Bus Detection for Adaptive Traffic Signal Control

Aravindh Mahendran
Carnegie Mellon University
Pittsburgh, PA - 15217
amahend1@andrew.cmu.edu

Stephen Smith (PI)
Carnegie Mellon University
Pittsburgh, PA - 15217
sfs@cs.cmu.edu

Martial Hebert
Carnegie Mellon University
Pittsburgh, PA - 15217
hebert@ri.cmu.edu

Xiao-Feng Xie
Carnegie Mellon University
Pittsburgh, PA - 15217
xfxie@cs.cmu.edu

January 6, 2014

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

List of Figures

List of Tables

Executive Summary

The ability to detect buses in oncoming traffic in real-time offers unique opportunities to improve overall traffic flow in urban environments. Buses regularly disrupt traffic flow as they pickup and discharge passengers. Yet, if traffic flows at a given intersection are not simultaneously blocked in multiple directions, there are often traffic signal control decisions that can be taken adaptively to minimize these disruptive effects (e.g., by servicing cross traffic) and reduce overall traffic congestion. Existing adaptive traffic signal control systems do not attempt to recognize and act upon the presence of buses in incoming traffic streams. Alternatively, existing approaches to bus prioritization start from the assumption that bus movement trumps all other vehicles, give no attention to how disruptive it is to overall traffic flow to keep buses moving, and relies on additional hardware, both within the vehicles and at each intersection.

This report describes progress made toward the development of the ability to use video streams from commercial traffic cameras to detect the presence of buses in real-time and to incorporate this information into an adaptive traffic signal control scheme. We focus specifically on analysis of video camera frames produced by the Traficon cameras mounted at the East Liberty pilot site of the Surtrac adaptive traffic signal system [26] and on the use of detected bus presence information within the Surtrac system. Surtrac implements a novel, decentralized approach to adaptive signal control that is designed for urban (grid) road networks. It provides real-time (second-by-second) response to observed traffic flows at individual intersections [34] while exploiting communication together with simple coordination protocols to achieve synchronized network behavior [32]. The initial Surtrac field test in East Liberty has demonstrated substantial performance improvements over the pre-existing signal control scheme. Its real-time nature makes it ideally suited for incorporating real-time information about the presence and movement of buses.

Our approach to bus recognition builds on recent computer vision research in exemplar-based support vector machines (SVM) recognition [23]. This work has produced an general object recognizer that we specialize to achieve real-time detection by exploiting domain specific constraints (e.g., fixed viewing angle, road boundary information, and anticipated scale of the vehicles). The challenge is to be robust to varying illumination and weather conditions, occlusions from other vehicles, and large variations in scale while producing recognition results in real-time.

To train and test the developed bus recognizer, a set of training samples (images) were collected from current video streams at the Penn Avenue and Highland Avenue intersection. A subset of these images were first used to develop a set of bus recognition exemplars. Subsequent evaluation of these exemplars on the remaining image subset has shown near 100% accuracy for recognizing buses close to intersections, with degradation as the distance to the intersection increases. These results span a number of different observation conditions (e.g., rain, snow, night, etc.). In parallel we have analyzed an extension to the Surtrac adaptive signal system designed for reacting to the detection of a bus that is stopped discharging or taking on passengers by shifting to servicing cross traffic in this situation. Using a microscopic simulation of the pilot test area, results show an overall reduction in vehicle travel times through the test site of 3-6% while simultaneously reducing bus travel times.

Although these results are promising, they nonetheless represent ”laboratory” results, and current work is aimed at investigating how they translate to the field. The first step will be to install the recognizer at the Penn Avenue/Highland Avenue intersection that our training data was extracted from (using a dedicated processor), and test

its ability to detect stopped bus events in isolation. If performance is determined to be acceptable, we will address hardware and software integration issues with Surtrac, and focus on realizing the above observed throughput benefits. Finally, further extensions will be made to the Surtrac intersection scheduler to further improve traffic flows by additionally incorporating bus prioritization actions in relevant situations.[1]

1 Introduction

Increasingly, traffic signal control systems are incorporating vision-based vehicle detection. In simple *actuated control settings*, the detection of waiting vehicles at an intersection can be used to trigger green time on demand for side street traffic. In more sophisticated *adaptive* traffic signal control systems, cameras are used to sense approaching traffic volumes in all directions and enable dynamic allocation of green time to various signal phases to maximize overall vehicle throughput. However, the ability of adaptive traffic control systems to optimize vehicle flows is limited by the detection capabilities of current commercial technologies. Current commercial video detection systems allow vehicle presence detection and counting based on movement through predefined zones in a video captured from a stationary camera [30], but they do not provide an ability to distinguish between different types of vehicles, in particular between buses and trucks. This distinction is important because buses have a distinct and often disruptive traffic flow pattern that could be exploited if they could be recognized. In current practice, bus detection is possible only through onboard signal transmitters and infrastructure receivers, and these systems are typically only used to implement transit priority.

In this report, we summarize research aimed at recognizing buses from video streams in real-time, and incorporating this information more broadly into adaptive traffic signal control decisions. The starting hypothesis for this work was that by exploiting various domain constraints to simplify the recognition problem, recently developed computer vision techniques for general object recognition can be adapted to effectively detect buses in real-time. As we describe below, we have developed a prototype bus recognizer based on this approach that performs quite well on test images and will be field tested shortly. We have also performed preliminary analysis of the benefit that can be gained by an ability to recognize buses that are stopped discharging or taking on passengers and blocking traffic flow. By imposing a green delay and servicing cross traffic in this situation, we have shown (in simulation) that the Surtrac adaptive signal control system [26] can achieve a 3-6% reduction in the average vehicle travel times while also decreasing bus travel times [33].

The remainder of the report is organized as follows. In Section 2, we first overview recent related work in object detection and bus detection. This is followed in Section 3 with a description of the hardware system setup that is assumed, including both the limitations of the target computing platform and the set of images extracted from a Traficon camera at Penn Avenue and Highland Avenue to serve as a dataset for training and evaluating the bus recognizer. In Section 4, we then briefly review the exemplar SVM algorithm that provides the basis of our approach and detail the various modifications we have made to boost its performance. Section 5 next analyzes the performance of our system on the extracted dataset. In Section 6, we shift gears and assume that we have the ability to detect the presence of buses, specifically when stopped at a bus

[1] This report draws from material that originally appeared in [21, 33]

stop and blocking traffic. We describe the above mentioned heuristic for delaying the corresponding green phase in this case, and present initial simulation results indicating the impact on performance. Finally in section 7 we summarize lessons learned and outline next steps.

2 Related Work

General object detection is a very widely studied topic in computer vision. Recent methods range from using a large number of templates as in the case of ensemble of exemplar SVMs [23], to fewer but structured templates with a deformation model to help express a large set of configurations [12], to a single template as in correlation filters like Maximum Margin Correlation Filters for Vector data (MMCF) [3]. These have shown great improvements for object detection in natural images. This suggests that object detection might be used practically for domain specific data such as traffic intersections. Some of the previous work in this domain has looked at general vehicle detection and classification [4, 7, 6]. These are capable of bus detection but are not tuned for this purpose. Their effort is orthogonal to ours in that we discuss how constraints specific to bus detection can be used to make a general object detection algorithm run fast and have high accuracy. Furthermore, bus detection at traffic intersections using computer vision methods has been studied in more constrained settings such as near the intersection or under good visibility and normal weather conditions [31, 13, 20, 28]. In some cases they also assume knowledge of camera parameters. It is unclear whether these approaches would be able to handle the large variation in scale present when dealing with both near and far buses while being able to run within a reasonable time frame on our computing platform.

Bus detection has been studied with several other sensing modalities such as GPS, RFID and loops placed underground in the detection zone [17, 11, 29]. These require additional equipment on the bus or at the intersection or both. On the other hand, our approach aims to use the computing infrastructure and video camera setup already installed for adaptive traffic control.

An application closely related to ours in that of vehicle and pedestrian detection for autonomous driving [8, 27, 5, 9, 19, 18, 16]. The autonomous driving application is, however, more general due to the articulated nature of pedestrians and fast moving cameras. They do not exploit the constraints available in our data. Also modern systems for this application typically have less noisy cameras with much larger computing power carried by the autonomous vehicle.

Inspired by the developments in general object detection and their successful application to autonomous driving, we propose to use the Ensemble of Exemplar SVM algorithm (ESVM) [23, 22] and then compare its performance with the mixture of deformable parts based model [12]. The structure in our data enables us to adapt these algorithms to both improve detection speed and average precision over novel test images. We have achieved 0.74 average precision using an adaptation of the exemplar SVM algorithm that runs at around 1.37 seconds per frame on the limited computing power available in our practical system. The system is able to detect some tiny buses as shown in Figure 1a and even some occluded buses (see Figure 1b).

(a) (b)

Figure 1: Qualitative results. (a) A tiny bus detected with score 1.000. (b) An occluded bus detected with score 1.000 shown in green and a false positive with score 0.926 shown in red. Figure 7 suggests that this is a very low false positive score.

3 System description

The system of interest consists of a camera placed over the traffic signal lights, looking down the road. The camera is connected to a computer placed inside the traffic control cabinet. The computer runs the bus detection algorithm and reports detections to the traffic scheduler, which in our case is assumed to be the Surtrac adaptive signal control system. As described in [26], Surtrac also runs on a separate computer in the cabinet at a given intersection (which ultimately will be the same computer running the bus detection algorithm), and issues calls to the hardware controller running the intersection that indicate whether to stay in the same green phase or shift to another.

3.1 The Data

For our purposes in this research, we consider a single traffic intersection. The Traficon camera of interest provides 640×480 RBG images at upto 30 frames per second. We are, however, aiming for only 0.5 to 1 frames per second for a traffic scheduling application. In other words, two seconds per frame is chosen as an upper bound for an algorithm's detection time for it to be used in the system. Our system is required to detect buses coming up this road and should ignore buses going down or laterally crossing the intersection. We call this category as "busfront". The system is required to operate throughout the day and in all weather conditions. We annotated data collected during a sunny afternoon, snowy winter night and a rainy evening. Each of these three image collections were temporally divided into training, validation and test data. This is referred to as the All Weather dataset. A single ensemble of SVM and Mixture of deformable parts based model was trained on the entire training + validation data.

To give the reader a feel for the challenges presented by this data we have provided three examples in Figure 2. We find occlusion, illumination change, large variation in scale and large amount of noise as the major obstacles for our outdoor application.

Table 1 and Table 2 summarize relevant statistics for this dataset. Near and far here refer to buses near the intersection and far from the intersection respectively. Nearness

	Training	Validation	Test
# images	2714	845	1293
# buses (near)	352	279	384
# buses (far)	267	259	272

Table 1: All Weather dataset statistics - Dataset size.

	Noon	Evening	Snowy night
# images	483	386	424
# buses (near)	143	155	86
# buses (far)	147	67	58

Table 2: All Weather dataset statistics - Number of buses and images in the three weather conditions in the test set.

to the intersection is determined by the upper y coordinate of the bounding box. An upper y coordinate less than 80 corresponds to a bus far from the intersection while an upper y coordinate more than 80 corresponds to a bus near the intersection. This boundary line is illustrated in Figure 2.

3.2 Computing Platform

The application targeted in this report is bus detection at traffic intersections. We believe that a decentralized solution, where the detection algorithm runs at the intersection, is more scalable than a centralized server processing images from several intersections. Our solution for "busfront" detection should therefore be able to run reasonably fast on a computer that can be deployed at this intersection at a low cost. Such a computer has to be of a small form factor and should withstand extreme heat and extreme cold. We use the EPC-2020 fanless computer with the D525 Intel Atom processor. This processor features a dual core intel atom with each core at 1.8Ghz with hyper-threading. A small cache size of 1MB influences the run time greatly. We use 2GB of DDR3 memory in this computer. It is important to note that it is not equipped with a GPU nor does the D525 have a OpenCL capable graphics chip leaving only the CPU for object detection processing.

Figure 2: The dataset is challenging because of noise, large change in illumination, heavy occlusions and large variation in object size in image. The yellow line indicates the boundary (based on the upper y coordinate) between buses near the intersection and buses far from the intersection.

4 Our Approach - Ensemble of Exemplar SVM + Domain Constraints

We detect buses in individual images independently. The bus detection procedure searches through boxes in the image trying to find boxes containing buses. This is followed by a post processing step called local maxima suppression that removes redundant detections. The above is commonly referred to as the sliding window approach and is a very popular approach used by object detection methods.

We choose the exemplar SVM (ESVM) algorithm to decide bus presence/absence inside each box. This choice is made because the simplicity of the ESVM algorithm enables us to take advantage of domain constraints. We first briefly review the ensemble of exemplar SVM algorithm (ESVM) in the context of binary classification.

The ESVM algorithm learns a bag of hyperplanes. Each hyperplane separates a fixed positive training example from all the negative training examples. It then classifies a test example as positive if any of the hyperplanes classify it as positive, negative otherwise.

Each hyperplane is trained as a Support Vector Machine[2] using hard negative mining[12]. Each hyperplane is identified by the one positive example used to train it. The ensemble is effectively a max pool of these independently trained hyperplanes. If w and b are the hyperplane slope and offset, then $w^T x + b$ is the SVM confidence score for bus presence.

These hyperplanes are calibrated to suppress those which are performing badly and strengthen those which are performing well on the training data. Platt calibration [24] is generally used for this purpose.

The hyperplane is learnt over gradient based image features called Histogram of Oriented Gradients [10]. These features are popularly used for object detection and capture local gradient properties and normalize them for illumination invariance.

We next discuss how domain constraints were incorporated into this framework.

4.1 Not a General Object Detection Problem

The exemplar SVM algorithm was developed for general object detection in natural images from the internet. It incorporates techniques such as multi-scale HOG pyramid and add flip detection (reflect the image horizontally and repeat the detection process) to handle novel object sizes (unseen in training data) and view point change. In its general form it is computationally very expensive and runs at almost 200 seconds per frame on our test machine. To increase detection speed we exploit the fact that all possible object sizes are already captured in the training data. Thus the multi-scale HOG pyramid can be replaced by single scale HOG features. In other words, each exemplar searches for buses of its own size. Removing the HOG pyramid increases the average precision by reducing the number of false positives. It reduces the computation time to around 15 seconds per frame.

Similarly, the add flip detection trick is unnecessary for our problem as the camera is stationary with respect to the road. Disabling it gives almost 2X boost to detection speed but does not significantly affect detection accuracy.

4.2 Exploiting the Stationary Camera

The stationary camera can be exploited further to hand pick the region of interest (road lane) in the image using a simple crop operation. It also yields a pattern between the y coordinate and the bus size. Small buses occur at the top part of the region of interest, while large buses occur at the bottom. We use this information by dividing the region of interest into two zones - near zone (zone 1) and far zone (zone 2). An exemplar is chosen to fire inside the far zone if and only if the corresponding positive example belongs to that zone in the training image. If not, it fires inside the near zone. We notice that small buses operating on zone 2 correlate much faster with the HOG image due to their small size, while large buses operating on zone 1 correlate very slowly but still faster when compared to correlating all the examples on the entire lane. This division also eliminates a few false positives because perspective distortion is the only reason for change in bus size in our dataset. Small buses are always found at the back end of the road (zone 2) while large buses are always found near the intersection (zone 1). The drop in recall due to this division is small if we provide for a small overlap between zone 1 and 2. The zones are marked by hand and made constant. At this stage the algorithm takes about 4.3 seconds per frame on our computing platform.

4.3 Randomly Sampling Models Per Zone

The correlation of exemplar models with the HOG image is the most expensive step in the detection pipeline. The small cache size on our computer makes the matrix multiplication form of this correlation more expensive than a multi threaded correlation popularly used by both Exemplar SVM (for small number of exemplars) and DPM. This operation requires computing time which increases with the number of exemplar models. We therefore randomly sample a fixed number of models per zone to fit within any time constraints imposed by the traffic scheduling algorithm. We analyze the impact of sampling 100 to 200 models per zone. The result of randomly sampling models is a decrease in average precision and a significant increase in detection speed.

4.4 Weibull Calibration

The magnitude of noise in the data makes it hard to provide good ground truth annotations. This noise combined with the small bus sizes means that some exemplars will generalize very poorly. This makes the calibration step very important as it can suppress the bad exemplars and strengthen the good ones. Platt calibration [24] is generally used to calibrate exemplar SVMs. It fits a sigmoid over points $(x, 1 - \epsilon_+)$ for all detection scores x corresponding to true positive detections, and (x, ϵ_-) for detection scores corresponding to true negative detections. We find that exemplars corresponding to small buses are suppressed more heavily than required leading to poor performance on test data for buses farther away from the intersection. Recent work has proposed extreme value theory based calibration [25] which only requires scores from the non match distribution (negative SVM scores). These are the negative scoring points and are used to fit a weibull distribution. The cumulative density function of the fit distribution is used to map SVM scores to probability values. The intuition behind this approach is that the maximum, of a set of points sampled *iid* from a distribution, is a random value drawn from one of three extreme value distributions. If the random variables are bounded then the maximum is a draw from a weibull distribution. This was first applied to SVM calibration in [25]. We attempt to use it because, unlike Platt cali-

Approach	Average Precision	Run Time per frame (sec)
ESVM+Domain	0.840	**4.269 ± 0.547**
DPM - 12 components	0.861	Out of Memory
DPM - 8 components	**0.882**	Out of Memory
DPM - 4 components	0.836	75.992 ± 0.4365
MMVCF	0.715	39.405 ± 0.126

Table 3: Comparison with DPM[12] and MMVCF[3] - Average Precision and Run time on EPC-2020.

Approach	Average Precision	Run Time per frame (sec)
ESVM+Domain	**0.840**	4.269 ± 0.547
No Zones	0.837	5.739 ± 0.599
No Zones and No Lane Cropping	0.804	15.103 ± 0.501
No Calibration	0.783	4.169 ± 0.427
Platt Calibration	0.725	4.089 ± 0.366
Multiscale HOG pyramid	0.813	29.954 ± 1.373
With Image flip and detection	0.837	8.033 ± 0.658
Sampling 100 models per zone	0.732 ± 0.007	**1.369 ± 0.130**
Sampling 200 models per zone	0.829 ± 0.004	2.627 ± 0.386

Table 4: Impact of domain constraints - Average Precision and Run time on EPC-2020.

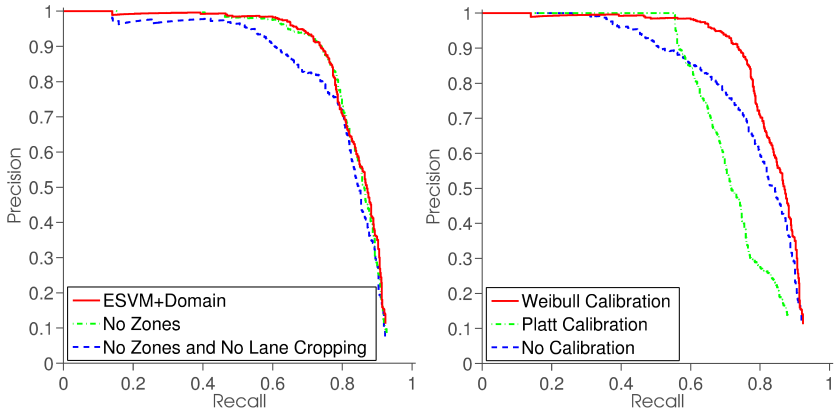
bration, it only requires negative SVM scores available in plenty in our case. We force negative SVM scores to have a lower bound by discard the scores less than -1.1. This extreme value theory based calibration method (referred to as "weibull calibration") greatly improves performance farther away from the intersection.

5 Performance Analysis

The performance of our system was quantitatively evaluated in laboratory where we trained on the training images and tested on the test images of the all weather dataset. We tested the impact of each of our modifications on this dataset and also compared our performance with that of the Deformable Part based Model (voc-release5) object grammar based object detection [14] [2] and the maximum margin correlation filter on vector data (MMVCF) [3]. We are interested in achieving high precision for moderately high recall so as to help the traffic scheduler but not mislead it. This fact will later be used in selecting a threshold based on score analysis.

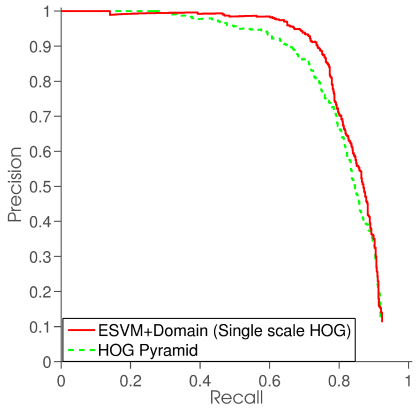
In this experiment "ESVM+Domain" refers to the Exemplar SVM algorithm op-

[2] voc-release5 deformable parts model code had to be modified to handle the large scale variation in our dataset. We added two extra octaves for detection. We further let the algorithm discard training images smaller than 1500 pixels when compared to 3000 pixels used originally. The later is necessary to learn a small enough model such that two extra octaves will enable it to scale down to the smallest buses in our dataset. In the absence of this modification recall for buses far from the intersection is zero . We also crop the lane to make the comparison with our approach fair.



(a) The effect of cropping and dividing lane into zones.

(b) Comparing Weibull calibraion, no calibration and Platt calibration.



(c) Effect of removing the multiscale HOG pyramid.

Figure 3: Precision Recall for Laboratory Experiments. "ESVM+Domain" is Exemplar SVM operating on a cropped image, with two zones, without image flip add detection, without model sampling, without a HOG pyramid, with weibull calibration. "No Zones" is the same except that the image is not divided into two zones. "No Zones and No Lane Cropping" is the same as "No Zones" except that the image is not cropped to focus on the lane. "No Calibration" is the same as "ESVM+Domain" but does not use any form of exemplar calibration, while "Platt Calibration" uses Platt Calibration instead of Weibull Calibration. "Multiscale HOG pyramid" is "ESVM+Domain" using a multiscale HOG pyramid instead of single scale HOG features.

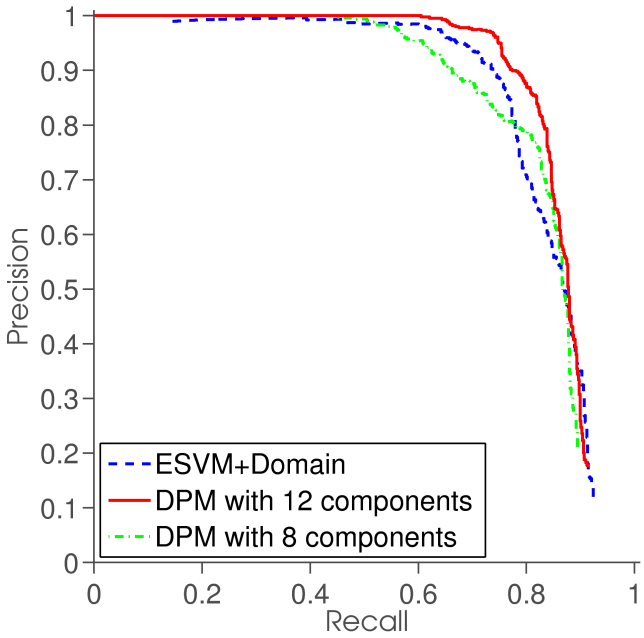


Figure 4: Comparison between ESVM+Domain and DPM (12 components and 8 components).

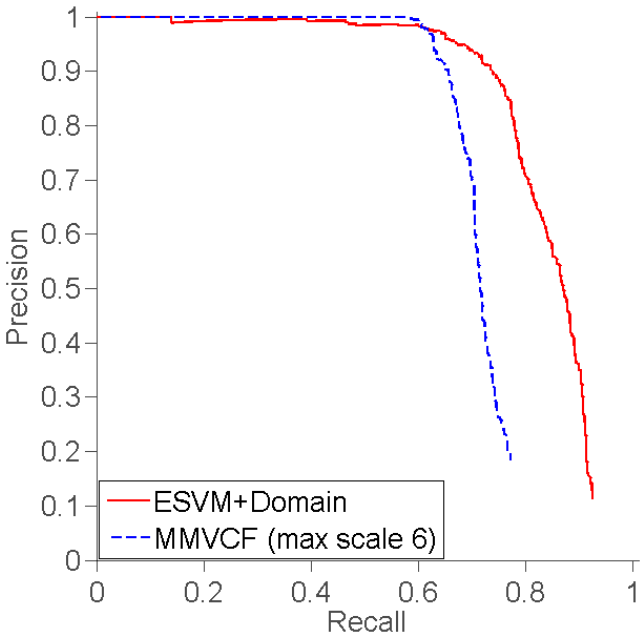


Figure 5: Comparison between ESVM+Domain and MMCF on vector data.

erating on the predefined region of interest (start (185,15) to end (450,315)), using 2 zones without the multi scale HOG pyramid and without the "detect add flip" trick. It uses all models in each zone without random sampling. The zones are defined using their y coordinates. The near zone extends from 60 pixels to 315 pixels. The far zone extends from 15 pixels to 100 pixels.

Figure 4 compares ESVM+Domain and DPM with 8 and 12 components. 12 components performed best on the validation dataset but we also illustrate the result for 8 components to signify that this parameter is not very critical for accuracy. Average precision and runtime per frame for DPM are shown in Table 3. We observe that DPMs perform a little better than ESVM+Domain but are unable to run on our targeted computing platform due to small memory limits. DPM with fewer mixture components is able to run on our computer but is slower than the targeted 2 seconds per frame detection speed. Arguably, several modifications could be made to the voc-release5 implementation akin to those described in this paper to achieve similar detection speeds. But we used the ESVM instead as its simplicity makes these modifications more intuitive and easy.

Figure 5 compares ESVM+Domain and MMVCF. MMVCF uses the same HOG features used for ESVM+Domain and DPM. We trained the MMVCF with parameters $\alpha = 1e-3, \beta = 1-\alpha, C = 1$ and a positive bias of 10. The resulting template was used to detect images using sliding window search without add flip detection, using a multi scale HOG pyramid with min scale 0.01 and max scale 6. The large value of max scale is required to detect small buses in our dataset at the cost of a very slow detection speed. To make the comparison with our approach fair we provide MMVCF with the region of interest discussed in section 4.2. Our method outperforms correlation filters. We believe that this is because of the large variation in bus sizes in our dataset as correlation filters perform comparably for buses near the intersection (Average precision 0.95 for correlation filters when compared to an average precision of 0.98 for ESVM+Domain) but poorly for buses far from the intersection (Average precision 0.28 for correlation filters when compared to 0.59 for ESVM+Domain).

We next examine the impact of the several domain constraints incorporated in this paper. Figure 3 show the precision recall curve for ESVM+Domain and with each of the modifications removed. This analysis helps understand the impact of each domain constraint in the final system. Table 4 contains the runtimes per frame and the average precision for all the cases discussed in Figure 3. In particular, removing the HOG pyramid gives an enormous boost in runtime with no compromise in accuracy. ESVM+Domain, without the HOG pyramid, is able to retain high precision with significant increase in recall relative to ESVM+Domain with the HOG pyramid. Dividing the image into two zones does not harm average precision and contributes to an increase in speed. The loss due to sampling models is not terrible and the gains in computing time are significant. Figure 6 shows two consecutive frames in our dataset with the bus in exactly the same position. Sampling models leads to the bus being missed in one case but detected in another. Disabling the "detect add flip" trick is virtually harmless. Weibull calibration performs better than Platt Calibration. Infact, we find that not calibrating the hyerplanes performs better than using Platt calibration. This might be due to Platt calibration overly suppressing some exemplars making their scores go below that of overlapping false detections from other exemplars.

We further analyze the laboratory results by dividing the data into two categories - Near the intersection and far from the intersection [3]. The precision recall curves (see

[3]This usage of near the intersection and far from the intersection is not the same as that used in section

Figure 6: Effect of sampling. The same bus detected in one case but missed in another.

Figure 8) for these show that we are close to a 100% accurate near the intersection but are doing much worse farther away. This division into two categories also gives insight for the difference in performance between Weibull Calibration and Platt Calibration (see Figure 9). Near the intersection, Weibull calibration scores an average precision of 0.98 while Platt calibration scores 0.96. On the other hand, far from the intersection Weibull calibration scores 0.59 against 0.10 for Platt Calibration. The comparison further illustrates that for buses far from the intersection weibull calibration is able to maintain precision more than 0.8 for recall around 0.4 which is important to avoid confusing the traffic scheduler with a large number of false positives. The difference in performance between these two calibration techniques is explained by noticing that Weibull calibration only requires negative scores which are obtained easily by correlating the model with background data. Platt calibration requires a few positive and a few negative detections. Positive detections are hard to obtain for exemplars corresponding to small buses far from the intersection. These exemplars are therefore not correctly calibrated by Platt calibration.

Based on the performance analysis presented in this section, we decide to use *ESVM+Domain+100 models per zone* as it provides a good tradeoff between detection time and detection accuracy. More importantly it runs within the 2 seconds per frame time limit mentioned in section 1. We next quantify the behavior of this approach in the three weather conditions constituting our All Weather dataset. Table 5 measures their average precision averaged over 10 runs on buses near the intersection. We restrict our study to buses near the intersection so that differences is visibility range (which influence the ground truth annotations far from the intersection) do not bias the results. We interpret these quantitative results as suggesting that our system is reasonably effective in all the three weather conditions.

We next present an analysis of the scores generated for true positives and false positives. This is important to pick a good threshold for the exemplar SVM scores. The score histogram for true positives and false positives is shown in figure 7. This suggests that the high score for the false positive shown in figure 1b is not a big problem. We can maintain high precision with moderately high recall using a conservative threshold of 0.98. This generates the detections shown in figure 10. We are able to detect occluded buses, small buses, buses in bad lighting conditions while avoiding false positives due

4.2, but matches the usage in table 1.

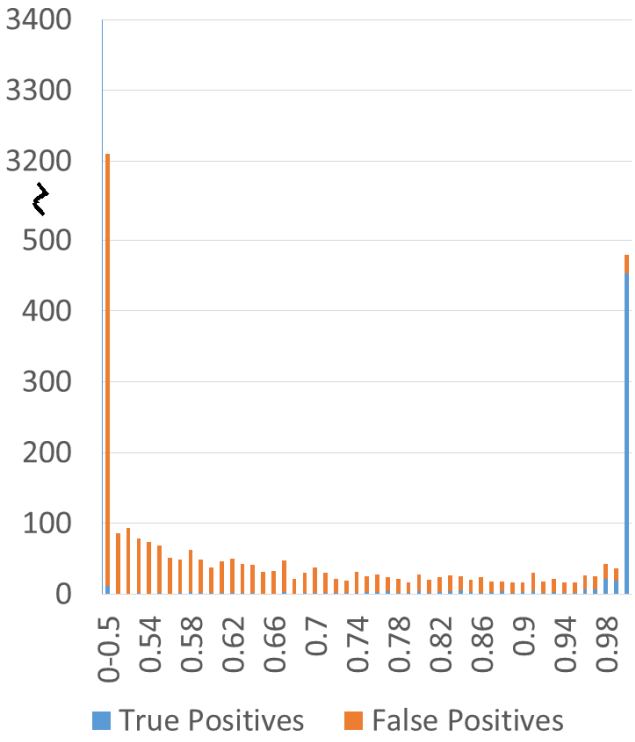


Figure 7: Score Analysis. Blue histogram bars correspond to true positives and orange histogram bars correspond to false positives.

Weather type	Average Precision
Noon	0.92 ± 0.01
Evening	0.97 ± 0.01
Snowy night	0.96 ± 0.01

Table 5: Performance across different weather conditions on buses near the intersection.

to trucks and crowded streets. The system makes a mistakes in the form of missed detections either due to sampling 100 models, the conservative threshold of 0.98 or tough cases such as very small buses, out of plane rotations and heavy occlusion. We also see some false positives due to cars/pairs of cars and occasionally due to trucks. Failure cases are shown in figure 11.

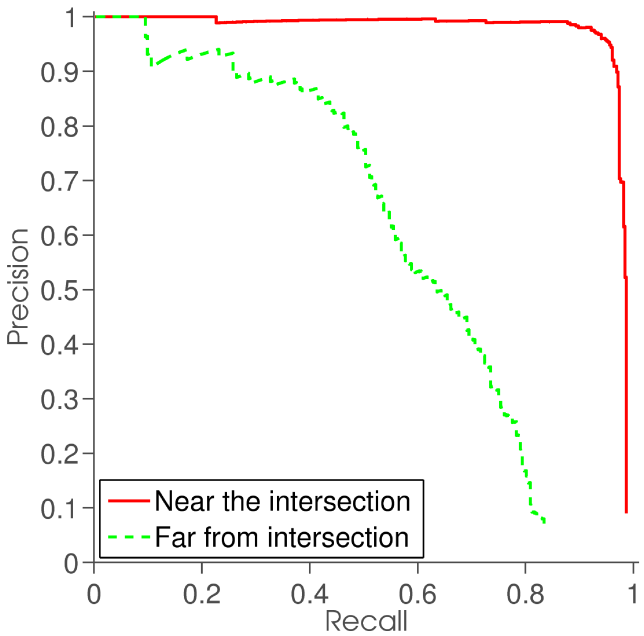


Figure 8: Detection performance near the intersection and far from the intersection for "ESVM+Domain"

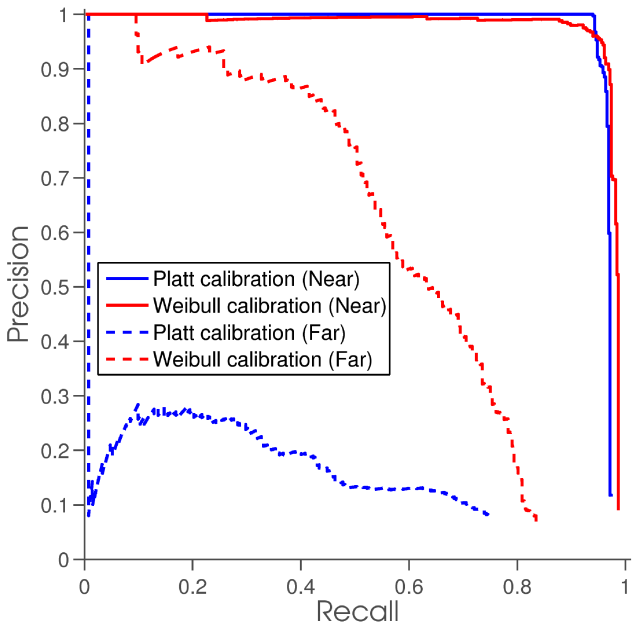


Figure 9: Platt Calibration vs Weibull Calibration - Difference in performance at different ranges

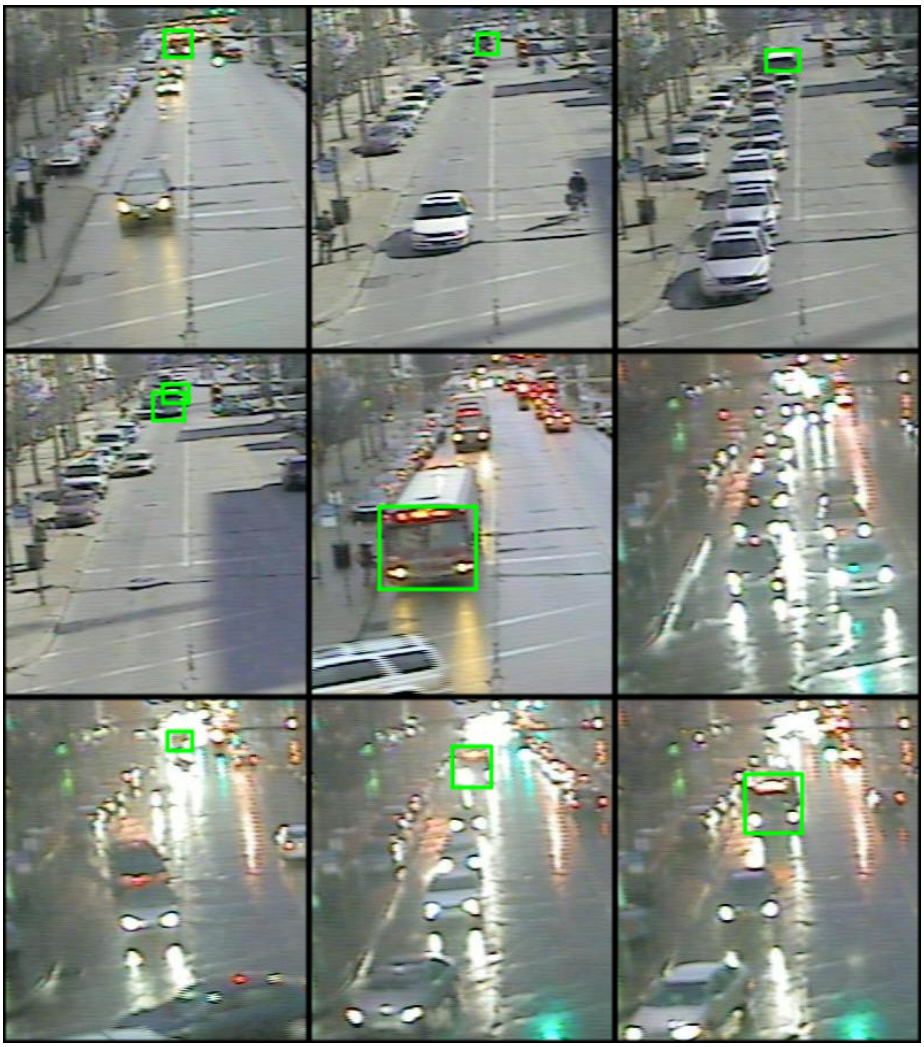


Figure 10: Good detections by our system with threshold 0.98. It is able to detect small buses, occluded buses and buses in bad lighting conditions while avoiding false positives in heavy traffic (middle right) and trucks(middle middle).

Figure 11: Bad detections by our system with threshold 0.98. It misses buses more often in difficult situations (top row and middle left - blue boxes). False positives occur with certain car configurations and occasionally with trucks (bottom row, middle middle and middle right - red boxes).

6 Bus Disruption Management

As a first use scenario of the ESVM-based bus recognizer, we have investigated the application of reacting to bus-induced disruptions in traffic flow. We define a flow disruption generally as a situation where traffic movement is fully blocked for some time interval during a green phase at the intersection. A common cause of a flow disruption that is frequently observed at the East Liberty pilot test site is that of a bus stop. Due to long dwell times of buses when picking up or dropping off passengers [15] , bus stops often significantly reduce the capacity of an intersection, and can have a major impact on vehicle delay. The congestion caused by a bus stop can also impose unexpected delays on subsequent buses.

To evaluate the potential benefit of an ability to recognize this situation, we propose an extension to the Surtrac adaptive signal control system that takes this information into account. In brief, Surtrac implements a decentralized approach to real-time traffic signal control. At its core is a novel intersection scheduling algorithm [34], which is used to determine the local allocation of green time to different phases at any given intersection on a cycle-by-cycle basis. The planned outflows that follow from a generated intersection schedule are then communicated to the intersection's neighbors to achieve coordinated, network-level behavior [32].

To produce a basic *bus disruption management* strategy, the intersection scheduling algorithm is modified to explicitly delay allocation of green time in the direction of the stopped bus. Specifically, an earliest start time $est_{n,i} \geq 0$ is imposed for each disrupted flow on incoming road segment n in phase i (essentially enforcing a delay of $est_{n,i}$) when the current inflows (vehicle platoons) are constructed for the local scheduler. Using this strategy, the SURTRAC scheduler provides a more accurate result when the disruption persists longer than the time $est_{n,i}$. The larger the value of $est_{n,i}$, the more likely it is that the new schedule will switch to serving the next phase early, though if there are many more vehicles in the current flow, the schedule will remain in the current phase. If the phase is switched, this strategy might impose some additional delay on the currently disrupted flow (if minimum green time constraints prevent switching back before the disruption is over). However, in this case the scheduler might service the current flow longer in the next cycle, as other flows will have been largely cleared. This strategy is expected to both reduce overall congestion and to reduce the delay for buses in the flow as well. In contrast, transit signal priority schemes, which often cause vehicle delays [1], might also suffer from congestion, since buses are sharing the flow.

To evaluate the impact of this disruption management strategy, we utilize a microscopic simulation model of the Surtrac pilot test site in the East Liberty area of Pittsburgh. The pilot test site, which is shown in Figure 12, is modeled using the SUMO (Simulation of Urban Mobility) [4] traffic simulator.

Several bus transit lines move through the pilot test site on Penn Avenue. As shown in Fig. 13a, there is a bus stop at the Penn Avenue and Highland Avenue intersection in East Liberty, and buses dwelling at the stop often cause disruptions on this road segment. For a incoming road segment n in phase i, a flow disruption is identified if the following condition is observed: the queue blocking state $QB_{n,i}$ is on, and no vehicle departs from the road segment for t_{NM} seconds during the green time. Figs. 13b and 13c give simulation results for average vehicle and bus travel times (in seconds), with and without use of the basic disruption management strategy. The vehicle travel times are averaged over all vehicles in the network. Here the bus frequency is assumed to be

[4] http://www.sumo-sim.org

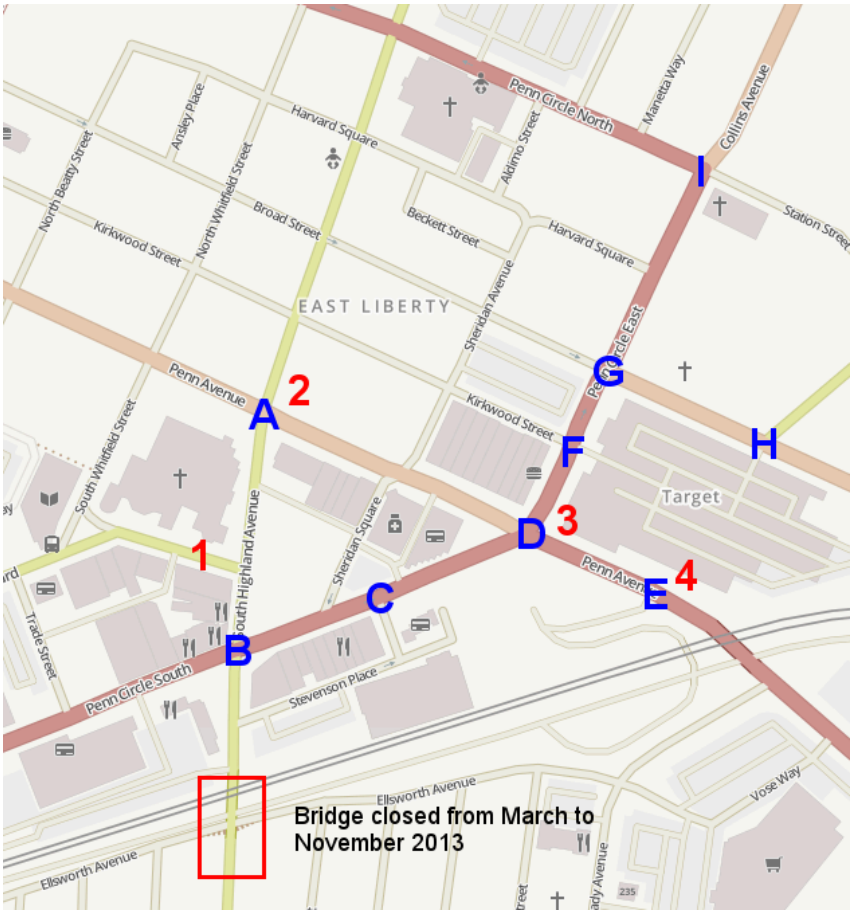
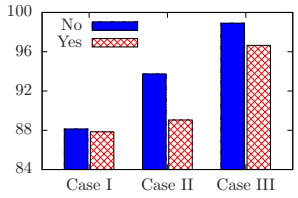
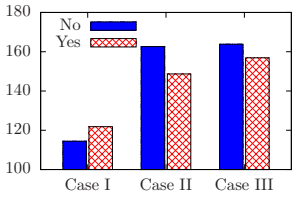


Figure 12: The East Liberty pilot test site with nine signalized intersections (**A**-**I**).

(a) Bus stop blocking traffic.



(b) Vehicle travel time (in secs).



(c) Bus travel time (in secs).

Figure 13: Disruption caused by stopped bus and results for vehicle and bus travel times, with and without the basic disruption management strategy.

20 buses per hour, which is quite close to actual bus traffic along the road segment **DA**. We assume that a stopping bus can be detected in 2 seconds (as per the requirements imposed in developing the bus recognizer) and consider different flow conditions and dwell times. Cases I and II have the same flow condition, with bus dwell times of 10 and 30 seconds, respectively. In case III, the flow is increased by 10%, and the bus dwell time is set at 30 seconds. For the vehicle travel time, cases II and III are significantly improved, whereas case I is only slightly improved. For the bus travel time, Case II and III are also improved, but case I is worse. Thus, this strategy is more likely to improve both vehicle and bus flows if the bus dwell time is longer.

7 Conclusion

This project has investigated the adaptation of recently developed techniques for object recognition in computer vision to the problem of detecting buses in real-time from commercial video camera image streams, and incorporation of this information into an adaptive traffic signal control strategy. Specifically, we experimented with Exemplar SVM (ESVM) Models, Deformable Parts Models and Maximum Margin Correlation Filters, three popular object detection algorithms, and developed a prototype ESVM bus recognizer capable of operating within the real-time requirements of the Surtrac adaptive signal control system. Performance analysis of this prototype system in laboratory experiments show that its use of domain constraints can achieve almost 100% accuracy on buses near the intersection, with acceptable degradation for buses further upstream. Complementary analysis with an extension to the Surtrac adaptive traffic signal control system showed that if one assumes an ability to detect buses stopped and blocking traffic in real-time, then it is possible to achieve a non-trivial improvement in vehicle travel times by servicing cross traffic without compromising bus throughput.

Our future plans are to turn these technology demonstrations into an operational system and realize these benefits in practice. The first step will be to test the ESVM recognizer's real-time detection capability. Our short term plans call for installation of the recognizer (running on its own separate processor) at the Penn Avenue/Highland Avenue intersection in East Liberty and a performance test of its ability in the field. We will run the system for an extended period, record bus detection events, and then retroactively compute the false negatives, the false positives and overall accuracy. We expect that this field test may uncover additional technical issues and lead to further refinement of the recognizer. Once performance is determined to be acceptable we will proceed to integrate the detection system with the Surtrac system that is currently controlling this intersection. Specifically, we will repeatedly extract image frames from the video stream and consider consecutive recognition events at the same designated bus stop location together with phase information to determine that a bus has stopped for passenger discharge and/or on-load. In such circumstances, the green delay heuristic will be used to modify the default behavior of the Surtrac intersection scheduling algorithm. Similar bus presence and phase information will also be used to determine if/when transit priority is appropriate, and modify the Surtrac intersection scheduler accordingly. At the hardware level, we will determine processor requirements to support both Surtrac and the bus recognizer, and develop an integrated communication infrastructure.

8 Acknowledgements

This research was funded in part by the University Transportation Center on Technology for Safe and Efficient Transportation (TSET) at Carnegie Mellon University, The Heinz Endowments, The Hillman Foundation and the Richard K. Mellon Foundation. Thanks to Gregory J. Barlow for his assistance in acquiring images from the video cameras at the Penn Avenue/Highland Avenue intersection, and in porting the prototype ESVM-based recognizer to the target intersection computer.

References

[1] Z. Abdy and B. Hellinga. Analytical method for estimating the impact of transit signal priority on vehicle delay. *Journal of Transportation Engineering*, 137:589–600, 2010.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Vishnu Naresh Boddeti. *Advances in Correlation Filters: Vector Features, Structured Prediction and Shape Alignment*. PhD thesis, Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, December 2012.

[4] Simon A Brock-Gunn, Geoff R Dowling, and Tim J Ellis. Tracking using colour information. *3rd ICCARV*, pages 686–690, 1994.

[5] A. Broggi, M. Bertozzi, A. Fascioli, and M. Sechi. Shape-based pedestrian detection. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 215–220, 2000.

[6] Norbert Buch, Mark Cracknell, James Orwell, and Sergio A Velastin. Vehicle localisation and classification in urban cctv streams. *Proc. 16th ITS WC*, pages 1–8, 2009.

[7] Yang Cai, Andrew Bunn, Peter Liang, and Bing Yang. Adaptive feature annotation for large video sensor networks. *Journal of Electronic Imaging*, 22(4):041110–041110, 2013.

[8] Hyunggi Cho, Paul E Rybski, Aharon Bar-Hillel, and Wende Zhang. Real-time pedestrian detection with deformable part models. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 1035–1042. IEEE, 2012.

[9] X. Clady, F. Collange, F. Jurie, and P. Martinet. Cars detection and tracking with a vision sensor. In *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, pages 593–598, 2003.

[10] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[11] Kevin Fehon, Jim Jarzab, Casey Emoto, and Deborah Dagang. Transit signal priority for silicon valley bus rapid transit. Technical report, Technical report, 2003.

[12] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[13] Mijail Gerschuni and Alvaro Pardo. Bus detection for intelligent transport systems using computer vision. In José Ruiz-Shulcloper and Gabriella Sanniti di Baja, editors, *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, volume 8259 of *Lecture Notes in Computer Science*, pages 59–66. Springer Berlin Heidelberg, 2013.

[14] Ross B Girshick, Pedro F Felzenszwalb, and David A Mcallester. Object detection with grammar models. *Advances in Neural Information Processing Systems*, pages 442–450, 2011.

[15] E.M. Gonzalez, M.G. Romana, and O.M. Alvaro. Bus dwell-time model of main urban route stops. *Transportation Research Record*, 2274:126–134, 2012.

[16] G. Grubb, A. Zelinsky, L. Nilsson, and M. Rilbe. 3d vision sensing for improved pedestrian safety. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 19–24, 2004.

[17] N.B. Hounsell, B.P. Shrestha, F. N. McLeod, S. Palmer, T. Bowen, and J. R. Head. Using global positioning system for bus priority in london: traffic signals close to bus stops. *Intelligent Transport Systems, IET*, 1(2):131–137, 2007.

[18] I. Kallenbach, R. Schweiger, G. Palm, and O. Lohlein. Multi-class object detection in vision systems using a hierarchy of cascaded classifiers. In *Intelligent Vehicles Symposium, 2006 IEEE*, pages 383–387, 2006.

[19] P. Lombardi and B. Zavidovique. A context-dependent vision system for pedestrian detection. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 578–583, 2004.

[20] Wenqi Ma, Hua Yang, and Yingkun Wang. A robust bus detection and recognition method based on 3d model and lsd method for public security road crossing application. In *Advances on Digital Television and Wireless Multimedia Communications*, pages 73–81. Springer, 2012.

[21] Aravindh Mahendran, Martial Hebert, and Stephen F. Smith. Exploiting domain constraints for exemplar-based bus detection for traffic scheduling. Unpublished working paper, submitted for publication, December 2013.

[22] Tomasz Malisiewicz. *Exemplar-based Representations for Object Detection, Association and Beyond*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August 2011.

[23] Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011.

[24] John Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[25] Walter Scheirer, Neeraj Kumar, Peter N. Belhumeur, and Terrance E. Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *The 25th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012.

[26] Stephen F Smith, Gregory J Barlow, Xiao-Feng Xie, and Zachary B Rubinstein. Smart urban signal networks: Initial application of the surtrac adaptive traffic signal control system. In *Proceedings 23rd International Conference on Automated Planning and Scheduling*, 2013.

[27] F. Suard, V. Guigue, A. Rakotomamonjy, and A. Benshrair. Pedestrian detection using stereo-vision and graph kernels. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 267–272, 2005.

[28] Xu Sun, Huapu Lu, and Juan Wu. Bus detection based on sparse representation for transit signal priority. *Neurocomputing*, 118(0):1 – 9, 2013.

[29] Advanced Vehicle Location Systems. Advanced vehicle location systems - http://www.vehiclelocationsystem.com/avl explained.htm.

[30] Traficon. Traficon - traffic video detection, http://www.traficon.com/index.jsp.

[31] Jing-wen Xiao, Zhi Yu, Pei-lin Nie, Xi-ying Li, and Dong-hua Luo. A bus video detection algorithm based on geometry and color. *Acta Scientiarum Naturalium Universitatis Sunyatseni*, page S2, 2005.

[32] Xiao-Feng Xie, Stephen F. Smith, and Gregory J. Barlow. Schedule-driven coordination for real-time traffic control. In *Proceedings 22nd International Conference on Automated Planning and Scheduling*, June 2012.

[33] Xiao-Feng Xie, Stephen F Smith, Gregory J Barlow, and T Chen. Coping with real-world challenges in real-time urban traffic control. In *Compendium of Papers of the 93rd Annual Meeting of the Transportation Research Board*, January 2014.

[34] Xiao-Feng Xie, Stephen F. Smith, Liang Lu, and Gregory J. Barlow. Schedule-driven intersection control. *Transportation Research, Part C: Emerging Technologies*, 24:168–189, October 2012.