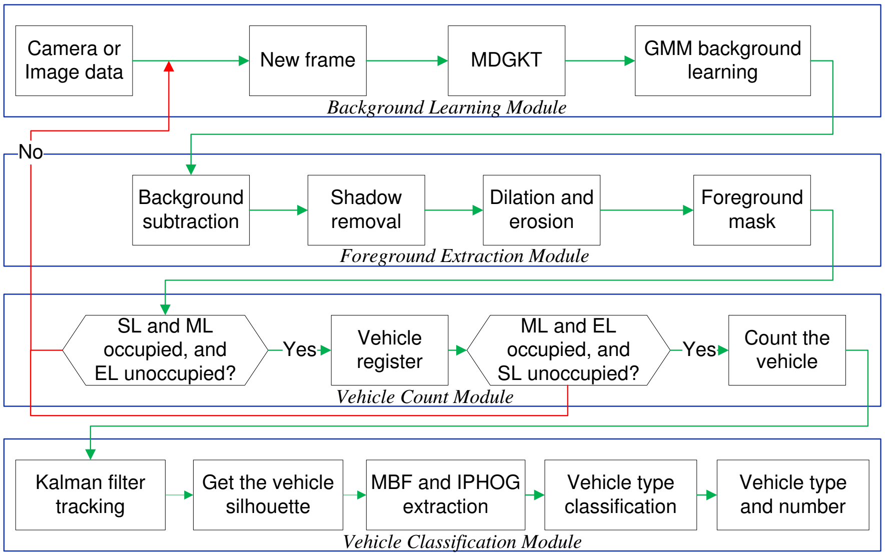


Fig.2. Flow chart of the AutoVDCS.

III. Background Learning and Foreground Extraction

A static camera observing a scene is a common task in a surveillance and monitoring system. Background modeling is often used to model the background, from which moving objects can be detected in the scene. The principal challenges are how to correctly and efficiently model and update the background model, and how to deal with shadows. A robust system should be independent of the scene, and robust to lighting effects and changeable weather conditions. It should be capable of dealing with movement through cluttered areas, objects overlapping in the visual field, gradual illumination changes (e.g. time of day, evening and night), sudden illumination changes (e.g. switching a light on or off, clouds moving in front of the sun), camera automatic gain control (e.g. white balance and iris are often applied to optimally map the amount of reflected light to the digitizer dynamic range), moving background (e. g. camera vibration, swaying trees, snowing or raining day), slow-moving objects, cast shadows and geometric deformation of foreground objects.

At the heart of any background subtraction algorithm is a statistical model that describes the state of each background pixel. In an urban traffic environment, a 'pure' background is not always available and can always be changed under critical situations by objects being introduced or removed from the scene, and slow-moving or stationary objects. To account for these problems, many background modeling methods have been developed. A Gaussian mixture model (GMM) was proposed for real-time tracking by Stauffer and Grimson [7, 8]. The algorithm relies on assumptions that the background is visible more frequently than the foreground and that the model has a relatively narrow variance. Zivkovic and Heijden [9] presented an improved GMM model using a recursive computation to constantly update the parameters of a GMM, which adaptively chooses the appropriate number of Gaussians to model each pixel on-line, from a Bayesian perspective. Many researchers have adapted this model for traffic analysis [10-12] using a fixed number of Gaussians. Rapid illumination changes constitute a particular difficulty when applying background subtraction in a real life setting. The work here is motivated by the need for robust vehicle detection and a classification algorithm that can be used in a traffic monitoring system to deal with such changes. Firstly, in our background learning procedure the Multi-Dimensional Gaussian Kernel density Transform (MDGKT) has been used to deal unwanted motions associated with camera vibration or swaying trees [13]. Secondly, for all pixels *s* in set *S*, a self-adaptive Gaussian mixture model using a global illumination change factor between the current image i_c and the reference image (modeled background) i_r is defined as:

$$g = median_{s\in S}\left(\frac{i_{c,s}}{i_{r,s}}\right) \tag{1}$$

to deal with sudden illumination changes and camera automatic gain control. Furthermore, the Mahalanobis distance of the *m*th Gaussian component is calculated as:

$$D_m^2\left(x^{(t)}\right) = \hat{\delta}_m^T \Sigma_m^{-1} \hat{\delta}_m \tag{2}$$

where $\hat{\delta}_m = g \cdot x^{(t)} - \mu_m$, g is global illumination change factor, $x^{(t)}$ is current intensity (colour), μ_m is the estimation of the mean of the *m*th Gaussian component, $\sum_m = \sigma_m \mathbf{I}$ and I is a 3×3 identity matrix. [14] gives further details regarding the self-adaptive Gaussian mixture model.

For the *j*th pixel value of $I_j = \left[I_{Rj}, I_{Gj}, I_{Bj}\right]^T$ in RGB space, the estimated mean is $E_j = \left[\mu_{Rj}, \mu_{Gj}, \mu_{Bj}\right]^T$. Considering balancing color bands by rescaling the color values by the pixel standard deviation σ_j, the brightness and chromaticity distortion become:

$$B_j = \frac{gI_jE_j}{E_j^2} \qquad CD_j = \frac{\sqrt{gI_j - B_jE_j}}{\sigma_j} \tag{3}$$

Then a pixel in the foreground obtained by the GMM above is identified by

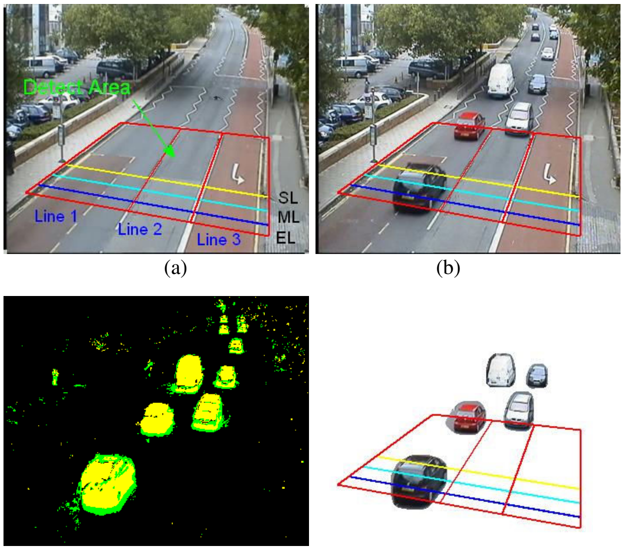
$$\begin{cases} shadow & CD_j < \gamma_1 \;\; and \;\; B_j < 1 \\ Highlight & CD_j < \gamma_1 \;\; and \;\; B_j > \gamma_2 \end{cases} \tag{4}$$

where γ_1 is a threshold value selected to determine the similarities of the chromaticity between the background learnt by GMM and the current observed image. If there is a case where a pixel from a moving object in the current image contains a very low RGB value, then this dark pixel will always be misclassified as a shadow, because the value of the dark pixel is close to the origin in RGB space and all chromaticity lines in RGB space meet at the origin. Thus a dark color point is always considered to be close or similar to any chromaticity line. A threshold γ_2 is introduced for the normalized brightness distortion to avoid this problem. This is defined as: $\gamma_2 = 1/(1-\varepsilon)$, where ε is a lower band for the normalized brightness distortion. An automatic threshold selection method was provided by Horprasert et al. [15].

IV. Vehicle Detection

There are several key considerations when implementing a vehicle detection algorithm, and they vary depending on the specific task. For traffic flow statistics, it is essential to count each vehicle only once. To ensure that vehicles will only be counted as they appear in the detection zone, a virtual loop detector is applied. The virtual loop is comprised of three detect lines, StartLine (SL), MiddleLine (ML) and EndLine (EL). These line detectors are sensitive to miss-detection as a consequence of the ragged edge of a vehicle boundary. To minimize this effect the detectors have a finite width to ensure a stable detection of the vehicle when it intersects the line (a width of 5 pixels was used in the experiments described later). The separation between detector lines depends on the average traffic speed, and was set to 30 pixels in our experiments. The traffic speed limitation is 30 miles per hour. The detector is configured to operate in both directions, to accommodate the two directions of traffic flow, and should be placed at a location where vehicles are clearly visible with minimal occlusion, i.e. usually closest to the camera. A detector is allocated to each lane to handle the measurements for each traffic stream.

Figure 3 illustrates the object detection procedure. Shadow, road reflection and reflection highlights pixels are removed, followed by a post-processing binary morphological opening to remove noise and small area objects. To ensure that vehicles are only counted once the detector considers a vehicle to be "present" only when both SL and ML are occupied and EL is unoccupied (for traffic moving towards the camera, i.e. lane 2 and 3). A vehicle is said to be "leaving" when ML and EL are occupied and SL unoccupied. A vehicle is counted only when it changes from the "present" state to the "leaving" state. This is reasonable in congested situations and even stationary traffic. In this way, the detector will not over-count in either case. If the proportion of pixels intersecting the detection line is above a threshold (30% of the lane width), the line is considered occupied, otherwise it is unoccupied. This threshold is chosen as a tradeoff between detecting small vehicles (such as bicycles and motorbikes) but being insensitive to small blobs associated with noise. It is only necessary to swap SL and EL to account for vehicles in the traffic stream moving away from the camera (e.g. lane 1 in Figure 3(a)).



(a) (b)

(c) (d)

Fig. 3. Vehicle detection: (a) background GMM and virtual loop detector; (b) current input image with detection lines. (c) background subtraction results: black pixels represent the modeled background, foreground object (yellow), shadow (green) or reflection highlights (red); (d) foreground image created by extracting the pixels from the original frame using the final foreground object mask.

V. Vehicle Tracking and Labeling

The output of the vehicle detection step is a binary object mask which is used to perform region tracking. This provides multiple instances of the same vehicle, each of which are independently classified. Tracking employs the centroid of each detected blob, using a constant velocity Kalman filter [16] model. The state of the filter is the centroid location and velocity, $s = [c_x, c_y, v_x, v_y]^T$, and the measurement is an estimate of this entire state, $y = \hat{s} = [\hat{c}_x, \hat{c}_y, \hat{v}_x, \hat{v}_y]^T$. The data association problem between multiple blobs is addressed by comparison of the predicted centroid location with the centroids of the detections in the current frame. The blob with it's centroid closest to the predicted location is chosen as the best match. The track class label is computed at each frame but the final label is assigned by a majority voting scheme which considers the entire track to make a decision on class type, rather than employing a single frame that could be corrupted by different noise sources. Fig. 4 shows the results of vehicle labeling with a track identifier.

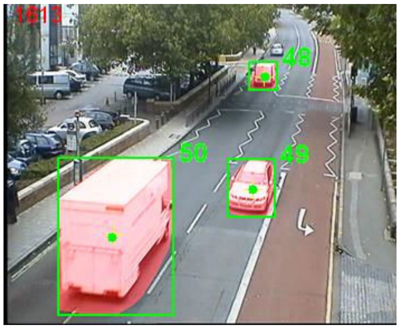


Fig. 4. Kalman filter track labeling results.

VI. Evaluation Metrics

To evaluate the performance of AutoVDCS, an extended confusion matrix for N classes (with absolute counts in rows/columns $C_1, C_2, \ldots, C_N$) is used as follows [17]:

$$\text{Predicted classes}\;\begin{array}{c|cccc|c}
\multicolumn{6}{c}{\text{Ground truth classes}} \\
 & C_1 & C_2 & \cdots & C_N & FP \\ \hline
C_1 & c_{1,1} & c_{1,2} & \cdots & c_{1,N} & c_{1,N+1} \\
C_2 & c_{2,1} & c_{2,2} & \cdots & c_{2,N} & c_{2,N+1} \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
C_N & c_{N,1} & c_{N,2} & \cdots & c_{N,N} & c_{N,N+1} \\ \hline
FN & c_{N+1,1} & c_{N+1,2} & \cdots & c_{N+1,N} & c_{N+1,,N+1}
\end{array}$$

If no overlapping ground truth vehicle is found, the automatic measurement is counted in the FP (false positive) column. All undetected vehicles are entered into the FN (false negative) row.

The metrics normally used for classification evaluation are (normalized) recall (REC), precision (PRE) and F_1.

$$REC = \frac{TP}{TP + FN} \qquad PRE = \frac{TP}{TP + FP} \tag{5}$$

$$F_1 = \frac{2 \cdot REC \cdot PRE}{REC + PRE} \tag{6}$$

The number of true positives (TP) for any class C_i are the corresponding diagonal elements $c_{i,i}$. The recall REC_{ci} for each class and the precisoin PRE_{ci} for each class are defined as:

$$REC_{ci} = \frac{c_{i,i}}{\sum_{j=1}^{N+1} c_{j,i}} \qquad PRE_{ci} = \frac{c_{i,i}}{\sum_{j=1}^{N+1} c_{i,j}} \tag{7}$$

The joint REC, PRE scores for all classes are:

$$JREC = \frac{\sum_{i=1}^{N} c_{i,i}}{\sum_{i=1}^{N}\sum_{j=1}^{N+1} c_{j,i}} \qquad JPRE = \frac{\sum_{i}^{N} c_{i,i}}{\sum_{i=1}^{N}\sum_{j=1}^{N+1} c_{i,j}} \tag{8}$$

The overall classification accuracy (ACC) and detection rate DTR are defined as:

$$ACC = \frac{\sum_{i=1}^{N} c_{i,i}}{\sum_{i=1}^{N}\sum_{j=1}^{N} c_{i,i}} \qquad DTR = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N+1} c_{i,j}}{\sum_{i=1}^{N+1}\sum_{j=1}^{N} c_{i,j}} \tag{9}$$

The overall false positive rate (FPR) and false negative rate (FNR) are defined as:

$$FPR = \frac{\sum_{i=1}^{N} c_{i,N+1}}{\sum_{i=1}^{N+1}\sum_{j=1}^{N} c_{i,j}} \qquad FNR = \frac{\sum_{j=1}^{N} c_{N+1,j}}{\sum_{i=1}^{N+1}\sum_{j=1}^{N} c_{i,j}} \tag{10}$$

VII. Training SVM with Synthetic Data

Manual data collection and annotation is time consuming and costly so we have experimented with using synthetic data to train a classifier. Once the camera is calibrated, a vehicle wireframe model can be projected onto the road surface to create a view-independent synthetic Measurement Based Feature (MBF, a silhouette outline of the model) to train the SVM (see [6] for further details of the camera calibration, vehicle wireframe models and MBF). To model the variety and range of real vehicle silhouettes and noise effects in the background subtraction results, Gaussian noise is added to the control points of the wireframe model, projected position and projected lane direction. This creates variation in the size, shape, position and orientation of the projected vehicle wireframe and hence the resulting vehicle silhouette. A closed convex polygon around their extremal boundary is constructed to create synthetic MBF. A total of 3600 synthetic samples (car: 1482, van: 927, bus: 878, motorcycle: 313) were created. The type distribution is similar to the ground truth data presented in [6]. The separability of the four vehicle classes can be visualized by plotting the first three most significant PCA (principal component analysis) components of a normalized (mean and standard deviation scaled in range 0-1) feature vector, where one class (motorcycle) is unambiguously separable (see figure 5). The car, van and bus categories exhibit a range of feature measures that are not clearly separable. For synthetic data, we choose the best classifier parameters to be γ =2 for the Gaussian kernel, and C=50. γ and C are two SVM parameters should be tuned according to dataset. A 10 fold cross-validation strategy was employed to evaluate the performance of the classification methods. The mean and *std* of the REC, PRE and F1 of each classes are given in Table I. The support vectors of the best performing SVM classifier are used to predict the entire data set. The associated confusion matrix is given in Table II. The row sum is the ground truth number of each vehicle class.

TABLE I. The Mean and std of the REC, PRE and F1 of Each Class from Synthetic Data

	Car	Van	Bus	Motorcycle
REC	0.949 ± 0.020	0.808 ± 0.033	0.951 ± 0.024	1.0 ± 0.0
PRE	0.898 ± 0.016	0.863 ± 0.033	0.976 ± 0.015	1.0 ± 0.0
F1	0.923 ± 0.015	0.833 ± 0.027	0.963 ± 0.014	1.0 ± 0.0

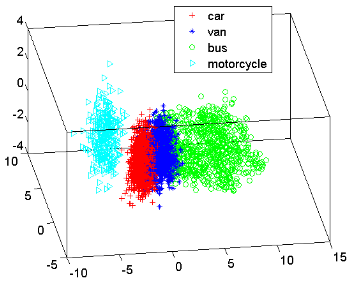


Fig. 5. The first three most significant PCA components of synthetic data

TABLE II. CONFUSION MATRIX FOR SYNTHETIC DATA

	Car	*Van*	*Bus*	*Motorcycle*
Car	1414	68	0	0
Van	148	761	18	0
Bus	0	32	846	0
Motorcycle	0	0	0	313

VIII. TRAINING SVM BY AUTOMATIC DETECTED DATA

In order to investigate if using synthetic data to train the system is sufficient to predict a vehicle's category, the SVM is trained with manually labeled and automatically detected foreground blobs. The silhouettes of the foreground blob are not well defined and occasionally only a small part of a vehicle is detected. Because of the complexity of the detected foreground blob, we only annotate and use blobs for which the proportion of pixels in the convex hull of the foreground that are also in the ground truth region is greater than 33%. We used 8 video clips (totally 117941 frames, approximately 78.6 minutes) which were acquired under different weather conditions (overcast sky, light and heavy rain). The capture rate is 25 frames per second and the image size was 352×288. Vehicle classification uses the majority vote for 5 sequential video frames. Each vehicle contributes 5 separate samples, giving a total sample of 5760 car, 530 van, 180 bus and 100 motorcycle observations. A 202-dimensional feature vector is constructed, comprising the MBF and intensity pyramid-based HOG (IPHOG) to train the SVM. For automatically detected silhouettes, the best parameters are γ =2 for the polynomial kernel, and C=5000, were determined by experiment.

TABLE III. METRICS OF EACH CLASS FROM DETECTED DATA

	Car	Van	Bus	Motorcycle
REC	0.998 ± 0.002	0.983 ± 0.017	0.983 ± 0.027	0.970 ± 0.068
PRE	0.998 ± 0.002	0.976 ± 0.021	0.995 ± 0.017	0.989 ± 0.035
F1	0.998 ± 0.001	0.979 0.014	0.989 ± 0.015	0.979 ± 0.051

We employed a 10-cross validation strategy, repeating the process 10 times and averaging the results in order to evaluate the performance of the classification methods. The mean and *std* of the REC, PRE and F1 of each classes are given in Table III. The support vectors of the best performing SVM classifier are used to predict the entire data set. The associated confusion matrix is given in Table IV. It shows that all samples are perfectly classified.

TABLE IV. CONFUSION MATRIX FOR DETECTED VEHICLES

	Car	Van	Bus	Motorcycle
Car	5760	0	0	0
Van	0	530	0	0
Bus	0	0	180	0
Motorcycle	0	0	0	100

IX. AUTOVDCS EVALUATION

To evaluate AutoVDCS in real scenarios, we use the video clips described in Section VIII. As previously emphasized, each vehicle must only be counted once. There are a total of 1402 cars, 216 vans, 82 buses and 27 motorcycles identified in the video. In order to extensively analyze performance, we compare 7 different combinations of test and training data:

Method 1: train SVM using synthetic MBF, classify foreground blobs obtained by background subtraction;

Method 2: train SVM using synthetic MBF, classify foreground blobs obtained by background subtraction and majority vote over 5 frames;

Method 3: train SVM using manually extracted MBF, classify foreground blobs obtained by background subtraction and majority vote over 5 frames;

Method 4: train SVM using detected MBF obtained by background subtraction, classify foreground blobs and majority vote over 5 frames;

Method 5: train SVM using detected MBF+IPHOG from background subtraction, classify foreground blobs and majority vote over 5 frames;

Method 6: train SVM using detected MBF from background subtraction, classify foreground blobs obtained by background subtraction and levelset detection, and majority vote over 5 frames;

Method 7: train SVM using detected MBF+IPHOG from background subtraction, classify foreground blobs obtained by background subtraction and levelset detection, and majority vote over 5 frames.

Methods 6 and 7 employ the improved levelset method proposed in [18]. The main advantage of this method is that it is an active segmentation method. The levelset energy formulation includes information on the mixture of multiple channels and multiple regions. Object boundaries that include different known colors are segmented against complex backgrounds and it is not necessary for the object to be homogeneous. The main drawback of using the level set method is that it needs long time to converge. It is not suitable for real time process systems.

At the training stage, 10-cross validation strategy is used to evaluate the system. The support vectors from the best performance are selected to classify the entire dataset. The comparison of vehicle detection accuracy is given in Table V shows that the level set algorithm improves detection performance.

The classification results in terms of JREC, JPRE and F1 are given in Table VI and show that method 5 results in the best combination, indicating that whilst using levelset detection improves the vehicle detection rate, it results in a lower classification rate. This is because when multiple vehicles are too close and in the detection area, they will be merged into a single blob by background subtraction, but levelset detection segments them into multiple regions. However, if a vehicle's color is similar to the background, it's silhouette may be significantly reduced, and as a result SVM classifies it incorrectly. The REC of the extended confusion matrix of method 5 for each class are given in Table VII. The classification accuracy is 94.69%. Figure 1 shows the final output as a count of each vehicle type for each traffic lane.

TABLE V. VEHICLE DETECTION ACCURACY

	DTR	FNR	FPR
Background subtraction	0.9444	0.0724	0.0168
Background subtraction + levelset	0.9639	0.0497	0.0136

TABLE VI. THE COMPARISON OF CLASSIFICATION ACCURACY

	JREC	JPRE	F1
Method 1	0.8135	0.8620	0.8371
Method 2	0.8193	0.8676	0.8428
Method 3	0.8054	0.8529	0.8285
Method 4	0.8674	0.9185	0.8922
Method 5	**0.8784**	**0.9301**	**0.9035**
Method 6	0.8399	0.8713	0.8553
Method 7	0.8817	0.9147	0.8979

TABLE VII. THE RECS OF EXTENDED CONFUSION MATRIX OF METHOD 5

	Car	Van	Bus	Motorcycle	FP
Car	0.9123	0.0972	0.0122	0	0.0086
Van	0.0043	0.7315	0.0366	0.0370	0.0278
Bus	0.0228	0.0926	0.7195	0	0.0732
Motorcycle	0.0007	0	0	0.7778	0.1852
FN	0.0599	0.0787	0.2317	0.1852	0

X. CONCLUSIONS

Acquisition of reliable vehicle counts and classification data is necessary to establish an enriched information platform and improve the quality of ITS. The approach proposed in this paper is a hybrid algorithm. In the background subtraction module, foreground extraction module and vehicle detection module, the improved background subtraction method is integrated to alleviate the negative impacts from camera vibration, shadow and reflection highlights, sudden illumination changes and more gradual changes. A levelset method has been used to refine the foreground blob. In the vehicle classification module, a Kalman filter and SVM are integrated to improve accuracy. Extensive experiments have been undertaken, comparing 7 combinations of detection and classification methods. Results show that the best combination is to train the SVM using MBF+IPHOG features extracted by background subtraction, classifying the foreground blobs using a majority vote over 5 consecutive frames. The results demonstrate a vehicle detection rate of 96.39% and classification accuracy of 94.69% under varying illumination and weather conditions.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the Royal Borough of Kingston for providing the video data.

REFERENCES

[1] S. Messelodi, C.M. Modena, and M. Zanin, "A computer vision system for the detection and classification of vehicles at urban road intersections," *Pattern Analysis & Applications*, 2005, 8(1-2):17–31.

[2] G.T. Kogut, M.M. Trivedi, "Maintaining the identity of multiple vehicles as they travel through a video network," in *Proc. IEEE Conference on ITS*, Oakland, California, 2001, pp. 756-761.

[3] S. Bhonsle, M.M. Trivedi, and A. Gupta, "Database-centered architecture for traffic incident detection, management, and analysis," in *Proc. IEEE Conference on Intelligent Transport. System*, Dearborn, Michigan, Oct. 2000, pp. 149–154.

[4] R. Chang, T. Gandhi, and M.M. Trivedi, "Vision modules for a multisensory bridge monitoring approach," in *Proc. IEEE Conf. on Intelligent. Transport Sysem.*, Oct. 2004, pp. 971–976.

[5] W. Hu, X. Xiao, D. Xie, T. Tan, and S. Maybank, "Traffic accident prediction using 3-d model-based vehicle tracking," *IEEE Trans. on Vehicle Technology,* May 2004, vol. 53, no. 3, pp. 677–694.

[6] Z. Chen, T. Ellis and S.A. Velastin, "Vehicle type categorization: A comparison of classification schemes," *14th IEEE Annual Conference on Intelligent Transportation Systems,* Oct. 5-7, 2011, The George Washington University, Washington, DC, USA. pp. 74-79.

[7] C. Stauffer, W. Grimson, "Adaptive background mixture models for real-time tracking," *Proc. of IEEE CVPR*, 1999, vol. 2, 246-252.

[8] C. Stauffer, W. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. PAMI*, 2000, 22(8): 747-757.

[9] Z. Zivkovic and F. Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letters*, May 2006, 27(7): 773-780.

[10] W. Zhang, Q. Wu, X. Yang and S. Fang, "Multilevel framework to detect and handle vehicle occlusion," *IEEE Transaction on Intelligent Transportation System*, 2008, 9(1), 161-174.

[11] J. Wang. Y. Ma, C. Li H. Wang and J. Liu, "An efficient multi-object tracking method using multiple particle filters," *WRI World Congress on Computer Science and Information Engineering*, 2009, 568-572.

[12] B. Johansson, J. Wiklund, P.E. Forssen and G. Granlund, "Combining shadow detection and simulation for estimation of vehicle size and position," *Pattern Recognition Letters*, 2009, 30(8), 751-759.

[13] Z. Chen, N. Pears, M. Freeman and J. Austin, "Background subtraction in video using recursive mixture models, spatio-temporal filtering and shadow removal," *Proc. of 5th ISVC*, in Lecture Notes in Computer Science, Vol. 5876, 2009, pp. 1141-1150.

[14] Z. Chen, T. Ellis, "Self-adaptive Gaussian mixture model for urban traffic monitoring system," *IEEE International Conference on Computer Vision Workshop*, Barcelona, Spain, 2011, pp. 1769-1776.

[15] T. Horprasert, D. Harwood, L. S. Davis, "A statistical approach for real-time robust background subtraction and shadow detection," in*: Proceedings of IEEE ICCV'99 Frame rate workshop*, 1999, pp. 1-19.

[16] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," *SIGGRAPH 2001 course 8. In Computer Graphics, Annual Conference on Computer Graphics & Interactive Techniques*. ACM Press, Addison-Wesley, SIGGRAPH 2001 course pack edition, 2001.

[17] *N. Buch*, J. Orwell and S.A. Velastin, "Urban road user detection and classification using 3D wire frame models," *IET Computer Vision*, 4(2), June, pp. 105-116, *2010*.

[18] Z. Chen, A.M Wallace, "Active Segmentation and Adaptive Tracking Using Levelsets," *BMVC 2007*, September 2007, pp. 920-929.