An Intelligent Video Surveillance Framework with Big Data Management for Indian Road Traffic System

R. Balaji Ganesh
Ph.D. Research Scholar (Part time)
Manonmaniam Sundaranar University
Abishekapatti, Tirunelveli 627012

S. Appavu
Professor
K.L.N. College of Information Technology
Madurai 630611

ABSTRACT
Intelligent Transportation Systems (ITS) and Advanced Traveller Information Systems (ATIS) are the emerging areas of research. They focus with keen interest to solve the issues in traffic management and planning and designing infrastructure to meet the demanding needs of the general public. Many research articles focus on developing video surveillance algorithms for processing video data captured at real-time traffic scenes, but as there is a huge demand for more sophisticated software systems, complexity of the algorithms gets increased in terms of data storage and large scale processing. This research article focuses on refining a framework for large scale video analytics while incorporating the simple, light-weight aspects of a video surveillance algorithm, and makes an insight by adopting blob tracking based video surveillance algorithm for large scale video analytics. The proposed system uses hadoop' map-reduce function to clean and pre-process the hours of traffic video captured in the local site stores. It summarizes and transmits the key frames of the video data to the central computing server to analyses the video frames. The key frame differencing method has been justified as a pronouncing method for data preprocessing and cleaning. Further this system follows the Blob detection, Identification and Tracking using connected components algorithm to determine the correlation between the vehicles moving in the real road scene.

Keywords
Large scale video analytics, Road traffic surveillance system, Blob detection, Traffic density estimation, Video Systems and Analytics, Intelligent Video Systems (IVS) in big data

1. INTRODUCTION
Intelligent Transportation System is an emerging area to solve multiple transportation issues. Advanced Travelers Information System is a core area of ITS which gives amenities to the traveler by providing valuable information which can be used by the traveler for path selection, travel time forecasting and time management. As, all of the ATIS applications require large amount of information to conclude the decision for its users, the data procured by these ATIS applications are needed to be precise so that the concluded decision can be justified. On the other side, traffic surveillance systems aid commuters, provides valuable data for traffic cops, traffic infrastructure managers, enforces laws and encourages safe driving. Despite all these things the leverage of software systems for traffic management has emerged. But the focus is narrowed down to develop a core video processing algorithm, which serves the purpose of traffic monitoring [4, 5]. The question comes in our mind, whether the core video processing algorithm is applicable for large scale video analytics when the input data is available in cloud storage? Obviously we do not have a justifiable answer. When we need to bridge the gap between video processing algorithms and large scale video analytics, research work is directed towards amalgamation of these ends to a single working architecture. As traffic data continues to grow, the average monthly data has now reached 10 terabytes. Since data such as pictures and video are stored in different data centers in different divisions, it has become difficult to use [4]. Moreover, some traffic management facilities, equipment, and application systems, need to be integrated. When integration comes to a large scale, there is a need to distribute the processing capabilities of the system at the local site and at the server site, so that the local site will provide summarized video for processing and in turn the workload of the server gets optimized to accommodate data from different local sites simultaneously. The Intelligent Video Surveillance System (IVSS) is defined as "any video surveillance solution that utilizes technology automatically, without human intervention, process, manipulation and/or perform actions to handle either the live or stored video images" [14]. These IVSS systems support the traffic management personals with an unblinking eye support both in terms of monitoring and decision making. It also encourages commuters to follow traffic laws for a safe driving by providing advanced information to make decision on the travel time for path prediction, travel time forecasting, and emergency vehicle path selection. Generally speaking these IVSS are of twofold: One is through hardware technologies which are to be designed to acquire video data from the local site, pre-process the video data to adopt it to the requirements of the core video processing algorithm, transmit the reduced video data sets to the central server [1, 2]. On the other hand, algorithm based analytics needs to be happened at the server site. The state of the art in computer vision, pattern recognition and computational intelligent systems helps the researchers, technocrats to develop more sophisticated, highly supportive surveillance systems. As the degree of sophistication gets increased, the centralized management of reduced traffic data needs to be enabled, followed by optimization of massive data usage. If we do so, high degree of precious traffic data can be gathered to support decision making. This paper investigates the solutions of the above two challenges. First, we propose a Vision-based traffic flow data extraction algorithm to support multi-channel live traffic video streams in real-time utilizing video co-processors and with the aid of apache' hadoop servers. Second, we propose a vision-based algorithm to validate the effectiveness at the server site. On its first fold, this paper focuses on designing a data management and analytics system for homogeneous transportation data acquired from real time cameras at hot spots. This data

management system will form the base for designing large scale video data warehouse and analytics engine.

2. IVSS ARCHITECTURE

A visual surveillance system is required to be fast in processing with low cost and high reliability. This IVS Architecture is implemented with the help of Intel Xeon Video Co-processors. The purpose of IVSS is to acquire, store, summarize, reduce and transmit the video to server site. An abstract view of the proposed Intelligent Video Surveillance Systems is shown in figure 1. Intelligent video systems and services incorporate hardware system integration, management, and video processing to endpoint users [9]. The system proposed here merges the CCTV Camera System, IP Camera System, Cloud-based video systems, Portable Camera, Embedded Camera System to acquire video data for long hours of duration. Although Embedded cameras have limited processing power, memory and bandwidth, this architecture rely upon embedded cameras which combines video sensing, video processing and communication within a single device. The extension of this device can be made available to accommodate different range of communication systems including cloud based video systems. The design and implementation of hybrid IVSS will help to acquire, integrate information from multiple cameras simultaneously [2]. Both the centralized mode and distributed mode of IVSS are used to collect video from different types of camera systems. In centralized mode, the workload of the processing algorithm is concerned with the server site; whereas in the distributed mode, the low cost, memory optimized embedded cameras have the task of acquiring and pre- processing the video data to make it ready for further processing at the server site [14, 6]. IVS proposed here fuses the concept of embedded and distributed processing of video data to well establish a good distribution of workload across local and server site. This establishment is more effective, as the first fold of the system to acquire information for video processing algorithms happens at the local site and processing of the acquired data happens at the server site.

3. DATA ACQUISITION AND DATA MANAGEMENT AT LOCAL SITE

To ensure the distribution of workload, the data acquired from multiple camera systems are integrated. The integration of information from multiple cameras is essential for an intelligent video surveillance system [6, 7]. Number of studies has been made in multiple camera, multi perspective, multi modal camera based surveillance systems. In this work, we present a collaborative framework for the intelligent video surveillance systems. IVSS proposed here encompasses data acquisition and management including archiving at local site. After archival, key frames which summarizes the video data is communicated to the cloud storage for further processing by the server site. The video summarization is carried out using color Histogramming followed by key frame selection. The local sites rely upon Hadoop Distributed File System (HDFS) to archive the data at its own memory area, thus by reducing the workload of the server site. At the same time, server site has its liability to enquire for video data in HDFS using time stamp. Hence both the distributed computing and centralized computing systems are amalgamated to have an agile process framework as shown in figure 2. This framework can be re-established to adapt to the changing requirements of the traffic infrastructure managers.

3.1 Apache-Hadoop Engine

Apache Hadoop is a popular open-source framework for storing and processing large data sets across clusters of computers. The Map Reduce function available in Apache-Hadoop software library allows for distributed processing of data across multiple local sites. There by local computation and storage across cluster of machines is achieved. The Apache-hadoop Engine with its built-in libraries is used since it has the add-on facility to detect and handle failures at the storage level or at the application layer [7].

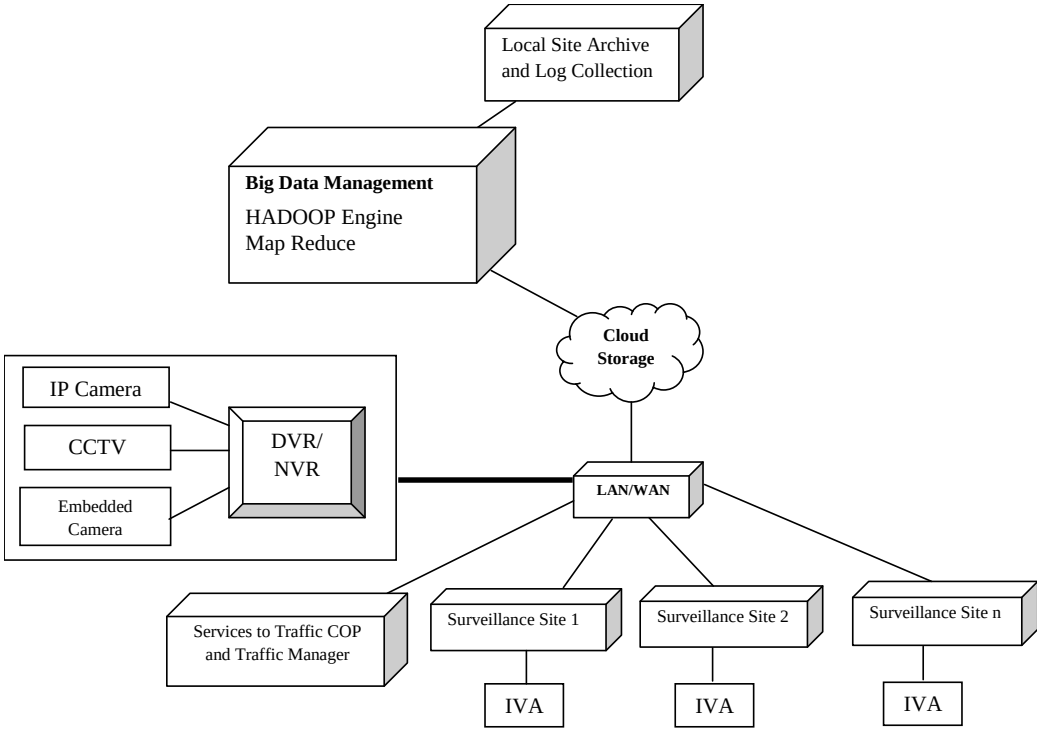

Figure 1. Proposed Framework for Intelligent Video Surveillance Systems

The architecture proposed in this article is suited to be implemented using Apache Hive data warehouse. This software facilitates querying and managing large datasets residing in distributed storage. Hive provides a mechanism to project structure onto this data and query the data using a SQL-like language called HiveQL. At the same time HiveQL also allow us to use traditional map reduce programming

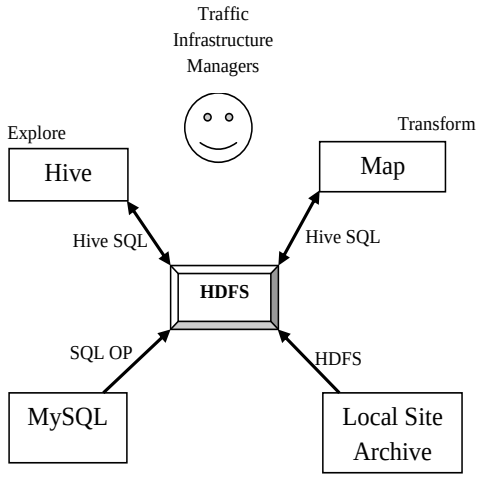

Figure 2. Implementation of HDFS for delivery and access of video data across local and surveillance sites

3.2 Map-Reduce

MapReduce is a framework for processing parallelizable problems across huge datasets using a large number of local sites, collectively referred to as a cluster. MapReduce allows for distributed processing of the map and reduction operations [7, 8]. Provided that each mapping operation is independent of the others, all maps can be performed in parallel. MapReduce has a functionality to sort video frames of petabytes in size in a limited time, so that it paves a way to acquire video frames of large scale from multiple cameras, multi perspective views at a large scale [9]. MapReduce is combination of 5-step parallel and distributed computation:

***Prepare Map( ) input*:** To specify the Map() to sort out the key frames from a large volume of frames across sites. Map() specifies the number of bins used to sort out the frames based on their color Histogram values and Texture based features.

***Map( ) Code*:** Map Code is written to sort the frames based on Color Histogramming and Texture features. The

variation across frames due to illumination has been overcome with the help of Map() code.

***Shuffle Map Output*:** Integrate the key frames from multi-view / multiple cameras and assign the unique key

value to all the frames from different cameras.

Run Reduce( ): Select the frames based on the least color Histogramming value.

***Produce MapReduced output*:** Store the reduced key frames in the cloud storage for further processing at the remote surveillance center.

The functions of MapReduce given in equation (1) are both defined with respect to data structured in (Frame No., Color Histogram Value(RGB), Texture) pairs. Map takes one pair of data with a type that satisfies the condition of least RGB value in one site, and returns a list of pairs in a different local site.

$$\text{Map}(k1,v1) \rightarrow \text{list}(k2,v2) \qquad (1)$$

In the Map function, the list of frames with least RGB value is produced, after that Reduce function collects similar list across different local sites with same least RGB value and group them together. This group of frames is defined as Key frames to be analyzed at the remote surveillance center. This reduce function is applied for the same timestamp across sites to generate a list of key frames as given in equation (2).

$$\text{Reduce}(k2, \text{list}(v2)) \rightarrow \text{list}(v3) \qquad (2)$$

3.3 Effective Data Pre-processing for Key Frame Selection Based on Color Histogramming Implemented Using Map-Reduce

In real world datasets, lots of redundant and conflicting data exists. The performance of a data cleaning and preprocessing algorithm is greatly affected by noisy information (i.e. redundant and conflicting data). These parameters not only increase the cost of cleaning process, but also degrade the detection performance of the proceeding algorithms [1]. The noisy data have to be removed to increase the efficiency and accuracy of the different surveillance algorithms [8]. This process is called as the reducing the dataset without losing useful information. In this method, we are going to address the ways to mine the useful information in the form of key frames from large scale video data sets captured in real world traffic. The key frame selection is based on the key frame differencing technique based on the frequency level information available in the video frames [1, 2]. The peak change in the RGB values of the pixel in each frame is taken into consideration. The Frame's RGB values are appropriately sampled in the 64 bin space which derives the key frame. By sorting each frame against the RGB Pixel value, the frame which most deviates from the rest of the frames is obtained. In the Color Histogramming technique the Mean R,G,B Value of each frame is calculated which then sorted and the deviation of R,G,B Mean value with other frames is listed out as shown in Table.1. The Frames with the highest deviation and lowest deviation are considered as key frames. This approach also reduces the constraints on the input requirements when compared to other techniques like edge histogram and wavelets and distance descriptor matching. The Method of Color Histogramming is easy to implement and cost effective in terms of both Space and Time complexity. Around 100 sets of video are applied over this algorithm to test the accuracy of the results. Color Histogramming can be applied to a large volume of data since it uses n-bin technique to group the pixels by which pre-processing is done at an earlier stage to avoid keeping unwanted information through all phases of the algorithm. Figure.3 shows the result of two key frames selected based on the color Histogramming technique using map reduce.

4. IVA AT SURVEILLANCE SITE

4.1 Overview

After data cleaning and pre-processing stage, the extracted key frames from multiple local sites are archived in the data store available in the cloud. These cloud store act as a medium of communication across multiple sites and as a source of information for traffic managers. The refined and cleaned information is sent to the Intelligent Video Analytics Engine which is present in the remote server location. This IVA encompasses the core video analysis algorithms with intent to track the moving objects across frames and analyze the activities. The proposed framework shown in figure 5 is

robust and efficient in terms of data cleaning, data pre-processing, data separation.

Table.1 Showing the list (V3) after mapreduce - Color Histogram values for the individual frames which are sorted based on deviation descriptor

Image Number	Red Mean Value	Green Mean Value	Blue Mean Value
img10.jpg	92.3166666	118.041025	109.735353
img11.jpg	92.3166666	118.041025	109.735353
img12.jpg	92.2833333	118.046153	109.711111
img14.jpg	92.2833333	118.046153	109.711111
img15.jpg	92.2385416	117.970256	109.670707
img16.jpg	92.2385416	117.970256	109.670707
img9.jpg	92.1916666	117.873846	109.586868
img8.jpg	92.1552083	117.869743	109.489898
img7.jpg	92.155208	117.869743	109.489898
img17.jpg	92.064583	117.710769	109.413131
img18.jpg	92.064583	117.710769	109.413131
img43.jpg	92.079166	117.817435	109.333333

Figure.3 Showing the Best 2 Frames which are more deviated from each other based on the Color Histogram Value

The different approaches available to do analytic tasks to extract useful information from the keyframes is detailed here with an insight to its adoption to big data environment [6]. Traffic analysis then leads to reports of speed violations, traffic congestions, accidents, or illegal behavior of road users. Various approaches to these tasks were suggested by many scientists and researchers [10]. One of the main aspects was to modify these algorithms to fit to real-time road monitoring processes, and as a consequence the prototype of system for traffic analysis was developed. Technically this system is based on stationary video cameras as well as computers connected to wide area network [13]. The application is utilizing image-processing and pattern recognition methods designed and modified to the needs and constraints of road traffic analysis. These are combined together to give functional capabilities of the system to monitor the road, to initiate automated vehicle tracking, to measure the speed [11, 14]. Image processing and object/pattern recognition of moving objects, chosen for the system, lead to complex mathematical, algorithmic and programming problems. Many articles have considered particular questions related: scene modeling, object geometry accounting, image contours processing. There is a lack of information on methods and algorithms used in digital monitoring technology, perhaps for commercial reasons [12]. The cell transmission model is a very simple model and yet it is able to recover most of the phenomena observed in real traffic. There are many other traffic flow models suggested in the literature that also reproduce traffic flow, in some cases with more precision than the cell transmission model. However, one of the challenges for real time on-ramp metering control consists on having calibrated traffic flow models [3]. Grid method is the most common method in road density analysis. However, the determination of grid size, position and orientation is rather arbitrary. It also fails to provide information within grids, and may bring obstacles to road selection process, since grid boundaries may 'split' the roads into several parts and ultimately give rise to the loss of information about connectivity [5, 4].

Fractal geometry method is devised by introducing self-similar fractal concepts. This method splits the whole study area into self-similar grids iteratively and the algorithm stops when the features within grids are homogeneous. It has the setback that the initial grid size exerts too much influence on computed road density, and the information lost at larger grid may not be recovered. Mesh density based on sub-region avoids several of the aforementioned setbacks. However, it neglects geographical characteristics of road networks and may not reflect information about the area each road is serving and its relative importance [5]. The problem of road monitoring as it is chosen in our research is presented as a sequence of independent processing steps intended to solve tasks logically connected to each other. These steps are in hand of the following order of algorithmic processing: Background noise removal, Blob Identification, Blob Coloring, Blob Tracking, Analysis of relative velocities of the Blobs. This paper mainly describes about the relative velocities of the moving blobs by utilizing the individual blobs velocity and draws the Probability Density Function (PDF) which then determines the density of the road surface.

4.2 Blob Detection and Tracking

An algorithm is proposed to detect each and every moving vehicle designated as blobs. A blob is defined to be a filled square and any square that can be reached from that square by moving horizontally or vertically. A blob is a connected region that can be found out by traversing through the pixels of the image. A kernel is experimentally designed to move over the image to find out all the connected components. A connected component refers to an object which has coherency of features when compared to the rest of the objects. The kernel shown in Figure.4 is moved over the image to find the connected components. For position S in the kernel neighboring pixels A, B, C & D are checked for the least label value. The pixel with minimal label value is added to the list of connected pixels. Similarly many groups of connected pixels are found out.

A	B	C
D	S	

Figure 4. Blob Detection Kernel used to analyze the Neighborhood Pixels for Connected Component

4.3 Algorithm

1. Read the Image Array
2. Initialize labelTable, xMinTable, xMaxTable, yMinTable,

   yMaxTable, massTable to 0
3. Initialize labelBuffer to the no. of pixels in the image
4. For Each Pixel S the nearest pixels are verified

   A B C

   D S

5. Assign Labels for each Pixel and enter it into labelTable
6. Iterate through pixels looking for connected regions.

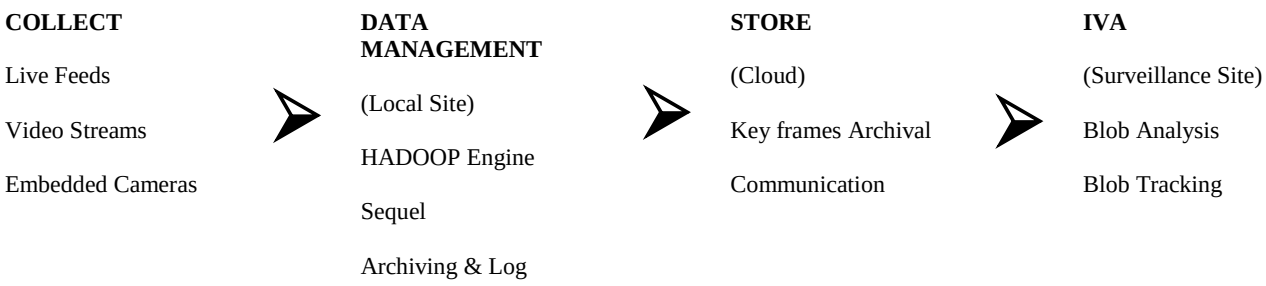

Figure 5. Framework for Large Scale Video Analytics

7. Find the neighbouring pixel with the least labelValue
8. If the found neighbour is a foreground pixel
    a. Set S Pixel Label value to min and assign the least labelValue to the pixel
    b. Update min & max X, Y values on the found neighbour into labelTable
    c. Increment the massTable value
9. If the found neighbour is not a foreground pixel
    a. Enter the min & max X, Y values into the xMinTable, xMaxTable, yMinTable, yMaxTable
    b. Increment the label value
10. Repeat Steps 3 to 8 for all pixels
11. Rearrange the labelValue of the each labelBuffer starting from 0
12. Assign color value to each labelValue (Blob)

The algorithm uses five values (xMin, yMin, xMax, yMax, MassTable) to identify the nearest and least neighborhood pixel. For each neighbor pixel, the labels are assigned and it is compared with the rest of the neighbors. The result of the above blob detection algorithm is shown in section 4.3. If the neighbor possesses Label value less than the current pixel's Label value, then the neighbor pixel is designated as the new pixel and its x,y position in the array are entered into the corresponding tables. If the neighbor is a foreground pixel, then the S value is set to the Least Label value and continued. From the Label Table, the objects are rearranged starting from the Smallest Mass to the Highest Mass and hence the blobs are numbered based on its mass value. To ensure the reliability of Blob Detection Algorithm, during the free flow traffic in which the vehicles can be passed through with greater velocity and to identify the missed vehicles between the two key frames, a Middle frame approach is suggested. The frame is taken into account, so that a moving object with higher velocity gets out in between the key frames and can be neglected in the further processing and the result is depicted in figure 7.

This algorithm is relatively simple to implement and understand, the two-pass algorithm iterates through 2-dimensional data. The algorithm makes two passes over the image: the first pass is to assign temporary labels and record equivalences among pixels and the second pass to replace each temporary label by the smallest label of its equivalence class. This algorithm is depicted using flow chart as shown in figure 6.

4.4 Blob Filtering

In this Section, the identified Blobs are labeled from 1 to n numbers. Each blob's X and Y position is calculated and the mass of each blob is calculated. The mass table in the above algorithm provides the value of each blob's spatial mass which can be calculated. The blob's list based on the algorithm in the previous section consists of all the blobs in the input frames. Blob filtering is carried out to filter out the stationary blobs which are basically the background pixels in the input frames. Three sets of conditions are applied over the blob list to find out the blobs which contain only the vehicles which are correlated between the two frames. Blobs which are stationary over a set of frames and Blobs that are having mass less than the specified threshold is calculated based on trial and error basis. Blobs whose change is very low, this is carried out to avoid erroneous blobs because of the internal movement of the camera and lightning conditions. The Blobs are Filtered using (3) and are sorted out. The filtered Blobs need to be tracked to find out the spatial movement of the blobs and their correlation between the frames. The correlation between the frames is found out to conclude whether the blobs have their movement independent or depend on some other blobs. This is found out by comparing the blobs that are having equal mass between the frames. The final list of blobs is analyzed for their spatial movement. A rectangle is drawn over the blobs to track it across the frame.

$$B_i = \begin{cases} \mathbf{b_i} \; where \quad ((\mu(b_i) > th1) \,\&\& \\ \qquad ((\delta(b_{i,f_a}(x), b_{i,f_b}(x)) > th2) \,\&\& \\ \qquad (\delta(b_{i,f_a}(y), b_{i,f_b}(y)) > th2)) \end{cases} \tag{3}$$

where, μ = Mass of the Blob, f_a = Frame 1, f_b = Frame 2, δ = Difference in the position of the object between frame 1 and frame 2, Number of Blobs Detected (i) = 0 to n, th1, th2 = Threshold. Bi is the list of blobs from the set bi which satisfies the threshold condition.

4.5 Blob Tracking

These moving blobs represent the vehicle motion in the images. It has asserted that the convex hull surrounding a vehicle in an image is a good approximation of the projection of a vehicle in the image. In the Blob Filtering, we exclude the blobs whose mass is less than 100 pixels. To characterize the moving blobs, we calculate the convex hull and calculate the centroid for the blobs. The area of each blob is calculated and

is indexed by the vertical location of its centroid in the image. This area is an indicator of blob size relative to its location in the image. Having located a vehicle in one image, the vehicle is tracked across images by enforcing co-linearity of the centroid of the convex hulls. The Figure 7 presents a representation of three convex hulls with centroid (x1, y1), (x2, y2), and (x3, y3).

4.6 Velocity Estimation

For a set of Two Blobs, Calculate the velocity of the blobs by fixing a reference point and calculate the displacement. The time difference between the frames is calculated using (4) followed by calculating the velocity of the individual blobs. A point in the previous frame, p, will be associated with a point in the current frame based on the minimum Euclidean distance, However, in order to determine the minimum distance, all the distance between reference point p and all other points c1 , c2, …, cn must be established.

$$E_d = \sqrt{(P_x - x_{cb})^2 + (P_y - y_{cb})^2} \quad ; \forall b \in B \qquad (4)$$

For every blob $b \in B_i$ in the image, the bounding box and centroid λ = (x_{cb}, y_{cb}) are extracted where x_{cb} and y_{cb} denotes the points connecting the diagonal of the convex hull marked over the blob. Centroid is computed by calculating the blob's spatial moment and is given in (5) and (6). The centroid will be used for tracking object.

$$u_{pq} = \sum_{x_b=0}^{m-1} \sum_{y_b=0}^{n-1} {x_b}^p {y_b}^q : (x_b, y_b) \in b, \forall b \in B \qquad (5)$$

$$x_{cb} = \frac{u_{10}}{u_{00}}, y_{cb} = \frac{u_{01}}{u_{00}} \qquad (6)$$

The ratio between the pixels and length in meters can then be defined by assuming that the ratio of world to pixel space is given as $\frac{\omega}{\zeta}$, which is calculated by determining the Systems DPI with relative to the input image. Further, we can derive the velocity in actual world unit using is given in (7) as

$$v_\tau = \frac{(\lambda_n - \lambda_{n-1})\frac{\omega}{\zeta}}{\tau_n - \tau_{n-1}} ms^{-1} \qquad (7)$$

4.7 Speed-Flow-Density Relationship

Speed, flow, and density are all related to each other. The relationships between speed and density are not diffi-cult to observe in the real world, while the effects of speed and density on flow are not quite as apparent.

Under uninterrupted flow conditions, speed, density, and flow are all related by the equation in (8)

q=k*v (8)

Where q = Flow (vehicles/hour)

v = Speed (miles/hour, kilometers/hour)

k = Density (vehicles/mile, vehicles/kilometer)

Because flow is the product of speed and density, the flow is equal to zero when one or both of these terms is zero. It is also possible to deduce that the flow is maximized at some critical combination of speed and density. Under Interrupted Flow Conditions, the Vehicle Speed, Density and Flow are related using a Gaussian PDF. The velocity of vehicles follows a normal distribution, the resulting PDF for the differential velocity v of two vehicles a and b is given in equation (9)

$$f_V(v) = \frac{1}{\sqrt{2\Pi}\sqrt{{\sigma_a}^2 + {\sigma_b}^2}} e^{-\frac{1}{2}\frac{(v-(v_a - v_b))^2}{{\sigma_a}^2 + {\sigma_b}^2}}$$

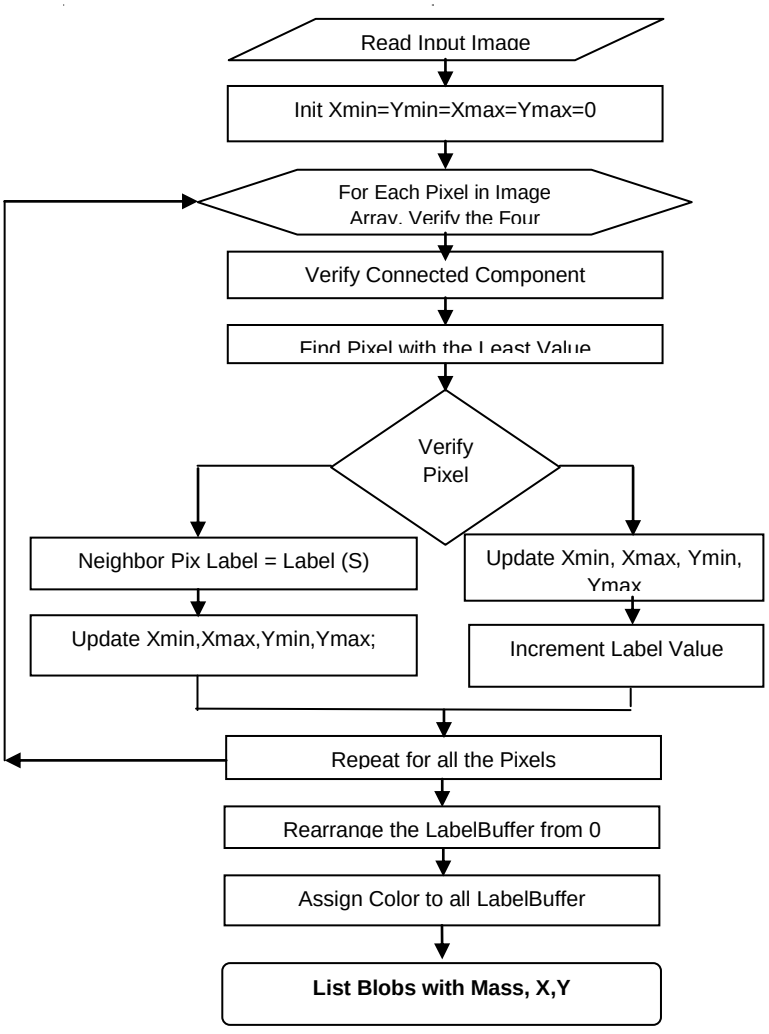

Figure 6. Flow Chart Depicting Blob Detection Algorithm based on Connected Pixel Region

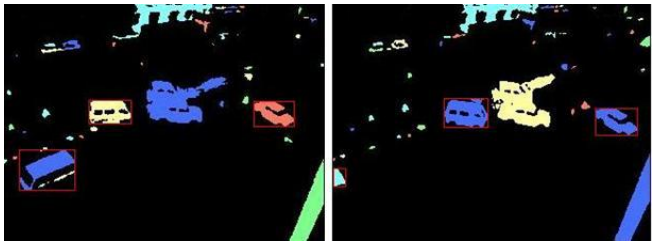

Figure 7. Blob Filtering Constraints Applied over the list of Blobs and Convex Hull Marked over similar Moving Blobs

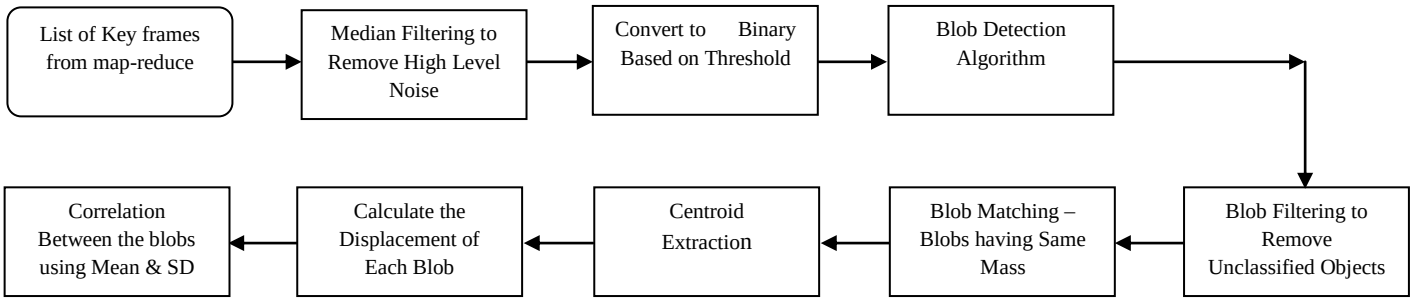

Figure 8. Architecture of the Intelligent Video Analytics at Surveillance Site

5. EXPERIMENTAL RESULTS

Experimental results of the proposed vehicle speed estimation are presented to demonstrate the real-time velocity estimation of random vehicles. The Data acquisition and Data Management at local site is implemented using MapReduce function of Hadoop, HiveQL and Java Framework to design analytics engine at surveillance site. The experiment was set up to monitor the speed of vehicles moving along the selected road lane. The real-world scene was recorded for 2 hours of long video at urban side which is map reduced to 30 minutes for analysis at surveillance site. The Proposed Method is tested over a range of videos taken at different scenarios like Bad Light, Rainy (or) Snow Days, Interrupted Traffic scene at urban road side, Uninterrupted Traffic Scene at National Highways, Video Taken at Tunnels. The Results of the different video types at various part of the Algorithm is given in Table.2. Since the key frame selection algorithm is used, the system is said to be in the state of intrinsic ie., the total input should be available before the start of the process. The overall performance of the Algorithm with 2 sets of Input data at various locations and conditions are shown in Table 3 and Illustrated in figure 9.

Table 2. Accuracy of the Piece-wise Algorithms under various dynamic video streams

Video Scenario	No. of Input Videos (120)	Frame Selection *	Blob Detection *	Blob Tracking*
At Bad Light	**38**	**91.8**	**95.4**	**89.7**
At Rain (or) Snow Days	**20**	**95.67**	**91.43**	**90.25**
Interrupted Traffic	**22**	**98.4**	**99.32**	**97.646**
Uninterrupted Traffic	**27**	**99.21**	**99.98**	**98.89**
At Tunnels	**13**	**98.12**	**97**	**89.91**

Table 3. The Accuracy of the Overall System at various Dynamic Conditions

Video Scenario	No. of Input Videos (120)	Accuracy of the Results (%)
At Bad Light	**38**	**92.3**
At Rain (or) Snow Days	**20**	**92.45**
Interrupted Traffic	**22**	**98.455**
Uninterrupted Traffic	**27**	**99.36**
At Tunnels	**13**	**95.01**

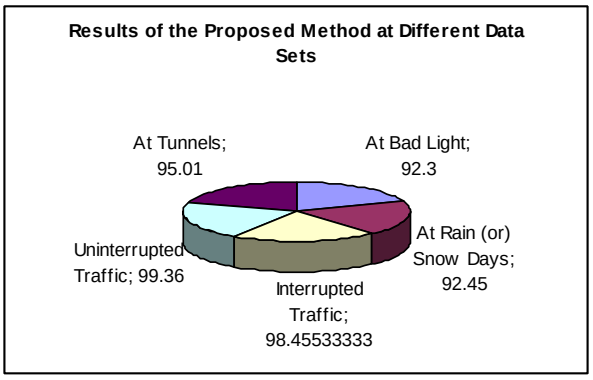

Figure 9. Results of the proposed method at different datasets

6. CONCLUSION

In this paper, we have devised an Intelligent Video Surveillance System to track and classify vehicles with the aim of replacing ILDs, particularly on highways. We have identified techniques and framework to manage hours of video data amounting to petabyte with map-reduce algorithms. We have observed that with the amalgamation of distributed and centralized computing parameters, video surveillance system will be optimized as a real time system for deployment at large scale. Our proposed system was

implemented in Hadoop' Hive Environment and Matlab Tools, and the experimental results thus produced is observed for Indian road traffic systems. As the framework proposed here is agile in nature, the system can be extended to include as many new features required for traffic managers. This work provides an abstract framework for the researchers and students to proceed towards large scale video analytics. Further research work can be directed to strengthen the IVA at server site by including core algorithms for vehicle classification, occlusion detection, congested traffic flow monitoring in urban roads. The method proposed in IVA deduces, whether the road traffic is congested or has free flow. As this algorithm also detects the number of blobs in the given traffic scene, it helps to make further analysis of the road traffic.

7. ACKNOWLEDGEMENTS

This work is supported in part by the University Grant Commission (UGC), New Delhi, INDIA -Major Research Project under grant no. F.No.:39-899/2010 (SR). This work resembles the work carried out by INTRANSE, an Intelligent Transportation Systems Endeavour, supported by Ministry of IT, Government of India.

8. REFERENCES

[1] Sadek, A., Smith, B., Demetsky, M.. A Prototype Case-based Reasoning System for Real-time Freeway Traffic Routing. Transportation Research Part C 9 (2001), 353-380.

[2] Yang Wang,, Real-Time Moving Vehicle Detection with Cast Shadow Removal in Video Based on Conditional Random Field, IEEE Transactions On Circuits And Systems For Video Technology, 2008

[3] G. L. Foresti, V. Murino, and C. Regazzoni, "Vehicle Recognition and Tracking from Road Image Sequences," IEEE Transactions on Vehicular Technology, vol. 48, no. 1, pp. 301-318, January 1999.

[4] C. Mallikarjuna, A. Phanindra; and K. Ramachandra Rao., Traffic Data Collection under Mixed Traffic Conditions Using Video Image Processing., Journal of Transportation Engineering ASCE / Apr. 2009

[5] Faro, D. Giordano, and C. Spampinato, "Evaluation of the Traffic Parameters in a Metropolitan Area by Fusing Visual Perceptions and CNN Processing of Webcam Images," IEEE Trans. Neural Netw., vol. 19, no. 6, pp. 1108–1129, Jun. 2008.

[6] Dunren Che, Mejdl Safran, Zhiyong Peng, "From Big Data to Big Data Mining: Challenges, Issues, and Opportunities," Database Systems for Advanced Applications, 2013

[7] Ling LIU, "Computing Infrastructure for Big Data Processing," Frontiers of Computer Science, 2013, Vol.7(2), pp.165-170, April 2013

[8] Nikolaos Mpountouropoulos, Anastasios Tefas, Nikos Nikolaidis, Ioannis Pitas, "Visual Information Analysis for Big-Data Using Multi-core Technologies," Advances in Intelligent Systems and Computing, Volume 297, pp 309-316, 2014

[9] Amir Gandomi, Murtaza Haider, "Beyond the Hype: Big Data Concepts, Methods, and Analytics" International Journal of Information Management, Vol. 35, pp. 137–144, 2014

[10] Cheng-You Cui, Ji-Sun Shin, Fumihiro Shoji, Hee-Hyol Lee," Traffic Signal Control Based on a Predicted Traffic Jam Distribution", IEEE Transactions on Artificial Life Robotics (2009) pp. 134–137

[11] Oscar Rosas-Jaimes, Luis Alvarez-Icaza," Vehicle Density and Velocity Estimation on Highways for On-Ramp Metering Control", Springer - Nonlinear Dynamics (2007) , Vol.49, pp.555–566

[12] E. Atkociunas, R. Blake , A. Juozapavicius , M. Kazimianec," Image Processing in Road Traffic Analysis", Nonlinear Analysis: Modelling and Control, 2005, Vol. 10, No. 4, pp. 315–332.

[13] LIU Xingjian, AI Tinghua, LIU Yaolin," Road Density Analysis Based on Skeleton Partitioning for Road Generalization", Geo-spatial Information Science 12(2): Volume 12, June 2009, pp.110-116

[14] Honghai Liu,Shengyong Chen, and Naoyuki Kubota, " Intelligent Video Systems and Analytics: A Survey", IEEE Transactions on Industrial Informatics, Vol. 9, No. 3, pp. 1222-1233, August 2013.