Attribute-based Vehicle Search in Crowded Surveillance Videos

Rogerio Feris
IBM T. J. Watson Research Center, New York, USA
rsferis@us.ibm.com

Behjat Siddiquie
University of Maryland, Maryland, USA
behjat@cs.umd.edu

Yun Zhai
IBM T. J. Watson Research Center, New York, USA
yunzhai@us.ibm.com

James Petterson
Australian National University / NICTA, Canberra, Australia
jpetterson@gmail.com

Lisa Brown
IBM T. J. Watson Research Center, New York, USA
lisabr@us.ibm.com

Sharath Pankanti
IBM T. J. Watson Research Center, New York, USA
sharat@us.ibm.com

ABSTRACT

We present a novel application for searching for vehicles in surveillance videos based on semantic attributes. At the interface, the user specifies a set of vehicle characteristics (such as color, direction of travel, speed, length, height, etc.) and the system automatically retrieves video events that match the provided description. A key differentiating aspect of our system is the ability to handle challenging urban conditions such as high volumes of activity and environmental factors. This is achieved through a novel multi-view vehicle detection approach which relies on what we call *motionlet classifiers*, i.e. classifiers that are learned with vehicle samples clustered in the motion configuration space. We employ massively parallel feature selection to learn compact and accurate motionlet detectors. Moreover, in order to deal with different vehicle types (buses, trucks, SUVs, cars), we learn the motionlet detectors in a shape-free appearance space, where all training samples are resized to the same aspect ratio, and then during test time the aspect ratio of the sliding window is changed to allow the detection of different vehicle types. Once a vehicle is detected and tracked over the video, fine-grained attributes are extracted and ingested into a database to allow future search queries such as "Show me all blue trucks larger than 7ft length traveling at high speed northbound last Saturday, from 2pm to 5pm".

See video demos of our system at:
`http://rogerioferis.com/ICMR2011/`

Categories and Subject Descriptors

I.4 [**Image Processing and Computer Vision**]: Applications, Scene Analysis - Object Recognition

General Terms

Algorithms

ICMR '11, April 17-20, Trento, Italy

Keywords

Vehicle detection, video analytics, search and retrieval.

1. INTRODUCTION

Searching for suspicious vehicles is a common and important task in criminal investigation processes. Eyewitnesses are typically asked by the police to fill out a *vehicle description form*, where they indicate the characteristics of the vehicle as seen at the moment of a suspicious activity or when a crime was committed. Those include direction of travel, license plate number, color, size, speed, body shape, wheel covers, decals, damages, presence of features such as sunroof, etc. Based on that description,the police manually scan the entire video archive looking for vehicles with similar characteristics. This process is tedious and time consuming.

Our goal in this paper is to automate this process by providing a vehicle search system based on semantic attributes. Our current implementation allows the user to search for vehicles based on color, size, length, width, height, speed, direction of travel, date/time, and location, but many more attributes could be considered, including measurements from non-visual sensors. Previous solutions have generally relied on license plate recognition [1] or vehicle classification [15, 12], which may not be effective for low-resolution cameras or when the plate number is not available. Instead we provide a complementary search framework based on fine-grained attributes. In addition to searching for suspicious vehicles, our system could also be used in other applications. For instance, transportation agencies are interested in finding trucks with a length or height larger than the permitted sizes in specific highways. There is also interest in correlating visual vehicle attributes with carbon emission measurements obtained from other sensors. Search based on attributes may also help identifying specific vehicles, e.g., DHL trucks or taxis can be often identified based on their color and size.

A key innovative aspect of our work is the ability to handle challenging urban conditions such as crowded scenes and lighting changes. Traditional surveillance systems based on background modeling [17, 8] generally fail to segment vehicles in crowds, as multiple vehicles tend to get clustered into a single motion blob. We propose a multi-view detection system that relies on a set of *motionlet classifiers*, which consist of detectors learned with vehicle samples clustered in the motion configuration space. Each detector is learned

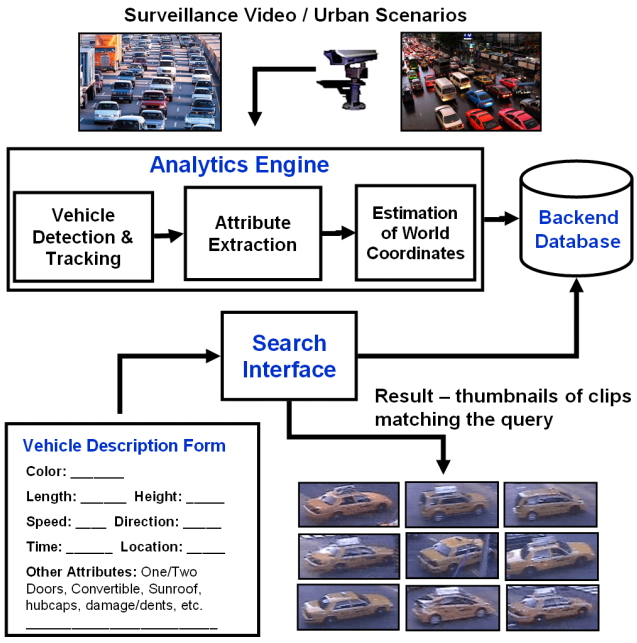

Figure 1: System architecture. At the interface, the user specifies a set of vehicle attributes (vehicle description form) and the system retrieves the events matching the provided description. Our system is designed to handle high volumes of activity.

by using massively parallel feature selection of local descriptors. In addition, multiple types of vehicles, such as buses, trucks, SUVs, and compact cars can be detected with our approach by training the motionlet detectors in a shape-free appearance space, where all training images are resized to the same aspect ratio. At test time, the aspect ratio of the sliding window is changed to detect multiple vehicle types.

Once vehicles have been detected and tracked, attributes are extracted and ingested into a database. In particular, measurements such as speed, width, height, and length of vehicles are converted to world coordinates through a simple calibration process, thus allowing search across cameras without perspective issues. Figure 1 shows a high-level overview of the architecture of our system

Summarizing, the key **contributions** of our paper are 1) a novel application for vehicle search in urban/crowded environments based on fine-grained semantic attributes and 2) a novel multi-view vehicle detection approach based on motionlet detectors which are learned in a shape-free appearance space with large-scale feature selection. We have implemented a complete system which has been deployed in several cities worldwide (see `http://rogerioferis.com/ICMR2011/InterfaceAndOperation.wmv`).

2. RELATED WORK

In the past few years, semantic attributes have received renewed attention in the computer vision community, providing effective high-level object representations for a variety of applications, including zero-shot recognition [10, 5], face verification [9], and people search [20]. Although semantic concepts and attributes have been inherently present in image retrieval, scene classification, and broadcast video search, we are not aware of any surveillance system capable of searching for vehicles based on their fine-grained attributes.

Most commercial surveillance systems rely on background modeling for detection of moving objects, in particular vehicles. However they fail to handle crowded scenes as multiple objects close to each other are often merged into a single motion blob. Environmental factors such as shadow effects, rain, snow, etc. also cause issues for object segmentation.

Appearance-based object detectors based on e.g., Adaboost Learning [21, 22] or SVMs [4] are more robust to crowds and environmental factors. A major challenge for these methods is how to deal with highly non-linear appearance changes in the training set, mainly introduced by object pose variations. Previous solutions that address this problem usually split the training data based on appearance clustering [22, 11]. Although tree-based architectures can improve efficiency, these techniques are relatively slow at test time as multiple detectors need to be applied. Small partitions can also lead to overfitting. More recently, Bourdev et al. proposed poselet detectors [2] which are trained with data clustered in the 3D pose configuration space. The clusters therefore contain semantic information and can be used to simultaneously detect objects and estimate their 3D pose configuration. The pose information in the training data, however, has to be manually annotated. In contrast, our approach based on motionlets *automatically* splits the training data according to motion clusters and offers advantages in terms of efficiency and accuracy at test time.

Another drawback of current appearance-based detectors is their limited ability to handle multiple vehicle types (buses, SUVs, etc.). The reason being that, at test time these techniques employ a sliding window with fixed aspect ratio and therefore cannot provide the precise width and height of different vehicle types. An alternative solution is to train separate detectors for multiple vehicle classes, but the number of classes can be large and many samples per class are required to avoid overfitting. Deformable part-based object detectors [6] allow deformable bounding boxes and have recently achieved state-of-the-art results in various object categories. However they are not suitable for low resolution images and not efficient to be used in real surveillance deployments which may require many video channels to be processed per server. Our approach learns to detect multiple vehicle types in a single appearance space, while running at more than 60Hz on conventional machines. We use the term *shape-free appearance space* to denote the image space of objects with the same aspect ratio, without taking into account their detailed shape outlines as in Active Appearance Models [3].

A vehicle detection system with similar efficiency as ours was proposed in [7]. However, in this work only view-dependent classifiers are trained and the method fails to detect large vehicles such as buses or trucks. Our approach differs in the application itself, the use of motionlets, the ability to detect multiple vehicle types, and the estimation of their dimensions in world coordinates.

Large-scale learning is an emerging research topic in computer vision and multimedia. Recent methods have been proposed to deal with a large number of object classes [16] and large amounts of data. In contrast, our approach deals with large-scale feature selection, showing that a huge amount of local descriptors over multiple feature planes coupled with

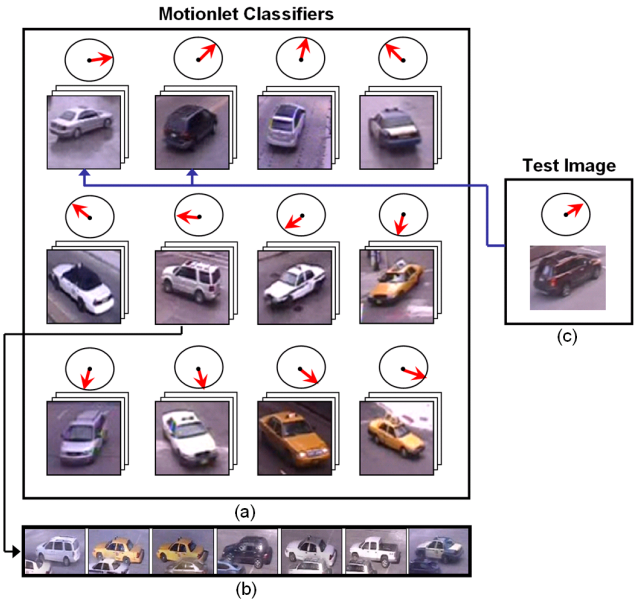

Figure 2: (a) The training set is automatically split into clusters of vehicles with similar motion direction (motionlets). (b) Artificially generated samples with occlusions are added to the training data to enhance the performance in crowded scenes. (c) At test time, the motion direction of a particular image patch is used to select specific motionlet classifiers which determine whether a vehicle is present or not.

parallel machine learning algorithms can improve not only the accuracy of motionlet detectors, but also their efficiency at test time.

3. CAPTURING VEHICLES IN CROWDED ENVIRONMENTS

In this section we describe our method for *multi-view* vehicle detection, which is designed to work well in typical urban surveillance environments - involving crowds, lighting changes, and different vehicle types, while running at high frame rates.

3.1 Training Data

We have collected a huge training set of images containing *nearly one million vehicles of different types, multiple lighting conditions, and many different poses*. The data was captured from a set of city surveillance cameras using a semi-automatic approach: motion blobs in the input videos are initially detected via background modeling [18] and the blobs in a user-defined region of interest having an acceptable size, aspect ratio, and direction of motion are automatically added to the training set. Then false positives are manually removed. Each training image in the dataset contains the associated motion direction of the vehicle (obtained through optical flow).

We use the term *motionlet* to describe a set of images clustered in the motion configuration space, i.e., images containing vehicles with similar motion direction. As shown in Figure 2a, we automatically split the training set into 12 clusters/motionlets. Since vehicles are rigid objects and move along their longitudinal axis, each motionlet is semantically related to the vehicle pose information. We used standard 2D optical flow to determine motion direction, but 3D methods such as structure from motion could also be applied.

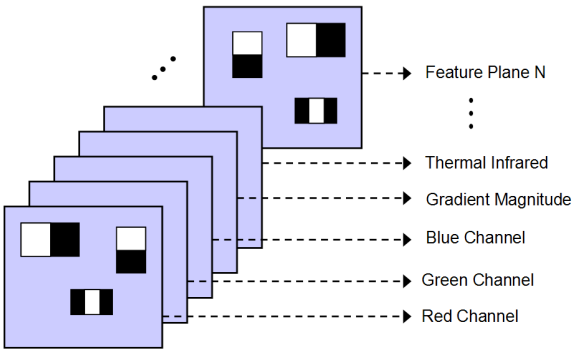

Figure 3: Motionlet detectors are learned using feature selection over multiple feature planes. A feature pool containing a huge set (order of millions) of feature configurations is generated.

In order to better deal with occlusions, which are common in crowded scenarios, we artificially generated more samples by synthetically creating images of vehicles occluding each other using Poisson image editing [14] (Figure 2b). Finally, we resize all training images so that they have the same aspect ratio - in our implementation, 24x24 pixels. This way we can model only the appearance of all vehicle types irrespective of their sizes. The set of resized images therefore belongs to what we call a shape-free appearance space, since the information about aspect ratio is lost (Figure 4a). As we will see in Section 3.3, the aspect ratio of the sliding window is varied at test time to detect multiple vehicle types.

3.2 Large-Scale Motionlet Detectors

For each motionlet, we learn a detector based on a variation of Adaboost learning [21] and call it a motionlet detector/classifier. The learning process for a particular motionlet receives as input a set of positive images, i.e. 24x24 vehicle images with similar motion direction, a set of negative images, which are 24x24 non-vehicle image patches obtained from around 1000 background images, and produces a classifier that separates vehicles from non-vehicles. In our implementation, we have learned 12 motionlet detectors covering 12 motion directions, but more or fewer motionlets could be considered depending on the amount of training data.

Our learning algorithm is similar to the cascaded Adaboost classifiers proposed by Viola and Jones [21]. The novel aspect is the introduction of multiple feature planes in the feature selection process, as shown in Figure 3. By considering feature planes such as red, green, and blue channels, gradient magnitude, Local Binary Patterns, and many others, we allow the final motionlet detector to be much more powerful, combining Haar-like features of different modalities. In this framework, feature selection is performed over a pool containing a huge set (potentially millions) of feature configurations. This poses a serious problem in terms of training time, as even with a single feature plane and a few thousand images Adaboost training takes days on a stan-

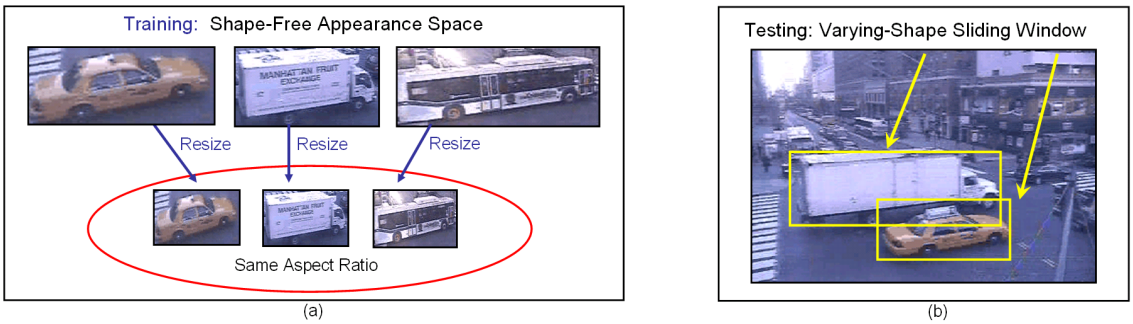

Figure 4: (a) Training images containing multiple vehicle types are resized to have the same aspect ratio. (b) At test time, the aspect ratio of the sliding window is changed to enable the detection of various types of vehicles such as buses, trucks and compact cars.

dard desktop machine. Therefore to deal with a huge pool of local feature descriptors as we are proposing we need a way to parallelize training.

Adaboost is inherently sequential [13], making it difficult to scale in general, but in this particular setup there is a simple solution: parallelization at the level of features. At each step during training we have to compute a large number of features for all training images and select the one that better classifies the data; this can be done in parallel, with each CPU working on a subset of the features, and the amount of synchronization necessary is minimal: each CPU has to report only the best feature of its subset.

Additionally, at each stage a set of negative patches has to be selected from the set of available negative images. The selected patches are the ones for which the current classifier fails. This is the most time consuming activity in later stages of the cascade training, taking hours even for a small training set. Parallelization can also be implemented here, with each CPU searching for negative patches in a different subset of the negative images. Again, the amount of time spent on synchronization[1] here is comparatively very small, allowing for an almost linear speed-up with the number of CPUs employed.

So far in our implementation we have considered parallel feature selection over four color planes (gray-scale, red, green, and blue channels). As we will show in the experimental section, by adding color we not only improve the robustness of the classifier but also get a sparser solution, with a smaller number of selected features. That, in turn, reduces computation time during inference. We are currently adding more feature planes (gradients and texture descriptors, multispectral planes, etc.), which should improve results even more.

3.3 Vehicle Detection and Tracking

In the previous sections, we described the training data preparation and the learning algorithm used to train motionlet detectors, which are offline processes. Now we describe how we apply the motionlet detectors at runtime.

Given a test video frame, we use the traditional sliding window approach [21] to detect vehicles. The novel aspect is that we scan the image not only at multiple positions and scales, but also allow the search window to deform its shape, in particular its aspect ratio, as shown in Figure 4b. This enables the detection of multiple vehicle types, ranging from large trucks or buses to compact cars. Since our cameras are fixed, we apply this search scheme only on motion blobs obtained by background modeling to improve efficiency and accuracy.

For each search window hypothesis, we extract the motion direction of the underlying image patch through optical flow, and use this information to directly select motionlet classifiers with similar motion direction (Figure 2c). In our implementation, we selected the top two motionlet detectors with closest motion direction. Those are applied to the candidate image patch and the results are combined by an OR boolean operator, i.e., if any motionlet detector fires then a vehicle is detected at that specific image location. As we will show in our experimental analysis, this approach offers significant advantages in terms of accuracy over methods that avoid non-linearities in the training set through clustering based on appearance and tree-based architectures.

Assuming fixed surveillance cameras, we capture the structure of the scene by analyzing the most selected motionlet classifiers over a period of time. For instance, some scenes contains vehicles in a single pose, requiring only a single motionlet to be applied. This analysis allows us to remove the optical flow computation step at test time and directly apply the selected motionlet detectors according to the scene structure, thus improving efficiency. Additionaly, we interleave the selected classifiers across the video frames to obtain higher frame rates. Our vehicle detection system runs at more than 60 frames per second, being appropriate for real surveillance deployments which require many video channels to be processed per server.

We also need to link the detections of the same vehicle over the video sequence to avoid indexing attributes of the same vehicle multiple times. This is accomplished by a simple correlation-based tracker combined with vehicle detection at every frame. More specifically, when a vehicle is detected, the correlation-based tracker is triggered. For the subsequent frame, if a vehicle is not detected by any motionlet classifier, then tracking is performed with the window given by the correlation tracker. Otherwise, if the vehicle detector reports a window result with close position and size to the current tracking window, then this vehicle detection window result is used to update tracking. This mechanism is important to avoid drifting.

[1]We used Message Passing Interface (MPI) in our implementation.

4. ATTRIBUTE EXTRACTION / SEARCH

For each vehicle track, we extract a set of fine-grained attributes as described below and automatically ingest the attribute metadata into a backend database system through a web-based service-oriented architecture. SQL event search queries such as "Show me all blue trucks larger than 7ft length traveling at high speed northbound last Saturday, from 2pm to 5pm" can then be placed by the user through a web-based interface. The interface issues requests to a web server, where Java servlets receive the information and issue queries to the database backend. The results are then presented to the user. Thumbnails of the detected vehicles are displayed, and the user can click on them to view a video clip of the selected vehicle. The framework and software architecture of our vehicle search system is similar to the IBM Smart Surveillance Solution [8]. The search interface and a demonstration of our system in operation can be seen in the following video:

`http://rogerioferis.com/ICMR2011/InterfaceAndOperation.wmv`

We currently extract the following metadata and attributes from vehicle tracks:

Date, Time and Location. We store a timestamp indicating the beginning, end, and duration of a track. In addition, the information about the camera and its location in a map is stored, so that the user can search for events in a particular region of the city covered by certain cameras at a particular date/time.

Direction of Travel. This information is is implicitly present in our motionlet classifiers. We currently support search based on 12 directions. A motion direction histogram is built for each vehicle track and the direction with larger number of votes is stored in the database.

Dominant Color. We extract the dominant color for each vehicle, allowing the user to search for vehicles based on 6 colors - black, white, red, green, blue, and yellow. The dominant color is computed by initially converting each input video frame into a bi-conic (hue, saturation, luminance) HSL space, and then quantizing the HSL space into 6 colors. This quantization is done by computing the hue angular cutoffs between the colors and, in a second stage, relabeling pixels as either white or black depending on whether they lie outside the lightness/saturation curve above or below the horizontal mid-plane. This is related to earlier work in color segmentation performed by Tseng and Chang [19]. A cumulative histogram with 6 bins in this quantized space is built over the vehicle images belonging to a specific track. The color corresponding to the bin which receives the majority of votes is then assigned as the dominant color.

Vehicle Dimensions. Our vehicle detection approach provides a precise bounding box and consequently the width and height in pixels for various types of vehicles. Pixel measurements, however, are sensitive to perspective, as a small car can look big if it is close to the camera and the other way round. We solve this issue by calibrating the scene and estimating the **width, height, and length** of vehicles in world coordinates, as described in next section. We take the median value for each dimension over the entire vehicle track and ingest those values in the database.

Speed. Once we have the position of a particular vehicle in world coordinates at each video frame (see next section), it is straighforward to compute its speed. We store the average speed of a vehicle track in the database.

Although our current implementation only considers the above attributes, more attributes such as wheel covers, body shape, presence of features such as sunroof, etc. could be considered.

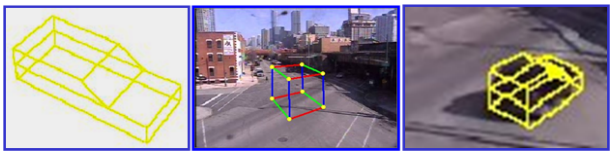

Figure 5: Left: polygonal vehicle model; Center: the UI for using a 3D cube to obtain the camera calibration matrix; Right: an example of fitted model to the observed vehicle using the calibration matrix.

4.1 Estimation of Real-World Features

In many situations like traffic counting and large vehicle monitoring (e.g., trucks), knowing the actual 3D characteristics of the vehicles such as width and length are important and critical. In this section, we present a real-world measuring method utilizing the camera calibration information. In particular, the 3D dimensions of detected vehicles (width, height and length) and their real speeds are estimated. To achieve this, a 3D vehicle model (Figure 5) is incorporated to represent the actual position and orientation of the vehicle in real-world.

The camera calibration is achieved by a manual specification process, where the user can define a 3D cube, which is sitting on the ground plane and having edges with equal length (specified by the user, e.g., 20 ft), through a convenient definition interface. The calibration matrix (a 3x4 matrix) is then estimated using the least square fit method by matching the 3D cube corners with their 2D correspondences in the image plane. The user can also refine the calibration by visually viewing how well the vehicle model is projected to the image plane and fitted to an observed vehicle (Figure 5). Note that in our application, only the projection matrix is needed. It is not necessary to know the exact parameters such as camera location, orientation, focal length, etc. Once the camera calibration matrix is obtained, it is used to match the 3D vehicle model with the target object.

In order to correctly project the 3D vehicle model onto the 2D image such that its projection fits the detected vehicle, three things need to be known for the model: its location on the ground plane, orientation of heading direction and the scale of the model. In our estimation process, the location is initialized as the intersection of the ground plane with the line that goes through the 2D vehicle centroid and the camera center (using backward projection). It is further refined once the other information is known. Assuming there are prior samples of the same detected vehicle from previous frames through the tracking process, its heading direction (denoted as $\mathbf{v}_o$) is estimated as the motion vector between its current 3D location and its previous location on the ground plane. The vehicle model is then rotated such that it aligns with this vector. If the vehicle is static, its previous heading direction will be used. For convenience, its perpendicular vector is denoted as $\mathbf{v}_o^T$.

There are many different types of motor vehicles, such as sedan, SUV, mini-van, medium truck, 18-wheeler, etc. Apparently, only knowing where the vehicle is located and to

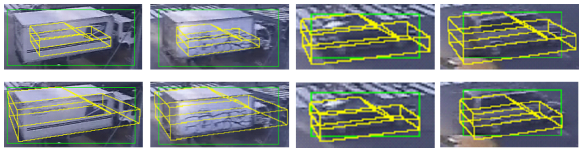

Figure 6: Top Row: projected vehicles before 3D scaling; Bottom Row: better fitted vehicle models after proper scaling.

where it is heading to is not sufficient to distinguish which type of vehicle it is. In this case, the scale of the vehicle must be estimated to make the correct inference. Here we provide a straightforward but robust method for approximating the vehicle scales. Given the bounding box of an observed vehicle in image, it can be denote as $BB_h(l, r, t, b)$, where (l, r, t, b) are the left, right, top and bottom coordinates of the bounding box respectively. Similarly, the bounding box of the 2D projected model can be also obtained and denoted as $BB_m(l, r, t, b)$. The aspect ratio difference between BB_H and BB_M in x and y directions are computed as $S_x = \frac{r_h - l_h}{r_m - l_m}$ and $S_y = \frac{t_h - b_h}{t_m - b_m}$. Utilizing the backward projection technique again, we can find the 3D vector $\mathbf{v}_x$ on the ground plane whose 2D projection aligns with the horizontal axis of the image, and similarly, the 3D vector $\mathbf{v}_y$ for the vertical axis of the image. Since these vectors are on the ground plane, their Y components are dropped. Assuming all the vectors are unit vectors, the scales in the length S_L and width S_W dimensions of the vehicle model can be estimated as following:

$$\begin{cases} S_L = \alpha_L(S_x||(\mathbf{v}_o \cdot \mathbf{v}_x)|| + S_y||\mathbf{v}_o \cdot \mathbf{v}_y||), \\ S_W = \alpha_W(S_x||(\mathbf{v}_o^T \cdot \mathbf{v}_x)|| + S_y||\mathbf{v}_o^T \cdot \mathbf{v}_y||), \end{cases} \quad (1)$$

where $||*||$ represents the absolute value of the vector dot product, and α_L and α_W are the normalization factors. Since the width and height of a vehicle usually are correlated, the scaling factor of model's height is defined as the same as S_W. Figure 6 shows some example results for 3D vehicle scaling. With the availability of vehicle's location, orientation and scale, its 3D features such as real length, width, height and speed can be effectively estimated.

5. EXPERIMENTS

For a quantitative analysis of our vehicle detection approach, we collected a challenging test set from a specific surveillance camera containing 229 images and 374 vehicles, mostly in side-view pose. Experimental analysis for other camera views and other vehicle poses are included as supplementary material at `http://rogerioferis.com/ICMR2011/MoreViews.jpg`. The images were captured in different months, covering different weather conditions incluing sunny and rainy days, different lighting effects, such as shadows and specularities, and different periods of time such as morning and evening. In addition, we split our test set into two groups: high activity, i.e., crowded scenes with many occlusions (104 images and 217 vehicles) and low activity (125 images and 157 vehicles).

We compared three methods in this test set: 1) a baseline method which learns a tree-based Adaboost vehicle detector using clustering based on appearance to deal with non-linearities in the training set, in the same spirit of [22, 11]; 2) Our proposed motionlet approach where clustering is based on motion and 3) our motionlet approach using large-scale feature selection. Figures 7a and 7b show the ROC curves for crowded and low-activity scenes, respectively. Clearly our approach based on motionlets offers significant improvement in terms of accuracy over the baseline. In addition, large-scale feature selection allows the classifier to be sparser as shown in Figure 7c. Sample detection results for several cameras can be seen in Figure 8. For a better feeling of how our method works, we refer the reader to a video demo at: `http://rogerioferis.com/ICMR2011/60HzVehicleDetectionCrowd.wmv`

This particular sequence was obtained from New York Department of Transportation (DoT). It has low-frame rate and high volumes of activity. Our method captures most of the vehicles while running at more than 60Hz. Another video demo showing the detection of multiple vehicle types in crowded conditions can be seen at: `http://rogerioferis.com/ICMR2011/TruckCarsDetectionCrowd.wmv`

Color retrieval was assessed on the DoT sequence above from which we automatically captured 2550 vehicles. Figure 9a shows an example of search for yellow vehicles. Note that in this case all retrieved events are taxi cabs. Figure 9b shows the confusion matrix and the initial distribution of vehicle colors. The percentage of images correctly classified is 96.76%. The average classification rate is 90.72%.

Evaluation of measurements such as width, height, length, and speed is not trivial as it is difficult to obtain the ground truth. So we provide a simple qualitative analysis by looking at search results associated with the query "search for vehicles with length larger than 18ft", using the same DoT camera used for color retrieval over a period of 30 minutes. A total of 360 events were retrieved. Figure 10 shows sample results for this search, including the timestamp and length information associated with each icon. By visually looking at the results, discriminating large cars from small cars (clearly above or below 18 ft), we get 314 large vehicles out of the 360 events that were returned by the search, thus obtaining 87% classification accuracy. The classification errors usually come from false vehicle detections or wrong bounding box estimation. Note that traditional surveillance systems which offer object size search based on background/motion segmentation fail to handle crowded scenes as multiple small objects close to each other are merged and considered as a single large object. The video below shows a demonstration of our vehicle dimensions estimation in world coordinates: `http://rogerioferis.com/ICMR2011/WorldCoordinates.wmv`

Discussion. The robustness of our vehicle detection method to crowded scenes and different lighting conditions comes from the data itself - our training dataset contains many images of vehicles occluding each other and covers many different weather conditions. The ability to handle multiple vehicle types comes from our shape-free appearance learning and varying shape sliding window. Although different types of vehicles may differ in appearance, they also have many similar structures, which facilitates learning. The proposed motionlet approach improves accuracy by clustering samples based on motion rather than appearance and also the efficiency, as detectors are directly selected based on motion information. The motion clusters are determined automatically, contrasting with other view-based methods that need manual pose annotation. Large-scale feature selection fur-

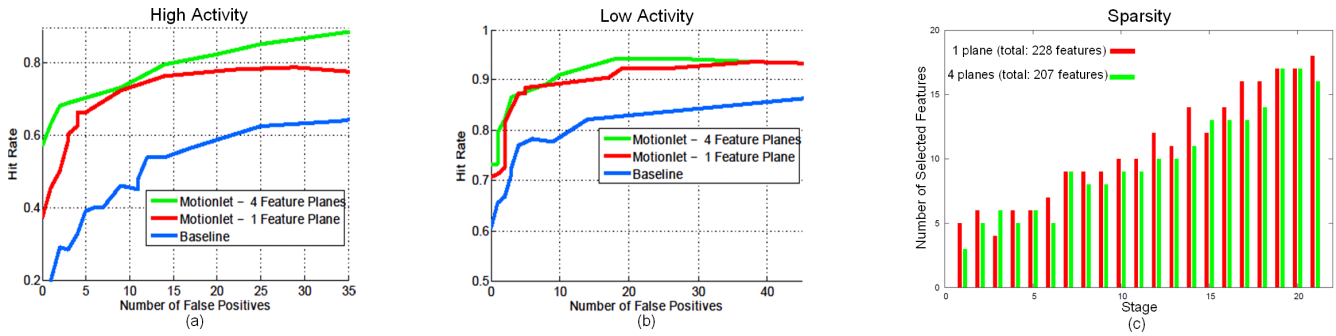

Figure 7: Comparison of our motionlet approach with a s tree-based Adaboost learning algorithm which uses clustering based on appearance (baseline). (a) Results on crowded scenes. (b) Results on low-activity scenes. (c) The motionlet detectors are sparser and therefore more efficient with large-scale feature selection.

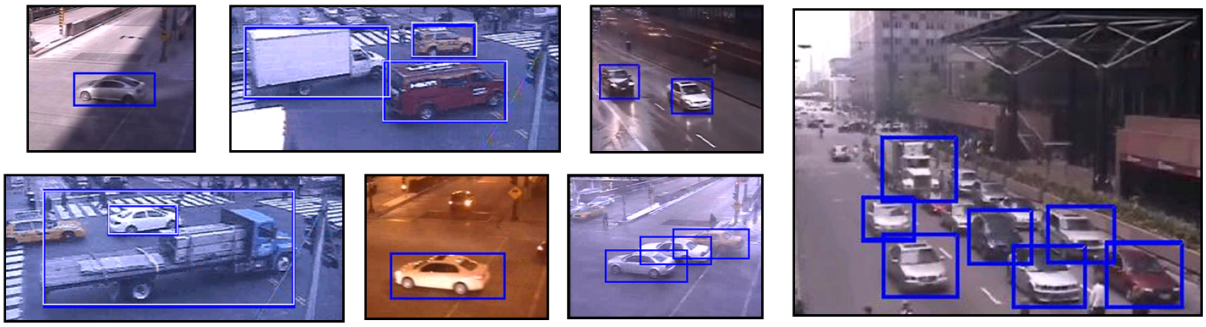

Figure 8: Sample detection results in challenging images including crowds, multiple vehicle types, different lighting conditions and environmental factors such as rain, reflections, and shadows.

ther improves accuracy and sparsity. We have noticed that several false detections occur in parts of vehicles, so adding those to the negative training set should improve results. Other false alarms could be pruned by using calibration information, i.e., pruning hypothesis that do not satisfy expected vehicle sizes at specific image positions. Our color attribute estimation could be improved by using learning methods to better deal with different lighting conditions.

6. CONCLUSION

We have presented a novel application for attribute-based vehicle search in urban surveillance scenarios. Our approach relies on a novel and robust vehicle detection method, followed by attribute extraction, tranformation of measurements into world coordinates, and database ingestion/search. As future work, we plan to extend the set of attributes used in our implementation, including other visual features and information from non-visual sensors.

7. REFERENCES

[1] C. Anagnostopoulos, I. Anagnostopoulos, I. Psoroulas, V. Loumos, and E. Kayafas. License plate recognition from still images and video sequences: A survey. *IEEE Transactions on Int. Transp. Systems*, 9(3), 2008.

[2] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3D human pose annotations. In *ICCV*, 2009.

[3] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *IEEE Transactions on PAMI*, 23(6), 2001.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[5] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

[6] P. Felzenszwalb and R. G. andD. McAllester. Cascade object detection with deformable part models. In *CVPR*, 2010.

[7] R. Feris, J. Petterson, B. Siddiquie, L. Brown, and S. Pankanti. Large-scale vehicle detection in challenging urban surveillance environments. In *WACV*, 2011.

[8] A. Hampapur, L. Brown, R. S. Feris, A. Senior, C. Shu, Y. Tian, Y. Zhai, and M. Lu. Searching surveillance video. In *AVSS*, 2007.

[9] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, 2009.

[10] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.

[11] R. Lienhart, L. Liang, and A. Kuranov. A detector tree of boosted classifiers for real-time object detection and tracking. In *ICME*, 2003.

[12] X. Ma, W. Eric, and L. Grimson. Edge-based rich representation for vehicle classification. In *ICCV*, 2005.

[13] S. Merler, B. Caprile, and C. Furlanello. Parallelizing AdaBoost by weights dynamics. *Computational Statistics & Data Analysis*, 51(5):2487–2498, 2007.

(a)

	Black	Blue	Red	White	Yellow	Green
Black	**0.9531**	0.0028	0.0058	0.0382	0.0000	N/A
Blue	0.3667	**0.6333**	0.0000	0.0000	0.0000	N/A
Red	0.0278	0.0000	**0.9722**	0.0000	0.0000	N/A
White	0.0206	0.0010	0.0000	**0.9774**	0.0010	N/A
Yellow	0.0000	0.0000	0.0000	0.0000	**1.0000**	N/A
Green	N/A	N/A	N/A	N/A	N/A	N/A
Distribution	856	21	27	807	839	0

(b)

Figure 9: Color retrieval results. (a) example results for a yellow car search. Note that for this particular scene all taxi cabs have yellow color. (b) Quantitative results - confusion matrix and color distribution for the captured vehicles.

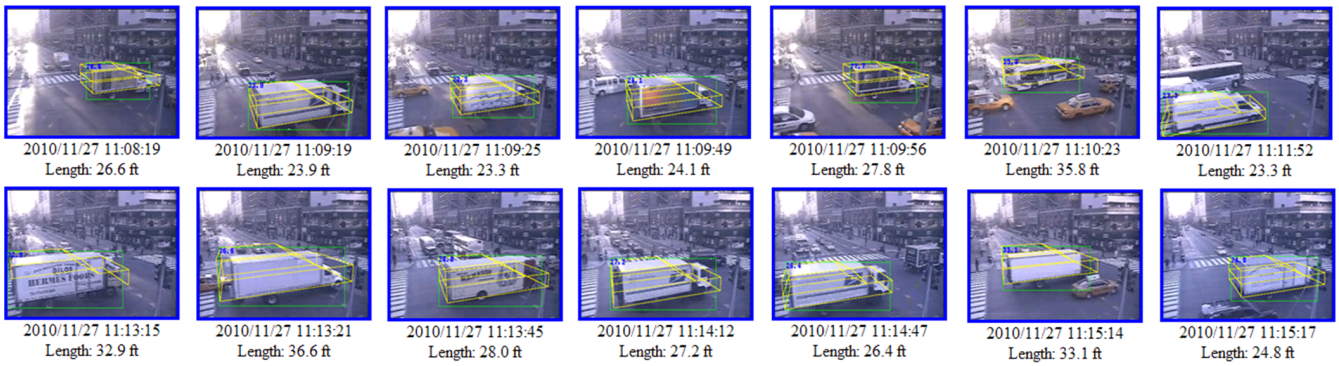

Figure 10: Example of a search for large vehicles (larger than 18ft). Our vehicle detection method provides precise bounding boxes for multiple types of vehicles. Real vehicle dimensions in world coordinates are then obtained through a calibration and 3D model fitting process.

[14] P. Perez, M. Gangnet, and A. Blake. Poisson image editing. In *SIGGRAPH*, 2003.

[15] J. Prokaj and G. Medioni. 3D model based vehicle recognition. In *WACV*, 2009.

[16] O. Russakovsky and L. Fei-Fei. Attribute learning in large-scale datasets. In *ECCV 2010 Workshop on Parts and Attributes*, 2010.

[17] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1998.

[18] Y. Tian, M. Lu, and A. Hampapur. Robust and efficient foreground analysis for real-time video surveillance. In *CVPR*, 2005.

[19] D.-C. Tseng and C.-H. Chang. Color segmentation using ucs perceptual attributes. In *Proc. Natl. Sci. Council: Part A*, volume 18, pages 305–314, 1994.

[20] D. A. Vaquero, R. S. Feris, D. Tran, L. Brown, and A. Hampapur. Attribute-based people search in surveillance environments. In *WACV*, 2009.

[21] P. Viola and M. Jones. Robust Real-time Object Detection. In *International Journal of Computer Vision*, 2001.

[22] B. Wu and R. Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. In *ICCV*, 2007.