Comparative Analysis and Detection of Street Parked Vehicles using Spatiotemporal Maps and Corner Detection Methods

Manali Jayebhaye
PG student
MGMs COE, Nanded

M.R.Banwaskar
Dept. of E&TC
MGMs COE, Nanded

ABSTRACT
Traffic control and efficient use of existing infrastructure are key challenges in recent time. This work carries out comparative analysis of various corner detection methods in different conditions, to detect presence of parked vehicles in street lanes. Different algorithmic implementations are used, to detect corners over targets in video stream and then classify them over a spatio-temporal analysis maps to identify parked vehicles. The system has been evaluated using i-Lids public dataset and has proven to be robust against common difficulties found in Closed Circuit Television (CCTV) such as high noise levels, varying illumination and the presence of momentary occlusion by other vehicles. This could be used to detect illegal and double-parked vehicles in metropolitan areas and to detect incidents on roads with lanes.

This paper implemented three corner detection algorithms, Harris, SIFT and FAST and analyzed the comparative performances of all three implementations over different parameters viz. precision, recall, false alarm and then conclude on best performing algorithm. The comparative results are presented.

General Terms
Object detection, Harris corner detector, SIFT corner detector, FAST corner detector.

Keywords
Parked vehicle detection, corner detection, object classification, spatiotemporal analysis, and video analysis.

1. INTRODUCTION
During past few years, urban traffic is a real problem for most medium sized and large cities. To reduce the problems caused by traffic congestion, Intelligent Transportation Systems (ITS) are being deployed to achieve a more efficient use of existing infrastructures [1]. The problem of tracking objects in different framework has been studied generally due to its applicability to infinite practical situations. One powerful application is the problem of detecting vehicles that have been parked illegal and double parked in different locations [2]. Traffic counts, vehicle classifications and speed are fundamental parameters for a variation of transportation projects, ranging from transportation organization to modern intelligent transportation systems. In computer vision, object detection and tracking is an active research area which has attracted huge attentions from multi-disciplinary fields, and it has applications in many fields like surveillance system, public security system, service robots, and virtual reality interfaces [3]. Dynamic traffic planning systems [4] achieve better regulation of the whole infrastructure by using complex models with many input traffic variables. Some examples of input variables for such systems are previous and instantaneous values of traffic data in different roads i.e. number of vehicles, mean stopping time, queue lengths, average speed etc. Detection of parked vehicles involves detecting objects that remain stopped for more than a certain time [5]. This vehicle detection system operates on 720×526 pixels images, vehicles are detected at 25 frames /second. System attain high frame rates working only with information present in a single 2-dimensional grayscale image. Results were compared from these three algorithms, and proved the FAST detector is excellent for detection rate, it is more efficient, robust, high repeatability rate, localization is good for all junction types and it take less time for processing [7]. The experiments are carried out using MATLAB and tested on video frames.

2. PREVIOUS WORK
In the large number of papers on video processing and image processing fields, the following topics were found relevant to the challenge of parked vehicle detection. So many researches of parked vehicles detection and vehicle identification have been approached by corner detection algorithms, tracking, and segmentation. Some of the researched work is as follows.

In Borango *et al.* [8] described the system for automatic robust video Surveillance and its application to the problem of locating vehicles that stop in prohibited area. In this paper, Ipsotek Visual Intelligence Platform is used for video processing, alarm generation and interfacing with the operator. The performance evaluation process is carried out with the I-Llids parked Vehicles refer dataset. Lee *et al.* [9] proposed methodology for detecting parked vehicles by applying a novel image projection. It reduces the dimensionality of the data, computational complexity of the segmentation and tracking processes. In this paper authors proposed methods are background modeling, 1-D projection, segmentation, tracking, reconstruction. Bevilacqua *et al.* [3] represents a method to detect stopped vehicles based on the detection of the tracked objects centroid position during short time intervals and uses background subtraction to detect foreground objects. Maddalena *et al.* [2] proposed methods for background subtraction, background and foreground modeling and neural network. The main aim of this paper is to obtain the objects that keep the user attention in accordance with a set of predefined features, by learning the trajectories and features of moving and stopped objects in a self-organizing manner. Viola *et al.* [5] described an approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates. This work is differentiated by three key contributions i.e. integral image, AdaBoost algorithm and a method for combining

increasingly more complex classifiers in a cascade structure. Cucchiara *et al.* [10] described two sets of image analysis algorithms to provide vehicle detection in traffic areas. Two different approaches have been exploited depending on illumination condition of day and night, in night images, spatio-temporal analysis on moving templates and in day time images. Mei H. *et al.* [11] described a method for tracking multiple objects whose number is unknown and varies during tracking. By using Gaussian- mixture based background modeling method to generate a binary foreground mask image and by using neural network based object detection module detects pedestrians.

3. ALGORITHM OVERVIEW

Detection rate of parked vehicles are evaluated and compared using Harris, SIFT and FAST algorithms. Firstly, corner points are detected using Harris. It was observed that Harris gives good results for detection rate, but its localization is not good for all junctions and it took more time for computation therefore Scale Invariant Feature Transform (SIFT) algorithm is used. This algorithm gives good results for detection rate as compared to Harris. The SIFT features are highly distinctive and features are correctly matched with high probability against a large dataset of features from video. SIFT gives better performance and the currently focuses on FAST corner detector through machine learning approach that has better performance. SIFT algorithm consumes too much time in the computation so the results in lagging on the frames (video) that reduces the frame quality significantly and more false corners are detected in the video frames. To overcome this issue we implemented the FAST corner detection algorithm. This method is compared with the results of SIFT and Harris detector. Experimental results show that the FAST detector gives excellent results when compared to Harris and SIFT. Using this detector more number of true corners is detected in the video frames. The main advantage of this detector is efficiency and it took less time for processing. Comparison of Harris, SIFT and FAST corner detection is shown in fig.1 which shows that FAST corner detector gives the better result than other two algorithms.

4. CORNER DETECTION ALGORITHMS

4.1 Harris Corner Detection Algorithm

This algorithm is realized by extracting feature points from images, which was developed by Harris and Stephens (1988) [12]. They needed a system to match corresponding points in continuous image frames, but were interested in tracking both corners and edges between frames. This detector has high computational demand. This is a far more desirable to detection and repeatability rate. The speed of this detector is low. The principal of Harris algorithm is: basically, a template of $m \times n$ size as detection window is designed as required, which scans the entire pixel along the 2-D grayscale image.

Fig 1: Comparison result of Harris (upper left), SIFT (upper right), and FAST (lower)

Determine changes of grey values within the template conforming to equation 1. where I_x and I_y present gradient map of the image in x and y direction, and $G(s)$ is the Gaussian template. The matrix of Harris algorithm is,

$$M = G(S) * \begin{bmatrix} Ix^2 & I_x I_y \\ I_x I_y & Iy^2 \end{bmatrix} \quad (1)$$

Where, M- autocorrelation matrix of pixels and the 2-eigenvalues $\lambda 1$ and $\lambda 2$ are proportional to the curvature of M when two eigenvalues are relatively small it means $\lambda 1 = 0$ and $\lambda 2 = 0$ indicating that the area near object point is 'flat region'. One large eigenvalue indicates 'edge region' and when both large eigenvalues with significant changes in any directions indicates 'corner region'. This indicates that the object point is located in corner region. Threshold value formula of corner can be driven according to this nature, in equation 2, when the changes at certain point is greater than given threshold value, then this point can be examined as the corner to be detected.

$$R = det(M) - k\, Trace(M)^2 \quad (2)$$

$detM$ and $Trace$ represents determinant of the matrix is empirical coefficient (k=0.04~0.06). The quality and quantity of corner extraction is dependent on the size of R-value: too large R will result in corner, on the other hand a too small R will extract some points above the corners. According to the attributes of the above mentioned eigenvalues at corner is,

$$R = \frac{det(M)}{Trace\ (M)^2} \quad (3)$$

Where, $det\, M$ is the product of eigenvalues $\lambda 1 \times \lambda 2$ and $trace(M)$ is the sum of eigenvalues $\lambda 1 + \lambda 2$. When both eigenvalues are greater, R tends to 1 and greater, other hand, R smaller eigenvalues will make R tend to zero. When both eigenvalues are small and less than 1 then R tends to 0.

The steps of Harris algorithm described in short as a below:

Input- 2-D Gray scale image, threshold value T, K value, Gaussian variance.

Output- Map indicating position of every corner.

1. Calculate the x and y derivative of an image.
2. Calculate product of derivative at every pixel.

3. Calculate the sum of product of derivative at every pixel.
4. Define the matrix at every pixel(x, y).
5. Calculate the response of the detector at every pixel R.
6. Threshold on value of R. Calculate nonmax suppression.

4.2 SIFT Corner Detection Algorithm

Scale Invariant Feature Transform (SIFT) corner detection is extracting distinctive invariant features from 2-dimensional images, which was developed by David Lowe (2004) [7]. The SIFT features are highly characteristics and correctly matched with high feasibility against a large dataset of features from video frames. This approach can strongly identify objects and achieve near real time application. SIFT has four stages and they are used to generate the set of image feature points such as scale-space extrema detection, key point localization orientation assignment and key point descriptor. SIFT transforms image into scale invariant co-ordinates relative to local features. The scale space of the image function,

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (4)$$

Where,

$L(x, y, \sigma)$ −Produced from the convolution of a variable and

$G(x, y, \sigma)$ − The scale Gaussian with input image and $I(x, y)$ is the convolution operator in x and$y$.

Author proposed using scale space extrema in the Difference of Gaussian function (DoG),

$$D(x, y, \sigma) = [(G(x, y, k\sigma) - G(x, y, \sigma)] * I(x, y)$$
$$= L(x, y, k\sigma) - L(x, y, \sigma) \quad (5)$$

The DoG function will give a strong response along edges, even if the location on the edge is poorly determined and therefore uncertain to small amounts of unwanted signal. The principal curvature can be computed from 2×2 Hessian matrix at the location and scale of key point,

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \quad (6)$$

Avoid these for which, the SIFT key points distinctiveness, it means individual features can be matched to a large data base of target is achieved if a high dimensional vector representing the image gradients with a local region of the image is obtained. The key points are shown to be invariant to image rotation and scale strong across a considerable range of affine distortion, change in brightness, and addition of noise. The SIFT key points perform well, for image blur also.

Fig. 8 shows the results of Easy and Hard sequences using SIFT detector. Fig.9 shows the results of corner classification of SIFT. Fig.10 shows the result of color maps and Fig.11 shows the results of spatiotemporal maps of Easy and Hard sequences.

4.3 FAST Corner Detection Algorithm

FAST means Features from Accelerated Segment Test. This algorithm was proposed originally by Rosten Edward and Tom in 2004 for identifying interest point in an image. In an image interest point is a pixel which has a well-defined position and can be completely detected. These interest points have high local information content and they should be ideally repeatable between distinct images.

FAST corner detection algorithm is used to extract interest points in many computer vision applications. As per its name it is fast and indeed it is faster than many corner detectors like Harris, SIFT etc. This algorithm is suitable for real time video processing applications because of its high speed performance and computational efficiency.

The algorithm steps are as follows,

1. First select a pixel, 'p' in the 2-D Gray scale image. Let the intensity of this pixel to be Ip. This pixel is to be identified as interest point or not.
2. Consider threshold intensity value T that will be 20%.
3. Next, consider a circle of 16 pixels surrounding the pixel p, this is called as Bresenham circle. Take the radius of this circle as 8.
4. Then 'N' adjacent pixels out of these 16 pixels require to be either above or below I_p by the value T if the pixel requires to be detected as a corner. $(N = 12)$
5. To generate the algorithm fast first differentiate the intensity of pixels 1, 5, 9 and 13 of the circle with I_p . Out of these four pixels three pixels should satisfy threshold criteria so that the corners will exists.
6. If at least three of the four pixels I_1, I_5, I_9 and I_{13} are not above or below $I_p + T$ then P is not a corner. In this case discard the pixel 'p' as a possible corner. Else if at least 3 of the pixels are above or below $I_p + T$, then checked for all 16 pixels and if 12 contiguous pixels lowering in the image. Then this procedure will apply for every pixel at a time one pixel. The pixel at position relative to denoted as $p \rightarrow x$ have three states,

$$S_{p\rightarrow x} = d, \quad I_{p\rightarrow x} \leq I_p - t \quad \text{(Darker)}$$
$$S, \quad I_p - t < I_{p\rightarrow x} < I_p + t \quad \text{(Similar)}$$
$$b, \quad I_p + t \leq I_{p\rightarrow x} \quad \text{(Brighter)} \quad (7)$$

5. SYSTEM MODEL

5.1 Processing of Every Video Frame

Fig.2 shows a flowchart of the processing performed for every frame from the video. The other blocks will explain in rest of the section.

First, given a frame at time 't', FAST corner detector detects the image corners. The execution in terms of detection of parked car is the same as FAST. Fig.3 shows the results of corner detection using three methods on video frames. The main point is that both of them to create large clusters of static corners at the location of parked cars. The result of corner detection is a list of corner points for every frame. Next step is masking, masking is one type of filter, and it selects a subset of relevant corners from results of corner points. Masking may also contain holes that are used to avoid static corners at selected co-ordinates to be identified as background corners and are automatically detected. In next step, the selected corners are divided into two parts i.e. static and dynamic; get two disjoint lists of corners S_t and M_t.

Any motion estimation method like block matching or optical flow could have been used at this end. After all, much simple technique has been used because the direction of the motion is not needed; it is only needed to know whether the corner is in

motion. Therefore, for each corner P_t^ with x_i and y_i co-ordinates of an image,

$$di = \max(x', y') \in N(ci)\{|I_t(x', y') - I_t(x', y')|\} \quad (8)$$

is calculated, where $I_t(x, y)$ is 2-dimensional gray scale image and $N(C_i)$ is the small neighborhood size around corner C_i. Here, the size of neighborhood is 11×11 in all experimentation. If d_i is over threshold dth, then corner is classified as a dynamic corner, otherwise it is classified as a static corners. The typical value of threshold is 10-25 for 8-bit quantized grayscale images. There is great computational saving in this simple approach compared with any possible motion estimation techniques. This simple technique can be used in case of high noise, sudden illumination changes or camera vibration.

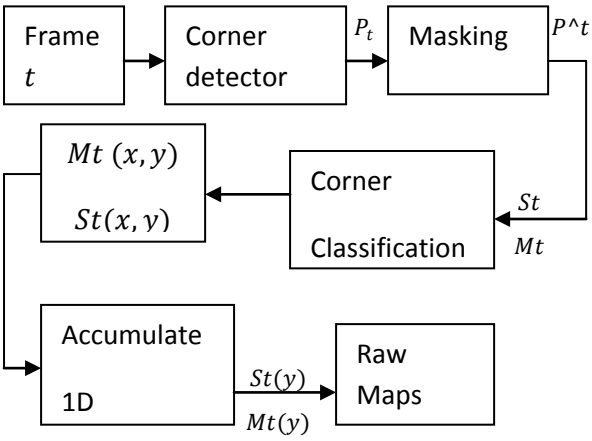

Fig 2: Sequential diagram of the processing performed for each frame.

It can be seen that dynamic points (green dots) are moving vehicles in motion. On the other hand, static (red dots) tend to concentrate on stooped vehicles except road marking.

Then, automatically avoid most of the static corners. Each pixel in $Asd\ (x, y)$ counts the number of static corners detected at the specific co-ordinates. Let, $St\ (x, y)$ act as a binary image that is built at instant time 't' by using the list of static pixels as,

$$St\ (x,y) = \begin{cases} 1, \text{ if } St \text{ contains a static corner at } x \ \& \ y \text{ co-ordinates} \\ 0, \text{ Otherwise.} \end{cases} \quad (9)$$

In a similar way, $Mt(x, y)$, act as a binary image that is built at time t by using list of dynamic pixels Mt as follows. Then, the accumulator of non-moving corners $ASd(x, y)$

$$ASd(x,y) = \sum_{t=0}^{Nt} St\ (x,y)$$

$$Mt(x,y) = \begin{cases} 1, \text{ if } Mt \text{ contains dynamic corner at } x\&y \text{ co-ordinate} \\ 0, \text{ Otherwise.} \end{cases} \quad (10)$$

Here, moving corners will also be of interest in the determination of whether the route is blocked. The processing of every frame ending by marginalizing the co-ordinates x in both $St(x, y)$ and $Mt(x, y)$ as,

$$St\ (y) = \int St\ (x,y)\ dx \quad (11)$$

$$Mt(y) = \int Mt\ (x,y)\ dx \quad (12)$$

5.2 Spatiotemporal Maps Formation

After, processing every frame, the next steps are to integrate the information from different time instants into a compressed representation. For that intention, spatiotemporal maps are assembled for both moving and non-moving corners by adding the column vector $St(y)$ and $Mt(y)$ of different time instants horizontally. Here, the vertical dimension of the raw maps corresponds to time. The spatiotemporal maps will be called as raw maps. Fig. 6 shows an example of raw static spatiotemporal maps $Smap\ (t, y)$

5.3 Filtering of Raw maps

5.3.1 Spatial filtering

Spatial filtering may be defined as a neighborhood size and an operation performed directly on pixels inside the neighborhood. Fig 3 shows, corners on a vehicle do not form a close framework. This indicates that corners on a vehicle might not present as a connected region, in a raw map. This problem is clearly visible in Fig 7. This region in the raw map can be connected if it is considered that there will be a maximum separation in the vertical dimension in between the corners of a specific vehicle. A precise look at Fig.7 shows that the presence of some long horizontal line that is not close to any other and have a height of 1 pixel. These horizontal lines can be easily discarded by additional filtering. Fig 8 shows the result of spatial filtering map of Fig. 7.

5.3.2 Temporal filtering

Temporal masking computes spatial location of a recent pixel and identifies at least one parallel reference pixel from a previous frame. After spatial filtering of static raw maps $Smap\ (t, y)$ numerous connected regions appear which correspond to parked vehicles as shown in fig 8. The horizontal dimension of these regions is directly related to time that they are remained static. Dynamic spatiotemporal maps are also temporally filtered here.

5.3.3 Combined raw maps

Raw maps of static corners can be adjusted both spatially and temporally. Therefore it is possible to combine them into a single image using different color for better visualization. The green region is used for dynamic map and the red region is for static map. Color spatio- temporal map shows in Fig 10. This diagram provides a quick view of what has happened in the lane during a time interval.

6. Raw MAP ANALYSIS

The information supported by spatiotemporal maps is much richer and allows additional information about the following to be mentioned;

1. **Parking time:** Measuring the horizontal dimension of red region in the raw map, it gives the parking duration of a parked vehicle.
2. **Frequent parking location:** Vertical position of red region in the raw maps is related to vertical positions on the original image. Fig.3 shows the results of Easy and Hard sequences using Harris corner detection method. Fig.4 shows the results of corner classification of Easy and Hard sequences using Harris detector. Fig.5 shows the results of combined spatiotemporal maps or color maps of four sequences using Harris. Fig.6 and Fig. 7.

7. EXPERIMENTAL RESULTS

This section presents the relative performance of Harris, SIFT and FAST corner detection. The experiments are carried out using MATLAB, and tested on i-LIDS dataset.

7.1 Harris Corner Detection

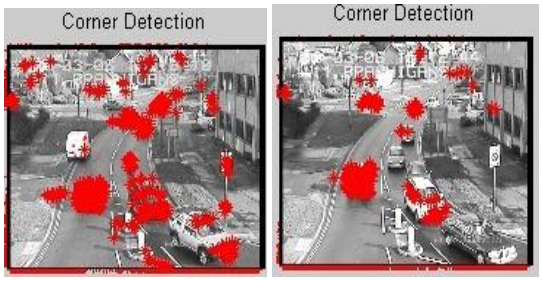

Fig: 3 Corner detection using Harris detector of Easy and Hard sequences

7.1.1 Harris corner classification

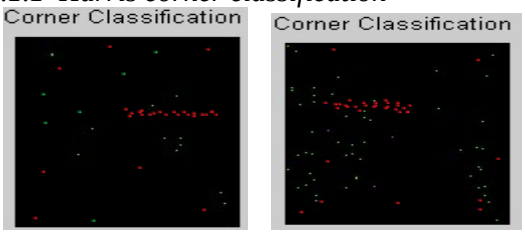

Fig 4: Corner classification of Harris detector for Easy, Medium, Hard and Night sequences

7.1.2 Combined spatiotemporal maps (color maps)

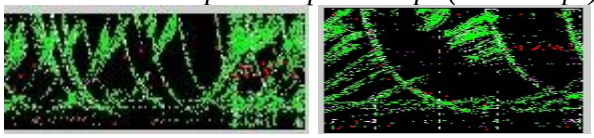

Fig 5: Color maps of Harris, Easy (upper left) and Hard (upper right) sequences.

7.1.3 Spatial filtering of static raw maps

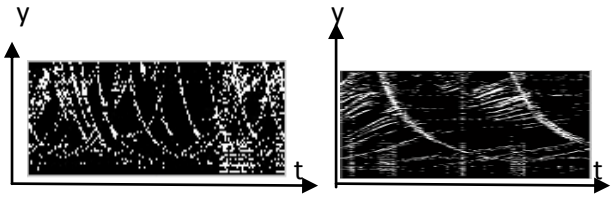

Fig 6: Filtered static raw maps for Easy and Hard sequences. X-axis- time, y-axis- frequent parking location

7.1.4 Spatiotemporal maps after spatial and temporal filtering

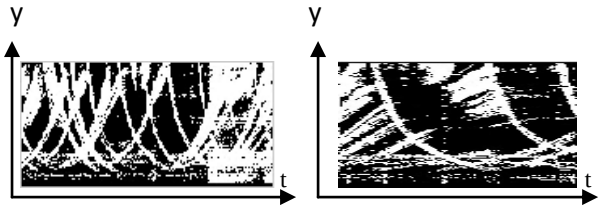

Fig 7: Spatiotemporal maps of four seq. X-axis- time, y-axis- frequent parking location.

7.2 SIFT Corner Detection

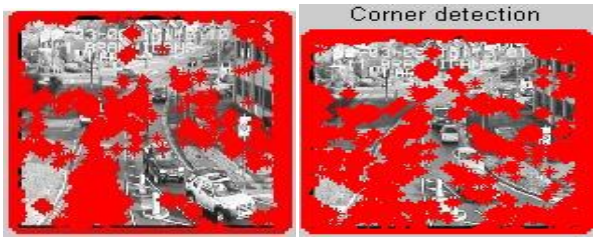

Fig 8: Corner detection using SIFT detector of Easy and Hard sequence

7.2.1 SIFT corner classification

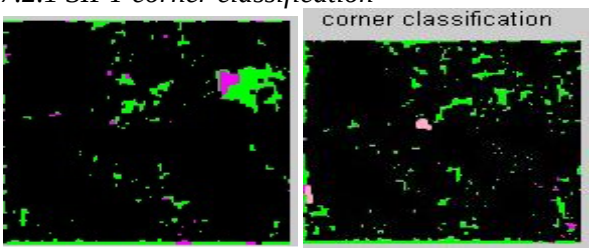

Fig 9: Corner classification of SIFT detector for Easy and Hard sequences

7.2.2 Combined spatiotemporal maps (color maps)

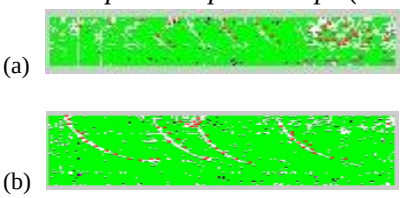

Fig 10: Color maps of Easy (a) and Hard (b) sequences using SIFT

7.1.1 Spatiotemporal maps after spatial and temporal filtering

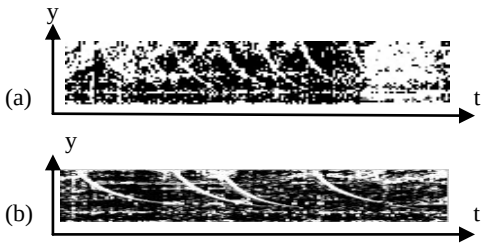

Fig 11: S- maps of Easy, Hard and Night sequences using SIFT. X-axis- time, y-axis-frequent parking location

7.3 FAST corner detection

Fig 12: Corner detection of Easy, Hard, Medium and Night sequences using FAST detector

7.3.1 FAST corner classification

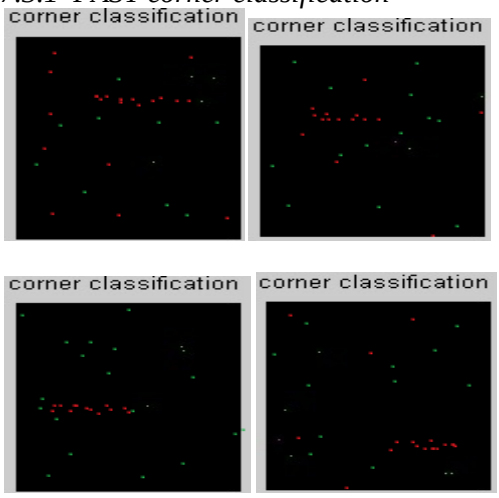

Fig 13: Corner classification of FAST detector for Easy, Hard, Medium and Night sequences

7.3.2 Combined spatiotemporal maps (color maps)

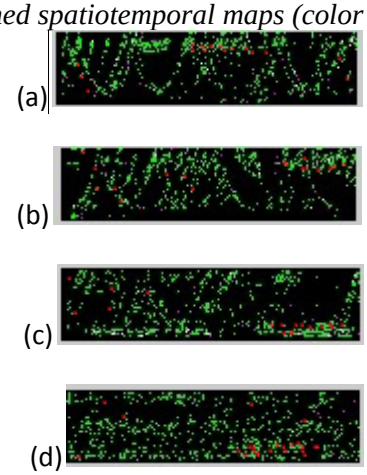

Fig 14: Color maps of Easy (a), Hard (b), Medium , (c) and Night (d) sequences using FAST detector.

Fig.12 shows the results of corner detection using FAST detector. Fig.13 shows the results of corner classification. Fig.14 shows the results of combined raw maps (static and dynamic) using FAST. Fig.15 shows filtered output of all sequences. Fig.16 shows the results of spatiotemporal maps using FAST.

7.3.3 Spatial filtering of static raw maps

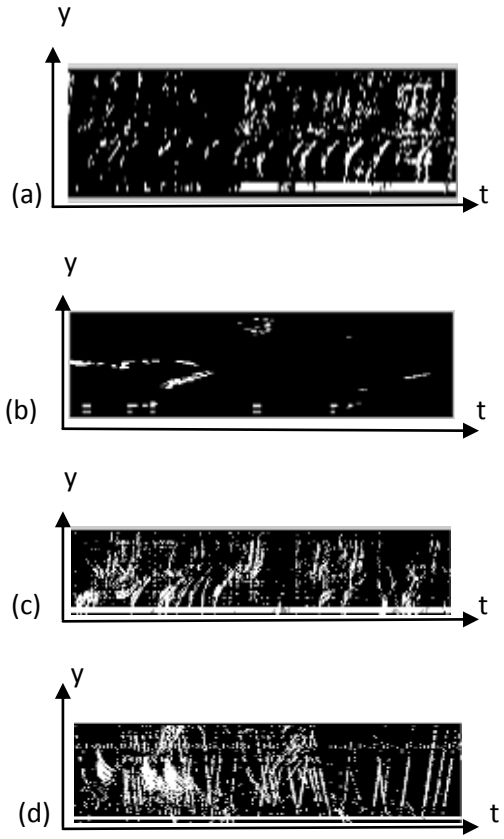

Fig 15: Results of the filtered output of static raw maps for (a) Easy, (b) Hard, (c) Medium and (d) Night sequences

7.3.4.Spatiotemporal maps after spatial and temporal filtering

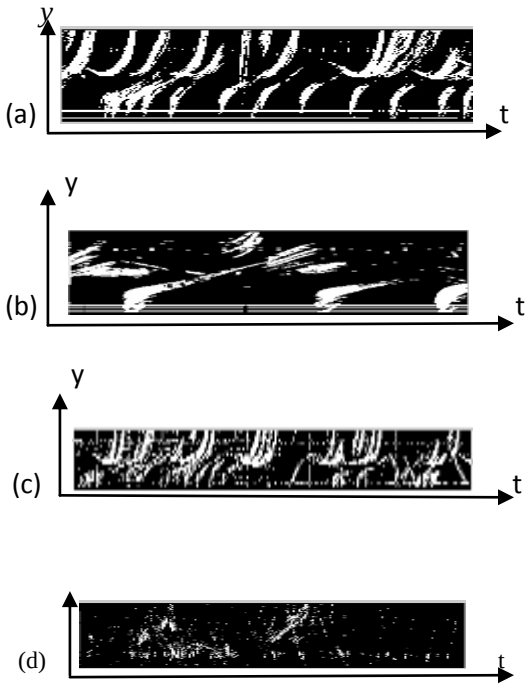

Fig 16: Spatiotemporal maps of (a) Easy, (b) Hard, (c) Medium and (d) Night sequences using FAST. X- axis- time, y- axis- frequent parking location.

8. DATA SET

The system has been evaluated using publicly available dataset i.e. i-Lids dataset for testing. i-Lids is an Imagery Library for Intelligent Detection System. This dataset consists of four sequences which are referred to as Easy, Medium, Hard and Night [13]. These four sequences are very short in duration, and each one consists of one or more parking event. The training dataset is a big collection of short-duration videos from three stages. Sequences have a resolution of 720*576 and 25 frames per second. They have been compressed using MPEG at a relatively high bit rate. In case of the parked vehicle scenarios, an event means that a new vehicle has been stopped for more than 60 second in a non-parking area.

8.1 Execution of Metrics for parked vehicle detection

To evaluate the execution of any system, it is important to define the metrics to be used. The following metric used for problem of parked car detection.

8.1.1 Hit Miss Metric

This metric measures the aptitude of the system to detect individual parked vehicles. Standard precision and recall measurements can be used for performance analysis,

$$P = \frac{\mathrm{Ta}}{\mathrm{Ta}+\mathrm{T}b} \quad , \quad R = \frac{\mathrm{Ta}}{\mathrm{Ta}+\mathrm{Tc}} \qquad (13)$$

Where,

1. '$\boldsymbol{T_a}$' - The number of seconds that the lane contained parked vehicles that were correctly detected (true positive).
2. '$\boldsymbol{T_b}$' - The number of seconds that the system indicates parked vehicle presence that was not true (false alarms).
3. '$\boldsymbol{T_c}$' - The number of seconds that the lane contained parked vehicles that were not detected (missed detection).

Table 1 shows the comparison of Harris, SIFT and FAST with total frames, stopped vehicle time and observable time. From the experimental results concluded that the FAST corner detector outperformed other two algorithms in terms of accuracy of detection of parked vehicle. FAST algorithm provides more accurate results compared to SIFT and Harris. FAST algorithm gives more accurate time compared to other two algorithms.

Table 1. Comparison of Harris, SIFT, and FAST with total frames, stopped vehicle time and observable time.

Algorithm	Video Sequence	Total Frames	Parked vehicle time	Observable time
Harris	Easy	5291	108 sec	98sec
Harris	Medium	3748	28 sec	60 sec
Harris	Hard	4361	72 sec	65 sec
Harris	Night	6104	145 sec	61 sec
SIFT	Easy	5291	108 sec	105 sec
SIFT	Medium	3748	28 sec	65 sec
SIFT	Hard	4361	72 sec	58 sec
SIFT	Night	6104	145 sec	63 sec
FAST	Easy	5291	108 sec	110 sec
FAST	Medium	3748	28 sec	69 sec
FAST	Hard	4361	72 sec	62 sec
FAST	Night	6104	145 sec	65 sec

Table 2. Detailed Results of sequences containing events

Algorithm	Video Sequence	P	R
Harris	Easy	0.72	0.68
Harris	Medium	0.70	0.66
Harris	Hard	0.68	0.63
Harris	Night	0.64	0.61
SIFT	Easy	0.79	0.74
SIFT	Medium	0.76	0.71
SIFT	Hard	0.72	0.68
SIFT	Night	0.69	0.62
FAST	Easy	0.86	0.82
FAST	Medium	0.84	0.81
FAST	Hard	0.79	0.74
FAST	Night	0.75	0.71

Table 2 shows the Harris algorithm identifies parked vehicle time moderately, with fewer false alarms compared to SIFT. Hence, Precision is better than SIFT but less P and R compared to FAST. SIFT algorithm identifies parked vehicle time better than Harris but less than FAST, but with more false alarms. FAST algorithm identifies parked vehicles time more accurately, hence more P and R.

Fig. 17 shows the comparison of Precision and Recall curves.

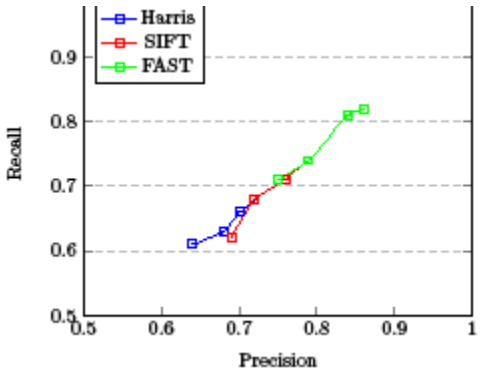

Fig 17: Comparison using Precision and Recall curves

9. CONCLUSION AND FUTURE SCOPE

In his paper, an approach for dealing with the problem of parked vehicle detection is presented. The solution is based on the analysis of spatio-temporal maps of static image corners generated using corner detection algorithms. Performance of different algorithms was compared over several criteria such as precision and recall, accuracy of output, false alarms etc. Harris algorithm identifies Parked vehicle time moderately, with fewer False alarms, compared to SIFT, so Precision is better than SIFT, but less P and R compare to FAST. SIFT algorithm identifies parked vehicle time better than Harris but less than FAST, but with more false alarms, hence Recall is more than Harris, though precision is less due to more false alarms. However, results of FAST algorithm outperformed other two algorithms, in terms of accuracy of detection of parked vehicle and calculation of duration for which it is parked, though it required more computational power. FAST algorithm raised fewer false alarms and shows less noise in output compared with other two, hence better P and R. So it is observed that, FAST algorithm provides more accurate and precise results compared to SIFT and Harris. The spatial and temporal filters help in increasing the robustness of the system. However, in some sequences precision and recall is less. So we need to work on improvement of precision and recall values and also need to decrease the time complexity.

10. REFERENCES

[1] A. Albiol, L. Sanchis, A. albiol, and J. Mossi. Detection of parked vehicles using spatiotemporal maps. IEEE Transaction on Intelligent Transportation System, vol. 12, No. 4, 2011.

[2] L. Maddalena, and A. Petrosino. Self organizing and Fuzzy modeling for parked vehicles detection. ICAR-National Research Council.

[3] A. Bevilacqua and S. Vaccari. Real time detection of stopped vehicles in traffic scenes. In Proc. IEEE Conf. Adv. Video Signal Based Surveillance. pp. 266-270, 2007.

[4] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A Survey ". ACM computer Surveillance, vol. 38, no. 4, pp. 1-45, Dec. 2006.

[5] P. Viola, and M. Jones. Rapid object detection using a Boosted cascade of simple features. Accepted Conference of Computer Vision and Pattern Recognition 2001.

[6] Z. Zhang, H. Lu, Xin Li, W. Li. and W. Yuan. Application of improves Harris algorithm in sub-pixel feature point extraction. IJCE, Vol. 6, No. 2, April 2014.

[7] Muthukrishnan R. and Ravi J. Image type- based assessment of SIFT and FAST algorithm. IJSP. Image Processing and pattern recognition vol. 8, no. 3, pp. 211-216, 2015.

[8] S. Borango, B. Boghossian, J. Black, D. Makris and S. Velastin. A DSP Based System for the Detection of Vehicles Parked In Prohibited Areas. In Processing IEEE Conference AVSS, Washington, DC, pp 260-265, 2007.

[9] J. T. Lee, M. Ryoo, M. Riley and J. Aggrawal. Real time illegal parking detection in outdoor environments using 1-D transformation. IEEE Trans. Circuis Sys. Video Technology, vol. 19, no. 7, pp. 1014-1024, 2009.

[10] R. Cucchiara, M. Piccardi. Vehicle Detection under Day and Night Illumination. Proceedings of 3rd International ICSC Symposium on Intelligent Industrial Automation (IIA 99).

[11] Mei H., A. Sethi, Y. Gong. A Detection Based Multiple Object Tracking Method. NEC Laboratories America, Cupertino, CA, USA.

[12] Konstantinos G. Derpanis. The Harris Corner Detector.

[13] i-LIDS dataset for AVSS 2007. [Online].Available: ftp://motinas.elec.qmul.ac.uk/pub/i-LIDS/.