2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)

Low Complexity Techniques for Robust Real-time Traffic Incident Detection

Kratika Garg, Alok Prakash, Thambipillai Srikanthan
School of Computer Science and Engineering
Nanyang Technological University, Singapore 639798
Email: kratika001@e.ntu.edu.sg, {alok, astsrikan}@ntu.edu.sg

***Abstract*—Traffic congestion is one of the leading reasons for the development of intelligent transportation systems(ITS). Traffic incidents are the second biggest cause of traffic congestions after infrastructural bottlenecks. Real-time traffic incident detection for timely clearing of roads is required to ensure smooth traffic flow. Apart from the real-time performance, scalable solutions which can monitor wide areas in a cost-effective manner are required. In this paper, robust, lean and real-time stationary foreground object detection technique to detect traffic incidents has been presented. We use block-based analysis in contrast to the conventional pixel-based analysis to lower the computational complexity of the proposed technique and achieve real-time performance. Experimental evaluations on widely used datasets demonstrate that the proposed method can achieve comparable accuracy to the existing state-of-the-art techniques. The real-time performance of the proposed system has also been demonstrated by implementing it on a low-cost embedded platform, Odroid XU-4, that still achieves a frame rate of 40 frames/second, thereby enabling real-time detection of traffic incidents.**

I. Introduction

The ever rising demand for mobility leads to increased traffic congestion that puts additional pressure on the already constrained road infrastructure, especially in urban areas. The major causes of traffic congestion can be classified into six categories, namely, bottlenecks, traffic incidents, work zones, bad weather, poor signal timing and special events. Among these, bottlenecks and traffic incidents are the leading causes that account for 40% and 25% of all the traffic congestions, respectively [1]. While the bottlenecks, which refer to the insufficient road capacity, can only be eased through additional infrastructure, traffic incidents, which refer to accidents, vehicle breakdowns, illegally parked vehicles, dropped objects, etc. can be effectively tackled by taking necessary steps to alleviate the congestions resulting from these incidents [2]. This can be achieved through timely dissemination of information to the relevant authorities as well as other road users and quick clearance of obstacles, collectively termed as stationary foreground objects (SFO), on the road after a traffic incident. This, in turn, necessitates mass deployment of low-cost intelligent sensors to enable automatic real-time detection of SFO.

Existing work in this area typically uses tracking based algorithms to detect stationary vehicles or abandoned objects [3] [4]. However, the computational complexity of these techniques increases significantly with increasing scene complexity [5] and hence are not suitable for implementation on low-cost embedded platforms needed for large scale deployment. Authors have also proposed 'persistence of foreground pixels' based methods to detect stationary foreground objects, but these are not robust in crowded situations [6] and typically generate many false alarms during heavy traffic. Few existing work [5] [7] have focused on low-complexity techniques for SFO detection, but they only target specific problems of detecting parked vehicles and hence are not suitable for the multitude of traffic incidents scenarios as highlighted above.

In our earlier work [8], we proposed a novel block-based technique to reduce the overall complexity in estimating traffic density on roads. A block variance based foreground detection strategy was proposed to estimate the percentage occupancy on the road in a compute-efficient manner. However, in that work we did not focus on the SFO detection, which requires additional processing to ensure that SFOs do not get assimilated into the background and are robustly detected without generating any false alarms during heavy traffic.

In this paper, we leverage the techniques in [8] to propose a block-based low-complexity, generalized stationary foreground object detection technique, which can cater to the detection of all traffic incidents namely accidents, stuck vehicles (e.g. vehicle breakdowns, illegally parked vehicles), dropped objects, etc. The proposed technique is intended to aid traffic authorities by rapidly, reliably and robustly relaying real-time information after any traffic incident. This can significantly reduce the response time for traffic incidents, which can lead to fast clearance of roads and smoother traffic flow.

The main contributions of this paper are:

1) A low-complexity block-based real-time stationary foreground object detection technique for various types of traffic incidents. The proposed technique can effectively deal with climatic changes, static and moving shadows, crowded situations as well as camera jitter.
2) The proposed techniques have been evaluated on a low-cost, low-power embedded platform, using the popular iLids Parked Vehicle Detection dataset to confirm its efficiency and potential for mass deployment.

II. Related Work

Cuevas et al. in [9] provided a detailed review of the various methods proposed in the literature for stationary foreground object detection. In the existing literature, detecting stationary foreground objects (SFO) has been mainly used to detect abandoned objects [9].

978-1-5386-1526-3/17/$31.00

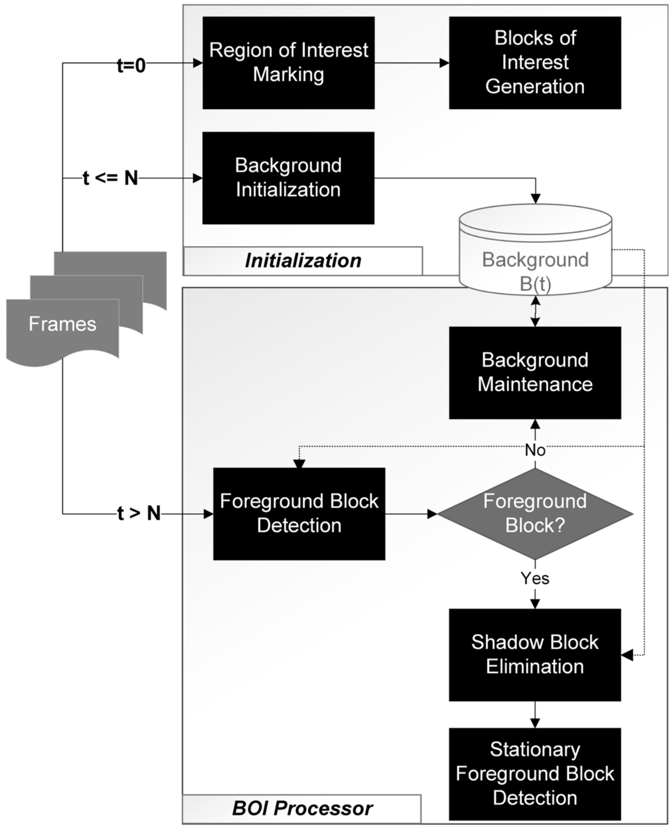


Fig. 1: Overall Flow

Traditionally, tracking based techniques have been most commonly used for detecting stationary foreground objects. Some of these methods were developed specifically to detect stationary vehicles [3], [10], while others used a more generalized approach of detecting stationary/abandoned objects [4], [11]. However, it has been highlighted in the existing literature that these techniques become computationally complex and ineffective when the scene gets crowded [5].

Apart from tracking, persistence is the next most widely used technique for detecting SFOs. Such techniques are more generic in nature and have been proposed for a generalized stationary/abandoned object detection. These techniques are based on the persistence of the pixels that are categorized as a part of the foreground (FG) for a predefined number of frames [12]–[14]. However, these techniques lead to false positive detections during crowded conditions e.g. heavy traffic [6].

In addition to tracking and persistence, the strategy proposed by Proikili et al. [15], called Dual FG Comparison(DFC), is also one of the commonly used methods for SFO detection. It uses two background models constructed at different learning rates, the background with fast learning rate which changes rapidly is used to detect the moving objects while the one with a slow learning rate detects the long-term changes i.e. SFOs. Many variations of this method have been proposed in the literature [16]–[18]. However, these techniques suffer from the fundamental issue of deciding the suitable learning rates for the long term and short term backgrounds which leads to low usability [9], [19].

Although several methods have been proposed in the literature for stationary foreground object detection, only a handful of them focus on the detection of traffic incidents. Some low-complexity techniques [5], [7], [20] that primarily focused on the real-time detection of parked vehicles on the roads were proposed after the i-Lids parking vehicle detection challenge dataset [21] was released in 2007. In order to lower their complexity, they restricted their region of interest to the 'no-parking zones' on the side of the roads. Pun et al. proposed a hybrid background model which was constructed offline to reduce complexity [5]. Lee et al. used a more generic approach of segmentation and tracking but in 1-dimension to reduce computations [7]. Boragono et al. proposed a DSP-based system for real-time detection of the parked vehicles [20]. However, these techniques achieve real-time performance by limiting the detection to a small area on the side of the roads, and hence are unsuitable for the detection of traffic incidents like accidents, vehicle breakdowns etc. that can potentially occur anywhere along or on the side of the road.

III. Proposed Method

Figure 1 shows the proposed two-phase approach for stationary foreground object detection. The first phase, called **Initialization**, is a one-time process in which the region of interest (ROI) is marked, block of interests (BOIs) are generated and a background is initialized. The second phase, called **BOI Processor**, is a recurring process in which each frame is divided into BOIs and sent to the **BOI Processor**. This phase performs Foreground Block Detection, Background Maintenance, Shadow Block Elimination, and Stationary Foreground Block Detection. In the following subsections, we explain the various steps involved in these two phases in detail.

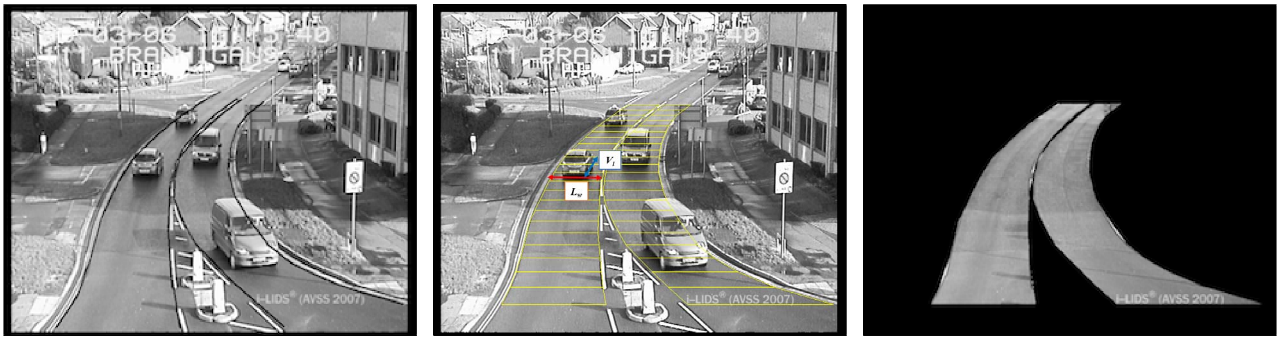
The techniques used for ROI Marking, BOI Generation, Background Initialization, Foreground Block Detection, and Shadow Block Elimination have been adapted from [8]. We have included the details of these techniques in this paper to present the proposed SFO detection technique in a holistic manner while highlighting the important changes made to these techniques to suitably adapt them for the detection of stationary foreground objects.

A. Region of Interest (ROI) Marking

The first step in the **Initialization** phase is to mark the region of interest (ROI) in the frame. Traffic incidents that take place within the lanes obstruct and degrade the traffic flow. Hence, we need to monitor and detect any SFO in the area within the lanes. Since **Initialization** is a one-time process, we manually mark the lane boundaries for each lane to get the ROI. The outcome of this step can be visualized in Figure 2a where a black line marks the boundaries of the lanes. This process can also be automated in future using a lane detection algorithm [22], [23].

B. Blocks of Interest Generation

Followed by ROI Marking, blocks of interest or BOIs are generated to be used for further processing. As stated in [8],



(a) (b) (c)

Fig. 2: (a) ROI (b) BOI (c) Background

the size of these blocks must be selected carefully based on the context of the problem. Blocks must be small enough to allow accurate localization of the smallest vehicles plying on the roads. At the same time, smaller block sizes reduce the efficiency of the block-based technique and increase the runtime of the algorithm.

In Figure 2b, the yellow blocks represent the blocks of interest in each lane. Since, this paper focuses on the detection of stationary foreground objects, which are predominantly vehicles, the size of BOIs have been selected to be smaller than the size of the smallest vehicle plying on the road. This enables us to localize even the smallest stationary vehicles. Hence, we have defined the length of a block as $V/3$ where V is the length of the smallest vehicle that crosses the road segment. To estimate V in image coordinates, we use the technique presented in [8], which uses the fact that the ratio of two distances in world coordinate system and the image coordinate system remains same. Hence,

$$LW/VL = L_w/V_l \tag{1}$$

where, LW and VL are lane width and smallest vehicle length in the world coordinate, while L_w and V_l are lane width and smallest vehicle length in the image coordinate system respectively. The LW is always fixed for a particular road segment and VL for the smallest vehicle (i.e. motorcycle) is around 1.8 m. The ratio LW/VL is defined as λ, hence eq. (1) can be rewritten as follows.

$$\lambda = L_w/V_l \tag{2}$$

For a lane width of 3.6 m, λ is calculated as 2. Using the above equation, each lane is divided into BOIs. Starting from the bottom the lane, the block length is calculated as follows:

$$\mathit{BOI\ Length} = V_l/3 = L_w/(\lambda * 3) \tag{3}$$

In Fig. 2b, it can be visualized that the length of each BOI is approximately equal to one-third of the length a small vehicle.

In our earlier work presented in [8], the BOIs are further divided vertically and only the central vertical division is used to estimate the traffic density. However, in this paper we refrain from creating the vertical divisions since all parts of the lane need to monitored to detect traffic incidents.

C. Background Initialization

The block based background initialization technique presented in [8] uses the stability of the signature of BOIs across frames to construct the background. The signature of a BOI has been defined as the *variance of the pixel intensities in a BOI*. When no vehicle passes through a BOI, the signature is expected to be the same across frames. This signature is illumination invariant as the intensities of all pixels change uniformly, especially since the size of the blocks is relatively small. In order to check the stability of this signature, the *variance of the variance values (VoV)* of a BOI from several frames is calculated. It is expected to be low when no vehicle passes through the BOI.

In this paper, we also use this property to initialize the background. However, a new background maintenance step, as explained in the next subsection, has been added after the initialization step to ensure that the SFOs do not get assimilated into the background.

Let the background and current frame for a BOI b_i, at time t, be defined as $B_t(b_i)$ and $I_t(b_i)$. For each BOI b_i, a circular buffer is constructed which stores the variance values of N most recent frames. Once the buffer is full, the variance of the variance values stored is calculated. We have used this VoV parameter, which can be mathematically defined as:

$$VoV(b_i) = \sigma^2(var(I_{t-N}(b_i)) : var(I_t(b_i))) \tag{4}$$

If $VoV(b_i) < T_B$, where T_B is a pre-defined threshold, the pixel intensities of BOI from the current input frame are duplicated to the background image, i.e.

$$B_{t+1}(b_i) = I_t(b_i) \tag{5}$$

As presented in [8], N is set to 4 and T_B is set to 100. This process is repeated until the background is constructed for all BOIs. An example of the constructed background for the Easy video in the iLids dataset [21] can be seen in Figure 2c.

D. Background Maintenance

Unlike the technique presented in [8], where the background maintenance is similar to background initialization, the proposed background update procedure is also dependent on the classification of a BOI as background or foreground, in addition to the check on the stability of the background signature used for the background initialization. This ensures that the stationary foreground objects do not get incorporated into the background. For the update procedure,

if $I_t(b_i)$ is background,

$$B_{t+1}(b_i) = \begin{cases} I_t(b_i) & VoV(b_i) < T_B \\ B_t(b_i) & VoV(b_i) \geq T_B \end{cases} \tag{6}$$

if $I_t(b_i)$ is foreground,

$$B_{t+1}(b_i) = B_t(b_i) \tag{7}$$

These regular updates to the background, ensure that the technique is adaptive to illumination changes and formation/fading of static shadows on the road.

E. Detection of Foreground Blocks

After background initialization, the detection of the blocks occupied by foreground objects is started. The foreground blocks detection technique proposed in [8] is based on the observation that there is a significant change in the block variance of the foreground blocks with respect to the background. This is characterized by the normalized variance difference with respect to the background for a BOI, which is defined as:

$$\Delta V(b_i) = \frac{abs(Var(B_t(b_i)) - Var(I_t(b_i)))}{max(Var(B_t(b_i)), Var(I_t(b_i)))} \tag{8}$$

However, this parameter fails to detect parts of the foreground objects that have similar variance as the background, e.g. the top of heavy vehicles. Although, even in these cases, there is an intensity difference between the background and the foreground pixels. Thus, in order to cope with such failures in [8], we used an additional parameter i.e. the ratio of foreground pixels in the BOI. The foreground pixels were generated from a thresholded difference image. This parameter can be defined as

$$FGR(b_i) = \frac{\text{Foreground Pixels in } I_t(b_i)}{\text{Total Pixels in } I_t(b_i)} \tag{9}$$

Here, we wish to highlight that foreground pixels were not solely used for detecting foreground blocks as they are more susceptible to slight illumination changes, background noise, etc., which requires additional pixel level computations to deal with the false positives.

Finally, the geometric mean of the two parameters i.e. $Occ(b_i)$ was used to classify the BOIs into background and foreground blocks, which can be defined as follows:

$$Occ(b_i) = \frac{2 * \Delta V(b_i) * FGR(b_i)}{\Delta V(b_i) + FGR(b_i)} \tag{10}$$

$$I_t(b_i) = \begin{cases} \text{foreground} & Occ(b_i) \geq T_O \\ \text{background} & Occ(b_i) < T_O \end{cases} \tag{11}$$

A threshold of $T_O = 0.3$ has been used as suggested in [8].

F. Elimination of Moving Shadow Blocks

The foreground block detection method detects any significant changes with respect to the background. However, this also leads to the detection of moving shadow blocks. In order to avoid detecting these moving shadow blocks as a foreground object, in this paper we have leveraged the shadow block elimination techniques presented in [8] with a minor adjustment. Next, we describe this technique and the adjustment used to eliminate the moving shadow blocks.

The technique is based on the observation that shadows do not lead to change in the texture of the background surface. We used the Normalized Cross Correlation (NCC) measure presented by Jacques et. al. on the foreground segmented pixels to estimate the texture similarity [24]. We took a logarithm of the NCC equation in order to reduce the computational cost of the pixel based calculations. The modified equation that was used is presented as follows:

$$\begin{aligned} &\log(NCC(i,j)) \\ &= \log(ER(i,j)) - \frac{1}{2}(\log(E_B(i,j)) + \log(E_I(i,j))) \end{aligned} \tag{12}$$

where,

$$ER(i,j) = \sum_n \sum_m B(i+n, j+m) I(i+n, j+m)$$
$$E_B(i,j) = \sum_n \sum_m B(i+n, j+m)^2$$
$$E_I(i,j) = \sum_n \sum_m I(i+n, j+m)^2$$
$$-N \leq n \leq N; -N \leq m \leq N$$

A pixel (i,j) is pre-classified as shadow pixel if $log(NCC(i,j)) > T_{ncc}$, where T_{ncc} is a pre-defined threshold. However, NCC is prone to wrong classification of dark foreground object pixels as shadows. In order to prevent such misclassifications, in [8] we used a refinement stage which was presented in [25] due to its ability to deal with shadows as well as reflections on the road. It uses the ratio of intensities between foreground and background pixels to differentiate between shadows and pixels from a dark object. For a pre-classified pixel (i,j), the intensity ratio $R(i,j)$ defined by [25] is presented as follows:

$$R(i,j) = (I(i,j) - B(i,j))/(I(i,j) + B(i,j)) \tag{13}$$

If $R(i,j)$ lies between $[-T_R, +T_R]$, the pixel is classified as a shadow. After the classification of all foreground pixels in a BOI as shadow/non-shadow pixels, blocks containing more than 95% shadow pixels are eliminated. This high threshold ensures that BOIs covered by both foreground objects and shadows are not classified as shadow blocks.

The thresholds had to be slightly adjusted from the ones presented in [8] for our proposed technique in order to achieve optimal performance. They were set to $log(0.95)$ and 0.4 for T_{ncc} and T_R respectively.

G. Stationary Foreground Blocks Detection

After detecting the foreground objects in the earlier steps, the proposed method uses a two-step process to identify the *stationary* foreground objects on the road. In the first step, the persistence of detection is used to detect candidate blocks for stationary foreground object detection whereas, in the second step the persistence in appearance is used to verify the candidate blocks and detect stationary foreground objects. The persistence of detection of each BOI is measured by a foreground counter defined as $fcount(b_i)$ and updated as follows:

if $I_t(b_i)$ is foreground,

$$fcount(b_i) = fcount(b_i) + 1 \tag{14}$$

if $I_t(b_i)$ is background,

$$fcount(b_i) = 0 \tag{15}$$

Once this value crosses a predefined threshold T_{FC}, the candidate blocks for detection of stationary foreground objects are identified. In the proposed algorithm, T_{FC} is set to the required alarm time, which can be set by the user.

Although, the persistence of detection has been a widely used parameter to detect stationary foreground objects, the persistence of detection as foreground blocks also takes place when there is heavy traffic on the roads [6]. Thus, in order to distinguish between heavy traffic and the stationary foreground objects the additional step is applied in the proposed method.

In this step, the persistence of the intensities in addition to the first step is used to differentiate between heavy traffic and stationary objects. In order to test this persistence, we evaluate the stability of the signature of the pre-identified candidate blocks. While the blocks occupied by stationary objects would experience a stability in their appearance, the blocks, seemingly occupied by heavy traffic, continuously change due to the movement of traffic. In section III-C, we presented a measure to calculate the stability in our approach for background initialization. Here, we use the same concept again to detect the stationary objects. Thus, in the second step, the candidate blocks which fulfill the following condition are finally classified as stationary foreground objects.

$$VOV(b_i) < T_B \tag{16}$$

Hence, the objects are detected as stationary foreground objects, if the following two conditions are met:

if $fcount(b_i) > T_{FC}$

$$I_t(b_i) = \begin{cases} \text{stationary foreground} & VOV(b_i) < T_B \\ \text{foreground} & VOV(b_i) > T_B \end{cases} \tag{17}$$

Once a stationary foreground is detected, an alarm would be turned on and the image would be transmitted to the manual operator to take necessary actions.

In addition to the detection of a stationary foreground object, it is also useful to know the exact time when it moved off (e.g. for vehicles) or was removed (e.g. for dropped objects). This is particularly important for the detection of illegal parking, since, typically there is a grace time for which a vehicle is allowed to wait before being classified as an illegally parked vehicle. In the proposed method we monitor the stationary foreground blocks to detect the time when the object is moved/removed. As soon as there is no change detected with respect to the background in this stationary foreground blocks, they are marked as background blocks and $fcount(b_i)$ is updated to zero. This signifies that the stationary object has moved or has been removed from the scene.

Algorithm 1: Stationary Foreground Object Detection

1 **for** *each* b_i **do**
2   **if** $I_t(b_i)$ *is foreground* **then**
3         $fcount(b_i) = fcount(b_i) + 1$;
4   **if** $fcount(b_i) > T_{FC}$ **then**
5             $VoV(b_i) = \sigma^2(var(I_{t-N}(b_i)) : var(I_t(b_i)))$;
6   **if** $VoV(b_i) < T_B$ **then**
7                 $SFO(b_i) = 1$;
8   Alarm: SFO Detected in b_i;
9   **end**
10   **end**
11   **else**
12         $fcount(b_i) = 0$;
13   **if** $SFO(b_i) == 1$ **then**
14             $SFO(b_i) = 0$;
15   Alarm: SFO Moved from b_i;
16   **end**
17   **end**
18 **end**

TABLE I: Dataset

	iLids [21]	**BMC** [26]
Sample Frame		
Image Size	720*576	640*480
Climatic Conditions	Cloudy, Sunny	Sunny, Cloudy, Foggy, Windy
Camera Jitter	Yes	No
Background Shadows	Yes	Yes
Moving Shadows	Long	Small

Algorithm 1 summarizes our stationary foreground object detection technique. For each BOI b_i, if $I_t(b_i)$ is classified as foreground, then foreground counter for that BOI $fcount(b_i)$ is incremented (Line 3). Next, a foreground persistence check is done to identify candidate blocks for stationary foreground object detection (Line 4). In order to identify the SFOs from these candidate blocks, the persistence of appearance check is applied, the SFOs are detected and SFO bit for the BOI is set (Line 5-9). If $I_t(b_i)$ is classified as background, the foreground counter is reset (Line 12) and if a SFO was previously detected in that BOI, it is reported that it has moved (Line 13-16).

IV. Results

In this section, we present details about the datasets used, quantitative and qualitative evaluations of the proposed method, comparison with state-of-the-art methods as well as the run-time performance. We implemented the proposed algorithms in C++ on a PC with Intel Xeon processor running at 3.50 GHz with 16 GB RAM using Windows 7 operating system. Additionally, to test the portability of the proposed techniques on low-cost embedded platforms, it has also been implemented and evaluated on the Odroid-XU4 platform [27]. We will discuss the details of this platform in Section IV-C.

The stationary foreground object detection technique has been tested on the *AVSS iLids Parked Vehicle dataset* [21]. It consists of three daylight video clips marked as Easy, Medium and Hard according to the difficulty level. In Easy, a van is parked close to the camera, in Medium a car is parked close to the camera, but it has sudden camera shakes which make it challenging. Lastly, in Hard, a car is parked very far away from the camera which is also partially occluded by a road sign making it extremely challenging. It also contains a night-time video clip. In addition to the iLids dataset, *Background Models Challenge (BMC) Dataset* [26] has been used to evaluate foreground block detection in different climatic conditions. More details about both the datasets used for evaluating the proposed technique are presented in Table I.

A. Qualitative Results

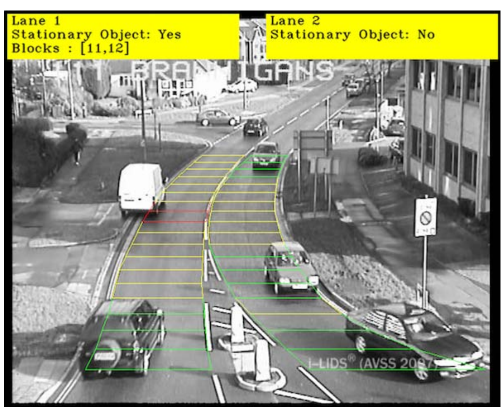
In this section, we present the qualitative results for the proposed method. First, the output images for the detection of the stationary foreground objects have been presented. This is followed by some additional results that prove the robustness of the proposed techniques in different climatic conditions.

In Figure 3, qualitative results of stationary object detection can be visualized. The labels on the images show the detection of the stationary object along with the lane number and block number. The stationary vehicles are effectively detected for all four videos namely iLids - Easy, Medium, Hard and Night, as can be seen in Fig. 3a, 3b, 3c and 3d respectively. The blocks marked in red are the ones which are detected as encapsulating the stationary vehicle. There are some false positive foreground detections in the Night video due to headlight reflections on the road. However, these do not lead to any false alarms for stationary foreground objects as the reflections coming from moving cars change continuously as well as rapidly and hence do not exhibit persistence in appearance, which is a necessary condition for the proposed stationary foreground object detection technique.

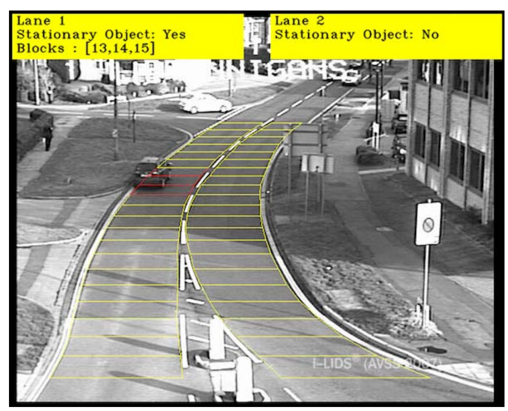
Next, we experimented with the BMC dataset to display the robustness in the detection of foreground blocks. Figures 4a-4d show the results for cloudy, sunny, foggy and windy conditions respectively. From these results, it can be seen that the proposed techniques are invariant to climatic changes. Specifically, from Figure 4c, it is evident that the proposed techniques can effectively differentiate between shadows and dark vehicles, hence detecting the dark foreground objects accurately. In yet another situation, as observed in Figure 4d, the proposed method robustly detects the grey vehicle even though its appearance is very similar to the road background.

B. Quantitative Results

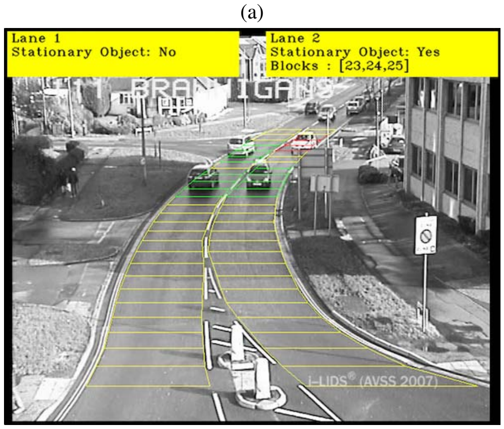
In this section, we provide the quantitative results and comparison with state-of-the-art SFO detection techniques.



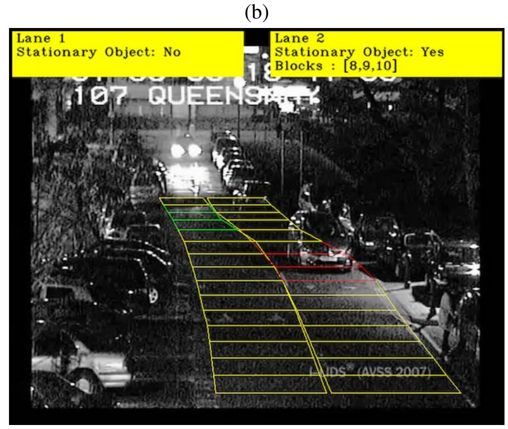
(a)



(b)

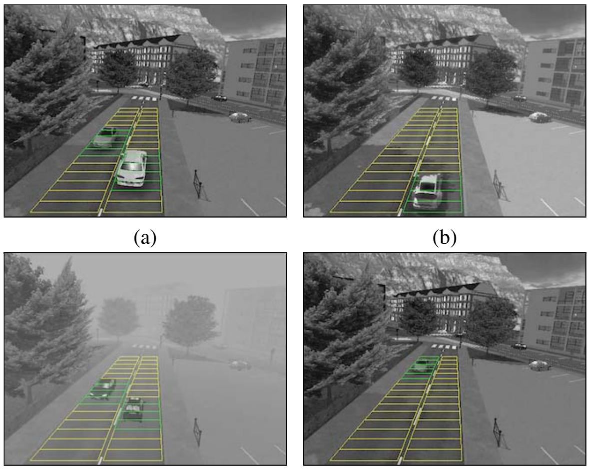


(c)



(d)

Fig. 3: Detection of Stationary Foreground Object for (a) iLids-Easy (b) iLids-Medium (c) iLids-Hard (d)iLids-Night



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

Fig. 4: Detection of Foreground Blocks for BMC
(a) Cloudy (b) Sunny (c) Foggy and (d)

1) Accuracy for Detection of Foreground Blocks: In order to assess the robustness of the detection of foreground blocks, we evaluate our techniques on the BMC Dataset. The ground truth provided by the authors for the BMC Dataset contains the classification of each pixel as foreground/background. To evaluate our block based approach using this ground truth, we map our BOIs onto their ground truth (18 per lane) and define any block that has at least 10% foreground pixels, out of the total pixels, as a foreground block. The blocks which have 0%-10% foreground pixels are ignored. Using this ground truth, the True Positive Rate (TPR) and False Positive Rate (FPR) is calculated.

An additional preprocessing step of histogram equalization is applied for this dataset. Table II presents the TPR and FPR values for the videos in the BMC Dataset which have been taken in different climatic conditions. We are able to achieve a high value of recall i.e. TPR for all climatic conditions. Even in challenging foggy conditions, we are able to achieve a TPR of more than 90%. Furthermore, the value of False Alarm Rate i.e. FPR is extremely low.

2) Stationary Foreground Object Detection Accuracy: As mentioned earlier, the AVSS iLids Parked Vehicle dataset has four videos namely Easy, Medium, Hard and Night, each containing an illegally parked vehicle. To compare the accuracy of illegal parked vehicle detection, the time duration of its detection has been used. It is characterized by the start time, i.e. 1 min after a vehicle has been illegally parked and an end time, i.e. when the illegally parked vehicle starts to move again. The absolute error with respect to the ground truth is used as a metric for comparison with state-of-the-art techniques.

TABLE II: Foreground Block Detection Accuracy

Video	TPR	FPR
Cloudy	99.60%	0.47%
Cloudy with noise	99.80%	0.47%
Sunny with noise	98.91%	0.48%
Foggy with noise	92.34%	0%
Windy with noise	99.83%	0.49%

For our approach, since one stationary object is detected in multiple consecutive blocks, the average value of the start and end times for the detected stationary foreground blocks is taken to compare with state-of-the-art techniques.

The detailed results are presented in Table III. The proposed method achieves the best average performance among the techniques which have been evaluated on all the four videos, including the extremely challenging Night video. Using our proposed approach, there is a slight error recorded in the end time for all videos. This error is attributed to the fact that the corresponding blocks take some time to get updated as the road background, which is the criteria established in the proposed method to detect when the stationary foreground object has moved as described in Section III-G. This error is slightly higher in the iLids-Medium video in which another vehicle follows the stationary vehicle on its tail end, when it starts moving, hence leading to a delay in the recorded end time. We will take steps to rectify this problem in our future work.

C. Run-Time Performance

We first measured the run-time performance of the proposed algorithm on Intel Xeon CPU, 3.50 GHz with 16 GB RAM running Windows 7. The frame rates achieved by the proposed method for the iLids Dataset are presented in Table IV. On average, it achieves a frame rate of around 260 frames/second for the iLids daylight videos, which has a resolution of 720*576 pixels. A higher frame rate is achieved for the night video as the moving shadow elimination block is not used.

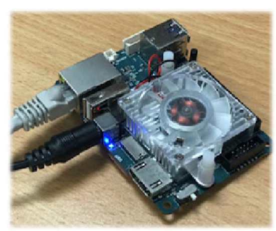


Fig. 5: Odroid XU4

In order to evaluate the efficiency of the proposed techniques on an embedded platform, we also executed and verified them on a state-of-the-art mobile application development platform — Odroid-XU4 [27] from Hardkernel shown in 5. This platform contains a Samsung Exynos 5422 mobile SoC that implements ARM big.LITTLE technology with a cluster of four ARM Cortex A15 cores (big cores) and a cluster of four ARM Cortex A7 cores (small cores). In our current setup, this platform runs on the popular Ubuntu 15.10 LTS operating system. We run our algorithm on a single A15 core, running at 2GHz. In contrast to the 130W thermal design power (TDP) of the Intel Xeon CPU, the Exynos 5422 SoC is constrained to a maximum of ∼10W TDP and is hence representative of a typical SoC used in low-cost low-power embedded platforms.

The third column in Table IV shows the frame-rates achieved on this platform. On average, we achieved a frame-

TABLE III: Performance on iLids Dataset

	Easy			Medium			Hard			Night			All
	Start	End	Abs. Err.	Start	End	Abs. Err.	Start	End	Abs. Err.	Start	End	Abs. Err.	Avg. Err.
Ground Truth	02:48	03:15	-	01:28	01:47	-	02:12	02:33	-	03:25	03:40	-	-
Boragano et al. [20]	02:48	03:19	4s	01:28	01:55	8s	02:12	02:36	3s	03:27	03:46	8s	5.75s
Venetianer et al. [4]	02:52	03:16	5s	01:43	01:47	15s	02:19	02:34	8s	-	-	-	9.33s
Guler et al. [11]	02:46	03:18	5s	01:28	01:54	7s	02:13	02:36	4s	03:28	03:48	11s	6.75s
Lee et al. [7]	02:51	03:18	6s	01:33	01:52	10s	02:16	02:34	5s	03:25	03:36	**4s**	6.25s
Pun et al. [5]	02:48	03:19	4s	01:31	01:50	**6s**	02:12	02:35	**2s**	-	-	-	**4s**
Proposed Approach	02:48	03:18	**3s**	01:28	01:55	8s	02:12	02:36	3s	03:23	03:42	**4s**	**4.5s**

rate of 40 frames/second for the iLids daylight videos. This clearly shows the efficiency of the proposed techniques and ensures that they can be implemented on low cost embedded platforms for mass deployment.

The real-time performance of our techniques can be attributed to the nature of calculations in the proposed approach. The proposed method is primarily based on a block level analysis, which leads to a major reduction in computational complexity, especially in comparison to techniques that rely on pixel-level analysis. The computationally complex shadow elimination technique based on NCC [24], has only been used on a limited number of pixels. It has also been modified to use ‘logarithmic’ calculations instead of multiplications and square root, enabling the use of look-up tables and hence suitable for implementation on low-cost embedded platforms. Owing to the low complexity of our method and portability on low-cost hardware platforms, it is suitable for mass deployment to enable real-time traffic incident detection.

TABLE IV: Achieved Frame-Rate

Video (*No. of BOIs*)	Intel Xeon	Odroid XU4
iLids-Easy (*36*)	270.05	40.47
iLids-Medium (*42*)	257.87	39.36
iLids-Hard (*54*)	270.32	41.28
iLids-Night (*30*)	453.50	69.18

V. Conclusion

In this paper, we introduced a novel technique for detecting stationary foreground objects on the road for traffic incident detection, which is a stepping stone towards pro-active traffic congestion management. The proposed method incorporates a block based background initialization, maintenance, foreground block detection with shadow elimination and a stationary foreground block detection technique which has significantly low computational complexity when compared to existing approaches that rely on pixel-based analysis. This technique can be used to relay real-time information of traffic incidents to the concerned authorities which would enable timely dissemination of information and fast clearance of roads, hence, reducing the probability of traffic congestion. Experiments on different traffic videos demonstrated that the proposed method is robust to illumination changes, climatic conditions and shadows; gives real-time performance and has comparable accuracy with existing state-of-the-art techniques.

References

[1] “Operations Story - Federal Highway Administration U.S.” http://www.ops.fhwa.dot.gov/aboutus/opstory.htm.

[2] R. P. Loce *et al.*, “Computer vision in roadway transportation systems: a survey,” *Journal of Electronic Imaging*, 2013.

[3] Q. Fan *et al.*, “Long-term object tracking for parked vehicle detection,” 2014.

[4] P. Venetianer *et al.*, “Stationary target detection using the objectvideo surveillance system,” 2007.

[5] C.-M. Pun, “A real-time detector for parked vehicles based on hybrid background modeling,” *Journal of Visual Communication and Image Representation*, 2016.

[6] D. Ortego *et al.*, “Multi-feature stationary foreground detection for crowded video-surveillance,” 2014.

[7] J. T. Lee *et al.*, “Real-time illegal parking detection in outdoor environments using 1-D transformation,” *IEEE TCSVT*, 2009.

[8] K. Garg *et al.*, “Real-time road traffic density estimation using block variance,” in *WACV*, 2016.

[9] C. Cuevas *et al.*, “Detection of stationary foreground objects: A survey,” *CVIU*, 2016.

[10] A. Bevilacqua *et al.*, “Real time detection of stopped vehicles in traffic scenes,” 2007.

[11] S. Guler *et al.*, “Stationary Objects in Multiple Object Tracking,” 2007.

[12] W. Hassan *et al.*, “Illumination invariant stationary object detection,” *IET Computer Vision*, 2013.

[13] L. Maddalena *et al.*, “Stopped object detection by learning foreground model in videos,” *IEEE TNNLS*, 2013.

[14] J. Pan *et al.*, “Robust abandoned object detection using region-level analysis,” in *ICIP*, 2011.

[15] F. Porikli, “Detection of temporarily static regions by processing video at different frame rates,” 2007.

[16] Y. Zeng *et al.*, “A Novel Abandoned Object Detection System Based on Three-Dimensional Image Information,” *Sensors*, 2015.

[17] R. Wang *et al.*, “Static and moving object detection using flux tensor with split gaussian models,” in *CVPRW*, 2014.

[18] W. Wang *et al.*, “A new approach for real-time detection of abandoned and stolen objects,” 2010.

[19] A. Albiol *et al.*, “Detection of parked vehicles using spatiotemporal maps,” *IEEE ITS*, 2011.

[20] S. Boragno *et al.*, “A DSP-based system for the detection of vehicles parked in prohibited areas,” 2007.

[21] “i-LIDS Parked Vehicle Dataset,” http://www.eecs.qmul.ac.uk/~andrea/avss2007_ss_challenge.html.

[22] Y. Li *et al.*, “On Automatic and Dynamic Camera Calibration based on Traffic Visual Surveillance,” 2007.

[23] M. Dubska *et al.*, “Automatic Camera Calibration for Traffic Understanding,” in *BMVC*, 2014.

[24] J. Cezar *et al.*, “Background Subtraction and Shadow Detection in Grayscale Video Sequences,” in *Computer Graphics and Image Processing*, 2005.

[25] A. Gawde *et al.*, “Lightweight and robust shadow removal for foreground detection,” in *AVSS*, 2012.

[26] A. Vacavant *et al.*, “A benchmark dataset for outdoor foreground/background extraction,” *LNCS*, 2013.

[27] “Odroid-XU4,” http://goo.gl/Nn6z3O.