Selection, Detection, and Tracking of Video objects Based on FPGA

Zaki Y. Abid (BSc)[1], Thamir R. Saeed[2], and Sameir A. Aziez[3]

[1] **M.O.I&M**

[2]**Eelectrical Eingineering Dept., University of Technology**, Baghdad

[3]**Electromechanical Engineering Dept., University of Technology**, Baghdad

e-mail: zaki_assiedy@yahoo.com thamir_rashed@yahoo.com sameir777@yahoo.com

Received: 27/5 /2014

Accepted: 11/9 /2014

Abstract – This paper presents a moving object tracker for monitoring system which can be used in a smart city. Kernel density estimation (KDE) algorithm has been used for representing a background model, while a minimum distance between the current image and the background has been used to extract the foreground. Also, morphological operations are carried out to remove the noise regions and to filter out ambiguous areas. The performance has been evaluated by determining the true, false, and miss detections of an object area. The optimal results have been obtained by adjusting the morphological operation sequence to be (close > thicken) combination by which the true-hits are 14 out of 16 while miss-number is 2 and zero false-hits, While, the percentage hit ratio was 87.5% (14 out of 16). Also, the salt noise introduction in video reduces the hit number from 14 to 11 when it increases from zero to 0.5 percent of the total frame pixels. The accepted absolute error ratio (in morphological properties of the matched object) is kept at 0.05 for all tests. The implementation has been built by using a combination of two platforms, ISE 14.6(2013) and Matlab(2013a) platforms, to avoid the size weakness of XC3S700A-FPGA board.

.

Keywords – ***Smart city, Object tracking, Morphological Operation, Image Processing.***

1. Introduction

The recent significant growth of the electronic and communication devices and the complexity of the urban civilization lead to innovateof a smart city by smart autonomous systems [1]. The computer vision is an important application of the smart systems; it is used in a wide range of fields from human computer interaction to the robotics [2].

Automated video surveillance represents one of the important applications of smart systems; they are used in real-time traffic monitoring where they can analyze traffic flows, track vehicles, classify, identify and detect accidents by video cameras [3, 4, 5]. The main challenges are to develop fully automatic systems that require limited processing time and storage capacity, they do not require task-specific thresholds and tuning. This underlines the importance of algorithms that are computationally efficient, task-, operator-, and threshold-independent and are capable of detecting and tracking activities in a scene of observation [6]. Smart systems usually comprise moving object detection stage; in which a difference image of successive frames, in a video sequence, is analyzed to identify objects that are in motion [7]. The main objective of the visual tracking algorithm is to perform fast and reliable matching of the target from frame to frame by image analysis techniques [8], which identify (detect) the pixels that are different from the background and are thus suspected to be a part of a physical object new to the scene (foreground) [9]. Therefore, to decrease the number of misdetections and, consequently, improve the quality of the detections, an efficiently updated background model must be used [1]. There are many types of the background modeling, such as Hidden Markov Models (HMMs), which models the background variations by representing the changes in the scene with different states, and the Gaussians Mixture Models of (GMMs), which makes use of a mixture of several Gaussians to obtain an adaptive model of each image pixel. Although these models are able to provide high-quality detections in many scenarios, they fail in environments where the pixel statistics cannot be described parametrically. To enhance reliability of results in these environments, several strategies for detecting moving objects by non-parametric kernel density estimation methods have been established. Nevertheless, since these strategies have associated with high memory and computational expenses, they incorporate some simplifications that downgrade the quality of the background modeling. To handle this problem, the sample pixels that have been previously classified as foreground are discarded, and reference sample sets are continually updated to be used to model the background [1].

Responding to the growth of machine vision applications for the latest generation of electronic devices provided with camera platforms, many moving object detection strategies have been developed in recent years.

In order to reduce the number of misdetections the kernel module was used by [1] for background modeling in any moving object detection strategy. In [10, 11], a multi object moving background and removed the unwanted object motion based on morphological technique and cellular automata based segmentation was worked. Regarding the camera, [12] has presented an automatic foreground object detection method for videos captured by freely moving cameras for extracting a single foreground object of interest throughout a video sequence, even if the contrast between the foreground and background regions is low. As well [4, 13] have presented a technique for object

identification and tracking based on background subtraction with optimized threshold binarization. While [14] has presented a method for accurate motion detection, dynamic scenes without any prior information about moving object or dynamic scenes, by segmentation of estimated optical flow field, which is calculated by classical Horn-Schunck algorithm. [3, 15, 16] have presented a multi-object detection and classification by using a Bayesian inference method. Also, [17] worked on multi-object detection and tracking, but it overcomes the problem of illumination variation, shadow interference, and object occlusion by using a kernel-based clustering algorithm. At the intersection of the traffic system [18] use Kalman filter for detecting and tracking a moving object. [8] Has presented two approaches for object detection and tracking in video streams, a recursive density estimation (RDE) using a Cauchy type of kernel for visual detection and the use of the recently introduced evolving Takagi-Sugeno (eTS) neuro-fuzzy system for tracking the object detected by the RDE approach. While [19] performed tracking by correlation between the locations of moving vehicles. Whilst [20] has proposed a performance evaluation metric for tracking algorithms. The metric which uses the unique label assigned to an object and size of the object detected by the tracking algorithm. Under limited memory, [21] used an adaptive and compact background model that can capture structural background motion over a long period of time by detection multiple layers of background representing different background layers. [22] used motion energy for a background estimation algorithm based on the spatio-temporal analysis of motion information. A smart system with real-time moving object detection, classification and tracking and a stationary camera for both color and grayscale video imagery was presented by [23]. However, to discriminate between adjacent objects and object with background for tracking, [24] evaluated multiple feature spaces by applying log likelihood values computed from empirical distributions of object and background pixels with respect to a given feature. [25] has evaluated the performance metrics of object detection algorithms in video sequences to classify the types of errors into region splitting, merging or merge-split, detection failures and false alarms with a fixed camera. It is worth mentioning [5] has present an algorithm that adaptively estimate the background image by analyzing the image sequences based on local-regions of the input image rather than pixels, so that his algorithm can employ recursive least square (RLS) adaptive filter to estimate the non-constant statistical characteristics of the noise in the region and update the background in a fast and accurate way.

2- Object Representation and Motion Detection

Object size plays an important role in cluster algorithm, which is based on kernel density estimation (KDE) algorithm [17]. Where, the KDE is one of the most common techniques for modeling the background in video processing [6].

$$p(x(t)) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=1}^{r} k_{\sigma_j}\left(x(t)-x(i)\right) \qquad (1)$$

where N is the number of sample frames, r is the number of color bands and it equals three, $x(t)$ is the color intensity value of a certain pixel in current, t^{th} frame, $x(i)$ denotes the color intensity value of the the same pixel in i^{th} frame; $k\sigma_j$ is the kernel function with bandwidth σj in j^{th} chanel. Usually N>10 but not excessively large because the background

may not necessarily be static [6]. Once the probability density functions (pdf), of a pixel to be a part of the background is estimated, it simply compares with a pre-defined threshold value. If it is lower than this predefined threshold, it is assumed that this pixel is part of the foreground. KDE is very accurate but very expensive approach in terms of the memory and computation time. The most commonly used kernel function is Gaussian [8] which leads to:

$$p\big(x(t)\big) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\sqrt{2\pi\sigma_j^2}}e^{-\sum_{i=1}^{r}\frac{(x_j(t)-x_{j(i)})^2}{2\sigma_j^2}} \quad (2)$$

while the points that lie within the surveillance region and whose height values are above a predefined threshold are used to estimate the probability function [17] as

$$p_t(x) = \hat{p}_{t-1}(x) + \frac{1}{G_t\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\left(\frac{x-x_t}{\sigma}\right)^2\right) \quad (3)$$

where G_t controls the learning rate at time t,

$$G_t = Gain \times \frac{2}{1+\exp\left(-(cnt-\beta)/\lambda\right)}$$

cnt increases proportionally with time. The inflection points are controlled by β and *Gain*, while the gradient can be changed by λ. For object representation, the centroid and the rectangular shape to cover the object boundary can represent the object. After calculating the centroid, find the Width W_i and Height H_i of the object by extracting the positions of pixels P_{xmax}, P_{xmin}, P_{ymin} and P_{ymin} in *X*Y Coordinate [9]. There is another important factor, it is the camera noise. And due to it, the sum average difference (SAD) between two consecutive frames may be nonzero even if there is no motion. Therefore, we need to set a threshold to filter out camera noise. The threshold also determines the sensitivity of motion detection [25]. Then the motion detection algorithm begins with the segmentation stage where foreground or moving objects are segmented from the background and producing one or more foreground blobs [26].

Tracking is the problem of blob's position association in consecutive frames. While the trajectory can then be calculated for the object by connecting its position over multiple frames [9]. Then the bounding boxes of blobs with at least a certain area are detected. Then a recursive process is undertaken to join boxes into larger bounding boxes which satisfy $d_x < t_{ax}$, $d_y < t_Y$, where *dx* and *dy* are the minimal distances in *X* and *Y* from box to box, t_X and t_Y are the corresponding distance thresholds. The recursive process stops when no larger rectangles can be obtained that meet the conditions [3].

3- Foreground Object Detection

The consensus foreground objects template CFOT uses as a query image over a number of video frames, and this CFOT will look for similar image patterns in each frame within this period of time.

To determine the most similar image pattern, a similarity test based on sum of absolute differences (SAD) is exhaustively performed to search for a region in each frame which best matches the CFOT. The calculation of SAD between a CFOT and a video frame t is defined as follow [12]:

$$(x_c, y_c) = argmin_{x,y}\{\sum_{w=0}^{W-1}\sum_{h=0}^{H-1}|CFOT(w,h) - I^t(w+x, h+y)|\} \quad (4)$$

where CFOT is the average foreground pixel model and the size of the CFOT is W x H (width by height). It could be determined that the smallest SAD output

shows the best matched foreground object region, and thus the upper-left corner of this region will be recorded by (x_c , y_c). After objects detection, the tracking algorithm is used to associate the object positions in consecutive frames and provide a trajectory of each object [17]. Then the codebook is refined in a temporal filtering step by separating the code words contributed by the moving foreground objects from the true background codewords [27].

4- Foreground Extraction and Background Update

For continuous foreground extraction, we can take the average mean value of the minimum distance between the current image and the background model [28].

$$Dist_d = \min_{\forall k}(C_k^d - x^d) \quad (5)$$

where $\hat{p}^d(C_k) > \frac{1}{N_d}$ $d = 1,2,3$ and C_k is the background histogram centers that have a larger probability (p) than $1/N_d$, and the foreground can be obtained by comparing D_{istd} with B_d as follows:

$$\left.\begin{array}{l} if\ \left(\frac{\sum_{d=1}^{3}(|Dist_d|}{1+Grad_{t-1,d}}\right) > \sum_{d=1}^{3} B_d \times \gamma) \\ \\ then,\ FG = 1,\ (foreground) \\ \quad Grad_{t,d} = (G_t - 1) \times \frac{Grad_{t-1,d}}{G_t} + \\ \quad w \times \frac{|Dist_d|}{G_t} \quad d = 1,2,3 \\ else,\ FG = 0,\ (background) \\ \quad Grad_{t,d} = (G_t - 1) \times \frac{Grad_{t-1,d}}{G_t} \\ \quad + |Dist_d|/G_t \quad d = 1,2,3 \end{array}\right\} \quad (6)$$

where FG is the pixel's identity (0/1 corresponding to fore/background), $Grad_{t,d}$ is the average of an absolute of $Dist_d$ at time t, w is a weight to control the speed of adaptation for the environments (0.1 to 0.3), and γ is a weight for the threshold (1 to 1.5).

Error rate of the object mask is adopted to present the effectiveness of the algorithm; it is defined as [29]:

$$Error\ Rate = Error\ Pixels\ /\ Frame\ Size \quad (7)$$

where *Error Pixel*s is the number of pixels from which the obtained object masks is different from the reference alpha plane.

5- Morphological operations

Morphological operations are usually preformed to remove noise regions and to filter out a strange region, unreal. Two morphological operations are widely used, they are opening and closing. Opening operation generally smoothes the contour of an object and eliminate thin protrusions. Closing operation is done to eliminate small holes and filling gaps in the contour [10, 11]. Figure 1 shows a sample for each step of the morphological reconstruction process.

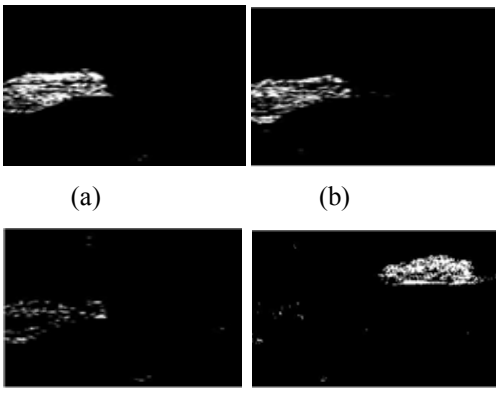
(1a) Input foreground

(1b) After hole-filling

(1c) Reconstructed blob (after opening)

Figure (1): Morphological reconstruction

Figure 2 shows how the adaptive implementation has integrated a recently stopped vehicle eventually into the background. In Fig. (2a-to-d), the fore-ground pixels corresponding to a recently parked car disappear gradually.



(a) (b)

(c) (d)

Figure (2): Static objects integrating with background

The morphological reconstruction treat-ment eliminated the false foreground detections and smoothed the object contours. An important morphological property that can be used for two-dimensional spaces is the Euler number which represents the difference between the number of connected components and the total number of holes in a binary image [21]; it also extracts the topological properties and global description of regions in an image. The topological features are unaffected by common transformation and rotation [30, 31].

6- Image Mapping

With object detection operation, area centroid position can be calculated and recorded; this treatment continues until the object disappears from image frame. After that, we can get the total displacement of the object.

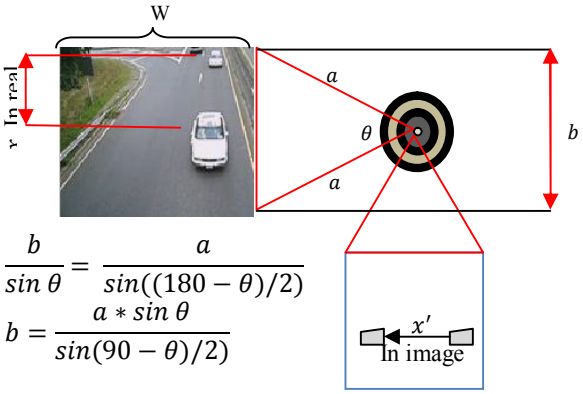


Figure (3): Image mapping

However, this displacement, in the image domain, is measured in pixel unit. For real movement, we may use the following relation:

$$x = \frac{b}{w} * x' \qquad (8)$$

as shown in Fig. (3) [4]. So, to get ***b*** we just need to know camera span ***θ*** and the distance from camera (***a),*** then velocity can be measured by starting a timer at object detection in the video sequence. The timer expires with the disappearance of the object. From this timer we get time span during which the object was in movement [4].

However, the major problem with this kind of application is the difference between the ground truth and detected objects area. Outdoor environment (shadowing and lighting), segmentation and morphological processes can produce such a difference.

So far, however, there has been little discussion about this problem. While in this paper, the problem has been analyzed and solved by treating each pixel in the

ground-truth as object/non-object and the output pixels as detected/non-detected [32], as follows:

$$U_{G^{(t)}} = \bigcup_{i=1}^{N_{G^{(t)}}} G_i^{(t)} \quad (9)$$

$$U_{D^{(t)}} = \bigcup_{i=1}^{N_{D^{(t)}}} D_i^{(t)} \quad (10)$$

where:

$G^{(t)}$: set of ground truth objects in frame *t*.

$D^{(t)}$: set of output boxes in frame *t*, produced by the algorithm.

$U_{G^{(t)}}, U_{D^{(t)}}$: spatial union of $G^{(t)}$ and $D^{(t)}$ respectively.

$N_{G^{(t)}}$ *and* $N_{D^{(t)}}$: the respective values of $G^{(t)}$ and $D^{(t)}$ in frame *t*.

7- Object Tracking Algorithm Based on FPGAs

FPGAs are frequently used in image processing algorithm applications due to their ability to handle large amounts of incoming and outgoing data stream, and their ability to execute multiple operations in parallel on that stream of data [33]. Xilinx-ISE14.6 (2013) provide a special tool that mixes between two platforms, ISE14.6 (2013) and Matlab 2013a platform; Matlab2013a provides components for FPGA that invoke VHDL code of filtering, segmentation and comparison from Xilinx platform to avoid the size weakness of XC3S700A-FPGA board. After the project runs successfully, we add a hardware component "hwcosim" (hardware co-simulation) from system generator, called Xilinx JTAG Co-Simulation block, which allows to perform hardware co-simulation using JTAG and a platform USB; JTAG Co-Simulation interacts with the FPGA hardware platform during a simulink simulation.

Contrariwise, when the data is read from co-simulation block's output ports, the block reads the suitable values from the hardware and pushes them on the output ports after connecting the target board to PC; thus they can be interpreted in simulink [34, 35].

8- Proposed Work

8.1 Proposed work Algorithm:

The proposed work algorithm is of two stages, each stage has many steps as shown in block diagram of Fig. (4), and they are as follows:

First Stage (first camera)

1. Detection of any (each) moving object enters the specific camera site. Then, bounding and numbering these detected objects.
2. Interactive selection of an interested object and bounding it by a different color box.
3. Extraction of selected object information and sending it to the second stage (of the second camera).

Second Stage (second camera)

1. When the objects start to appear in the second camera site, the optimal scaling comparison between the received data and the scaling data of appearing objects will start.
2. When the desired object enters the camera site. The algorithm recognizes it, then, it will be bounded by a box and its path will be shown as well.

There are many reference morpho-logical properties of the selected object. They are transmitted as a property vector to a second tracking camera site. The tracking algorithm will compare the received property vector with the incoming moving objects' data to determine the tracked object (that has been selected in the first camera site).

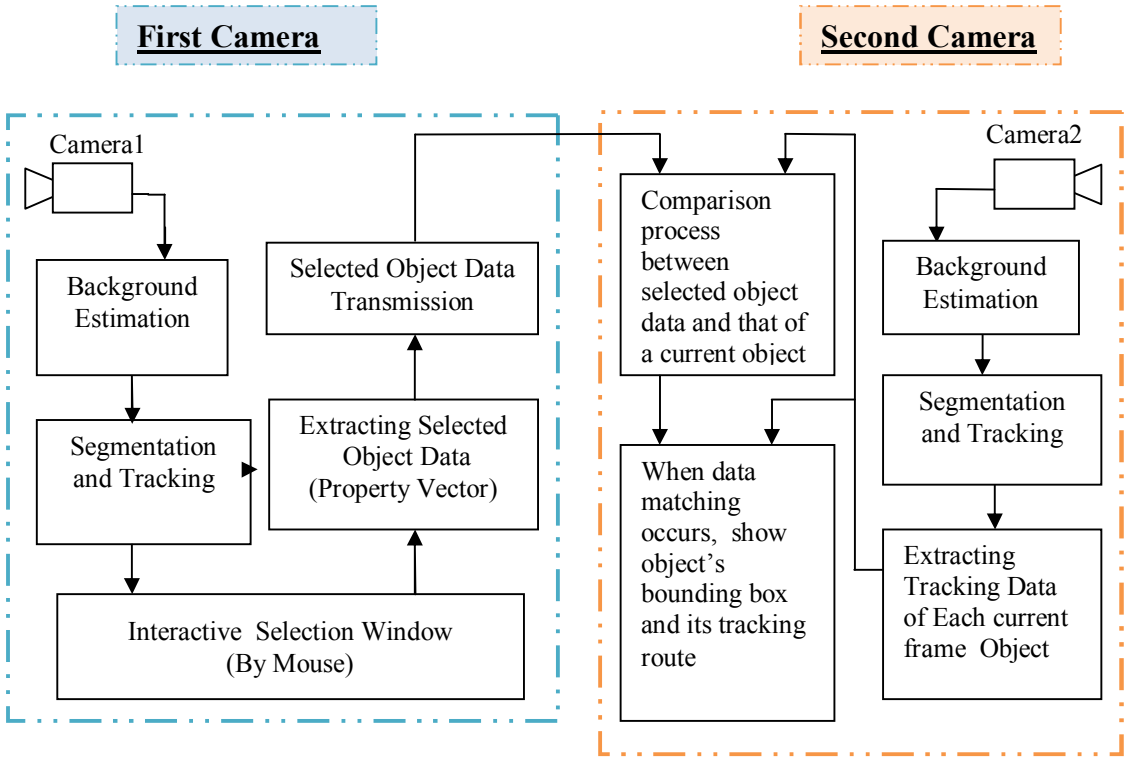


Figure (4) Proposed work block diagram

8.2 Results and discussions:

8.2.1Object Selection and Tracking

When the objects enter the first camera site, the algorithm tracks all objects of current frame; it alsogives each object an ID number and assign its tracking data to an individual data structure, but no information is displayed, on the attached UI (user interface), before an object is selected, as shown in Fig. (5a). At the time an object is selected (Object # 2 in this case), its ID number, bounding box, and path are added as augmented data to the frame and will be shown in red color, as shown in Fig. (5b), While its tracking data is displayed in the yellow zone of the same UI.The object is treated this way until it leaves the camera viewing area (being outside the region of interest ROI) as shown in Fig. (5c). The stored information about selected objects will be transmitted to the second camera tracking system.

(5a)

(5b)

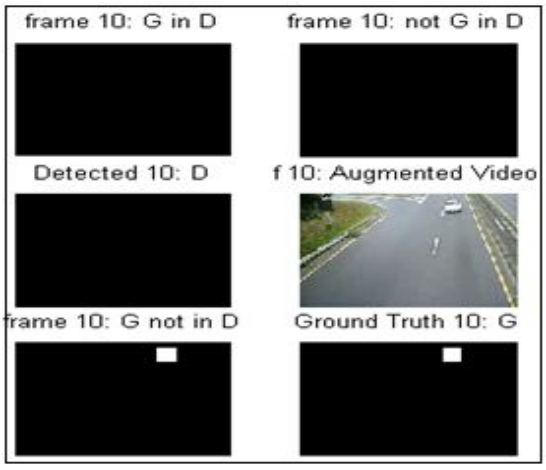
(5c)

Figure (5): Interactive object selection

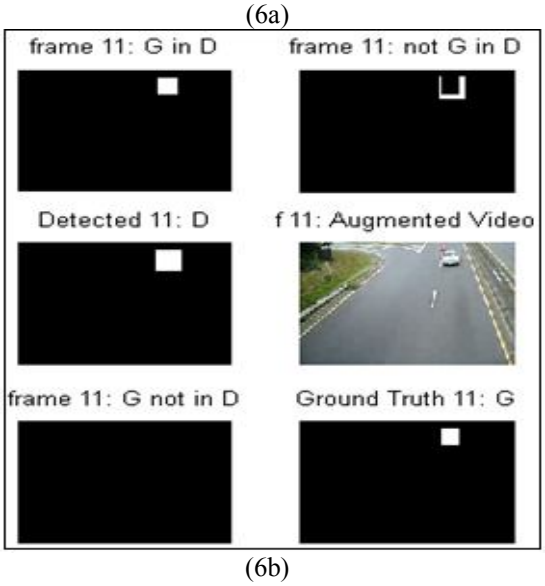
The accuracy of the transmitted object properties depends on the accuracy of object mapping and the relation between its ground-truth and detected-truth.

Figure (6a) shows the object before entering the viewing site; it is clear that the ground-truth has not been detected (G not in D).

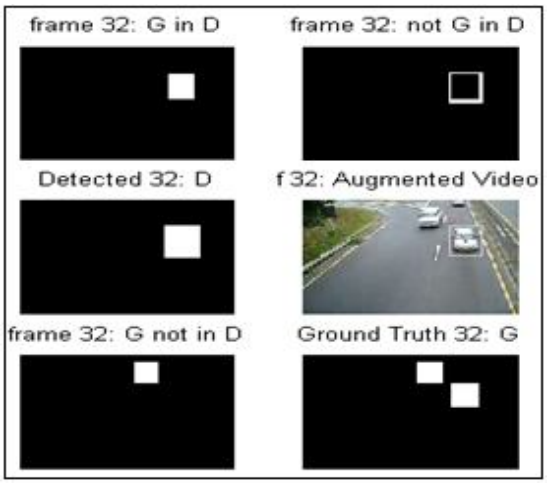
In Fig. (6b), there is a shadow and the segmentation stage has detected it as part the moving object area (not G in D). However, in Fig. (6c) some of the ground-truth has not been detected because its color, at that moment and light angle, was the same as the background. Whilst Fig. (6d) shows a special case when one ground-truth splits into two objects.



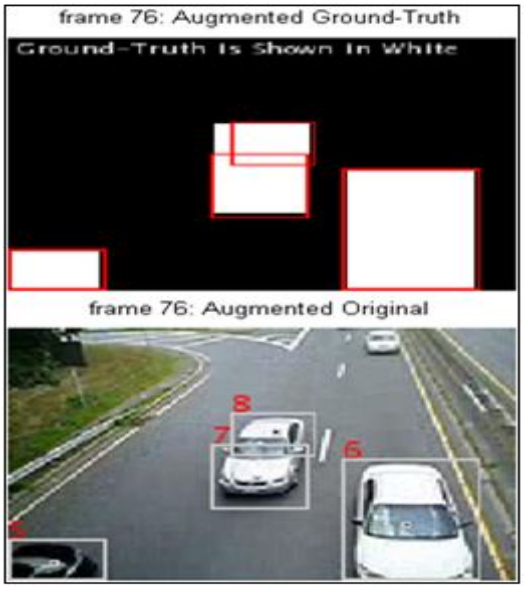
(6a)



(6b)



(6c)



(6d)

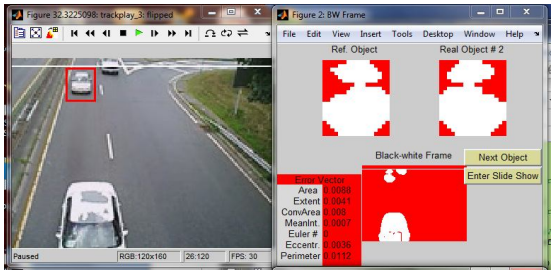
G: Ground- truth area.
D: Detcted area.
G in D : Detected ground-truth.
G not in D: Undetected ground-truth.
not G in D: Not ground-truth artifact

Figure (6): Differences between ground-truth and detected objects

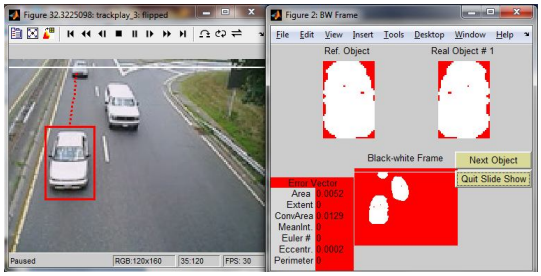
In order to get best detection and identification results, information is extracted for each object in a real-time manner using the same morphological treatment in each camera tracking system.

8.2.2 Object Detection and Tracking

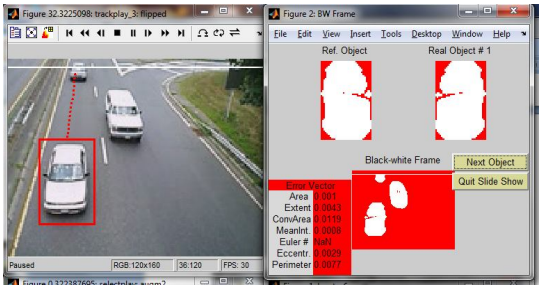
When objects pass in front of the second camera, the algorithm compares between each object's measured properties and the received reference properties from first camera, while displaying the properties of the object being examined. When each corresponding measured and reference property elements are close to within an acceptable absolute error ratio (0.05), the object is declared matching, bounded with a red box and its path is shown in dotted red line. Also, the sub-figure, where object's properties are displayed, is changed to red, as shown in Fig. (7).



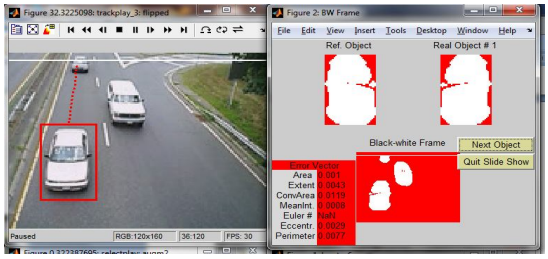
(7a)



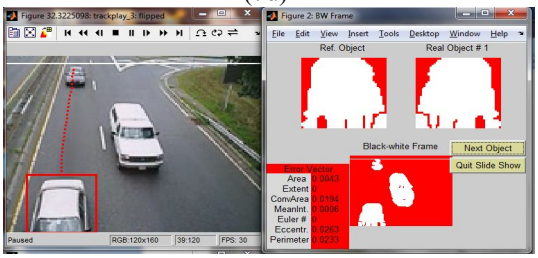
(7b)



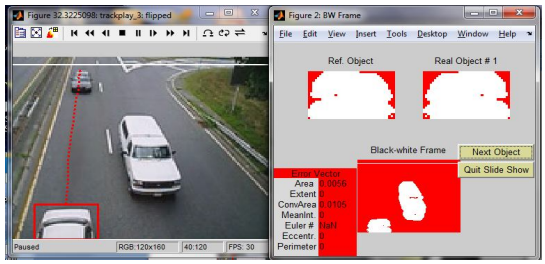
(7c)



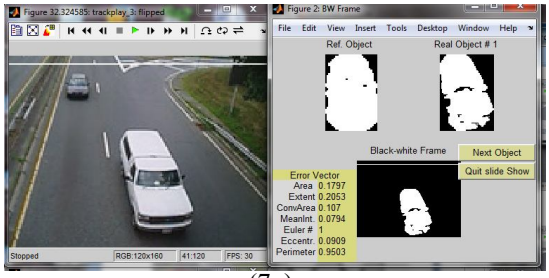
(7d)



(7e)



(7f)



(7g)

Figure (7): Object detection and tracking.

As the tracked object exits the second camera site (being outside the region of interest ROI), the red color changes to dark yellow again, Fig. (7g).

Tracking process at both camera sites is performed in a synchronous manner, i.e., there is no pre-tracking data since objects are supposed to appear for the first time.

Real-time tracking data and black-white image of object on the screen are shown, one at a time, on the attached figure.

Also, the input video has been left-right flipped so that algorithm immunity against object orientation can be examined.

8.2.3 Ground-Truth and Detected Areas

Figure (8) shows the detected-to-ground area-ratio; the points where the curve touches the zero line are where the detector failed to detect any object. While the discontinuous curve segments represent zero ground objects and zero detected ones; so the detector did not fail at these frames since there were no objects to detect.

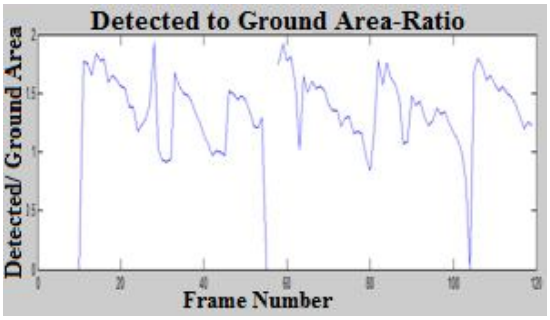


Figure (8): Truth and detected objects area.

All results have been automatically computed by "Performance Evaluation" specially developed software package.

8.2.*4 Effect of Morphological Operations on Object Recognition*

The function “bwmorph(bwImg,’fill’)” fills the holes of single pixel; while “imfill(bwImg)” function fills all holes of any size which tend to reduce the Euler number. When an object is treated by this function, it will have no holes any more, losing this way an important feature that may be used to recognize a pre-selected object. Thus, other objects can be erroneously detected as the selected one, making recognition process not as good as desired.

If we let tracked objects keep more of their holes, a good discrimination ability will be added to the algorithm and the whole press gets a strong boost, as is evident in Table (1).

Table (1): Morphological operations, close > fill

function	**imfill**	**Bwmorph (’fill’)**	**Selected Object ID**
Hit number	12	12	2
Miss number	4	4	2
False hit number	3	0	2
Total frames	16	16	2

Obviously the recognition process has got improvement, as there are no more false detections, yet the hit number is still not as high as enough to declare a good recognition process. The best results can be achieved by modifying the morphological operation sequence to be (close > thicken) combination rather than (close > fill) as in Table (2).

The results of Table (2) confirm the crucial role of morphological operations in a successful recognition process that may get dramatic improvement when “fill” operation is excluded from morphological operations and more region holes become a strong feature of a black-white tracked object.

The average hit ratio of total object occurrences, number of frames in which object has been detected, was 91.4 %.

Table (2): Morphological operations, close > thicken

Tracked Object ID	Hit number	Miss number	False Hit Number	Total object Occurrences
1	16	1	1	17
2	16	0	1	16
3	13	0	0	14
4	10	3	0	13
5	18	1	0	19
6	15	1	2	16
7	14	2	0	16
8	1	0	0	1
9	16	1	0	17
10	11	3	0	14
11	14	1	0	15

8.2.5 Error testing:

The error in object's morphological properties due to environmental (lighting) and processing (morpho-operations) reasons is unavoidable, i.e., detecting and identifying an object by one hundred percent property matching in all cases and circumstances is not possible.

As a good compromise, the properties may be accepted as matched within an accepted absolute error-ratio (*error*) which has been determined empirically by examining the hit, miss, and false curves of Fig. (9). The *error* value has been chosen corresponding to global best results zone (where false and mis-detections were at minimum while hit number was at acceptably high value).

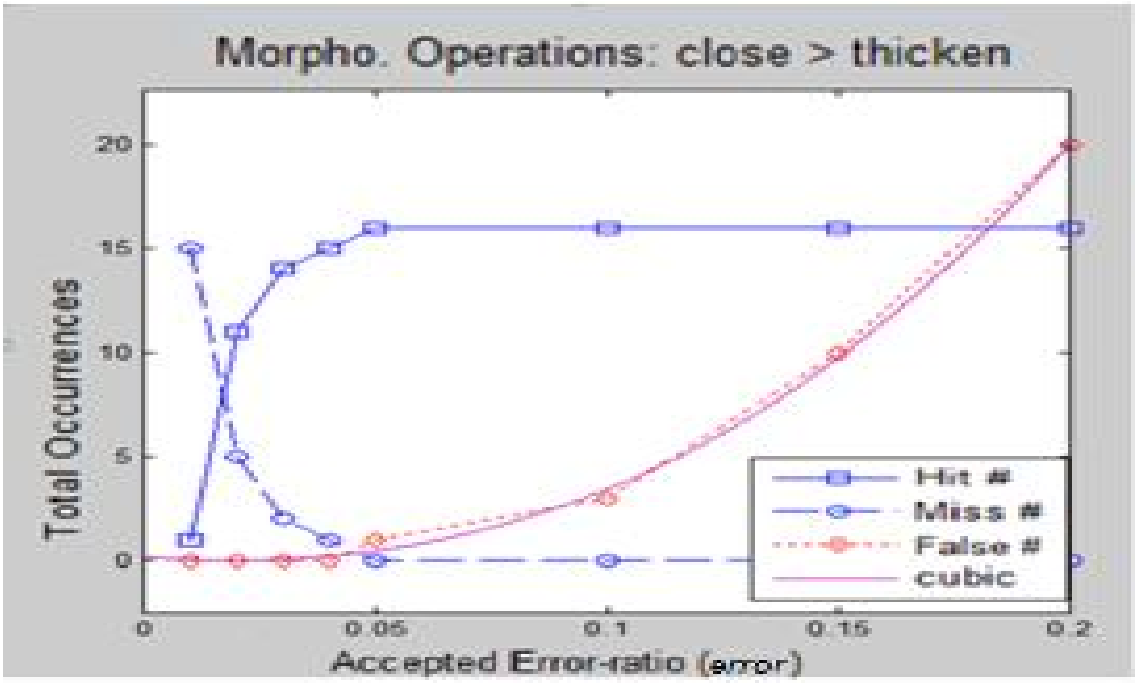


Figure (9): Hit, Miss and False detections against accepted error-ratio (error).

error has been set to (0.05).

Absolute error-ratio of an object property has been calculated as follows:

$$\text{Absolute Error-Ratio} = \text{Absolute Value}\left(\frac{\text{Current Value} - \text{Reference Value}}{\text{Reference Value}}\right) \quad (11)$$

The accepted error-ratio (*error*) has a strong impact on object recognition process. As shown in Fig. (9), calculations were performedfor a single example object in the whole video are counted for a single object using fixed optimized morphological operations. Fig. (9) shows that the hit number is best when *error* equals or more than (0.05), so is the miss number; while false recognition is best below (0.04) error-ratio; so, choosing the value (0.05) for *error* would give global best recognition results.

8.2.6 Noise testing :

Black-white image has been used to detect and recognize a selected object (interested object); to examine the algorithm immunity against noise, an image with added random noise would look like the samples in Fig. (10).

This type of noise is called 'salt and pepper' noise, to which, the algorithm shows a significantly high sensitivity since it (the noise) highly affects morphological operations by changing object properties like area, perimeter, and holes which are considered as object features. Statistically, adding 0.5 percent (0.005) of salt and pepper noise to an image caused hit ratio reduction from 14 to 11 out of 16. While adding 5 percent (0.05) of noise reduced hit ratio to be close to zero percent, as in Figure (11).

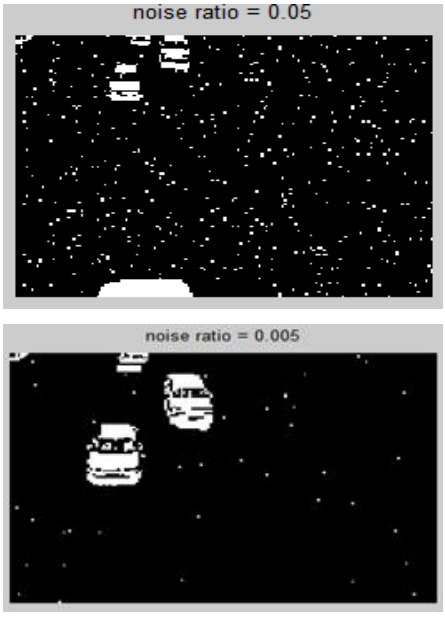


Figure (10): Adding random noise

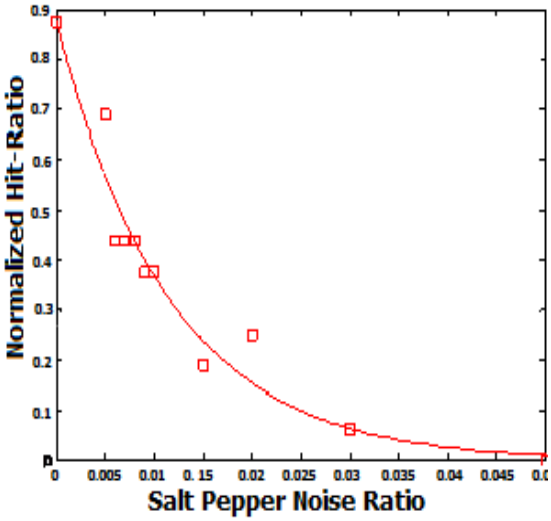


Figure (11): Hit number versus noise ratio

8.2.7 Processing Time-Segments and Related variables:

The frame processing time T_{total} is an important factor; it determines the maximum frames per second FPS_{max} that can processed by the algorithm; it is the sum of many time-segments taken by different steps.

The function "locate" searches video acquired frames for the wanted object; it comprises many processing stages and hence, its total processing time ***Ttotal*** is composed of detection time ***Tdet*** which is the time taken by segmentation part of function and ***Trecog***, object recognition time segment. Also, there is the number of objects in each frame ***Nobj,*** a logical variable ***Match*** that takes logical one at matching condition, and ***FPSmax*** which is the instantaneous maximum frames per second.

All calculations are performed on a computer with Intel Core i3, 2.13 GHz processor, and 3MB L3 cache memory.

Time-segments and related variables are show in the composite diagram, Fig. (12).

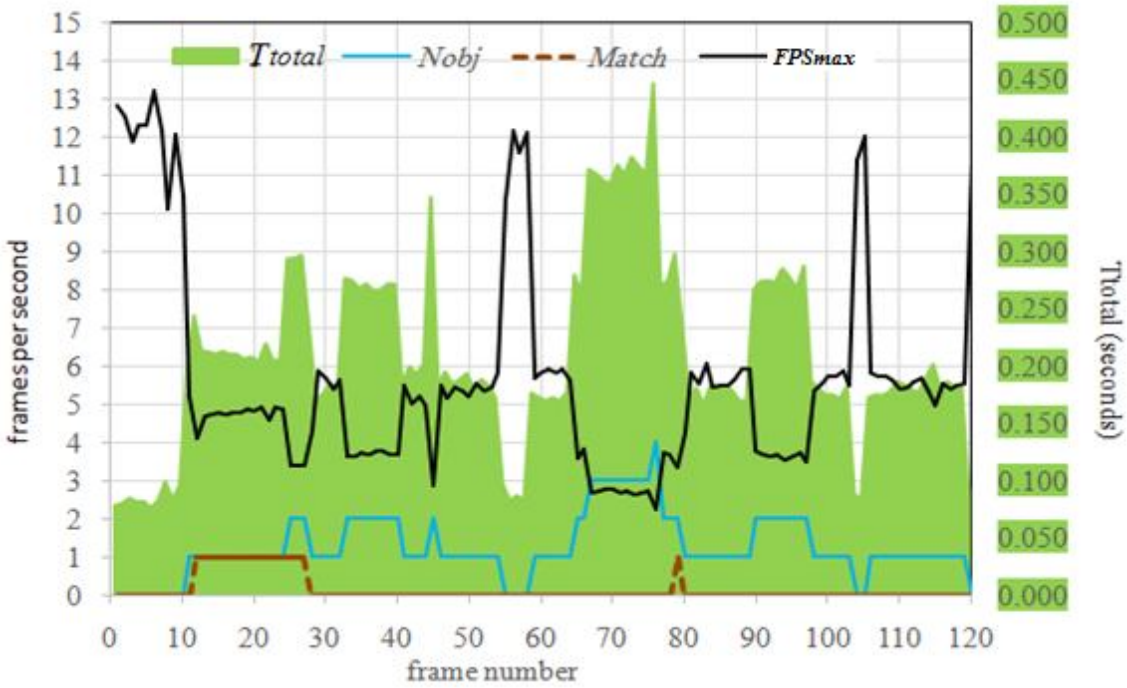


Figure (12): Processing time segments (in seconds) and maximum instantaneous frames per second

8.3 Hardware Implementation

Figure (13) illustrates the addition of hardware component 'hwcosim', which can implement the matching detection operation on hardware in Xilinx FPGA board from Matlab program. The comparator represents one of the important stage in the proposed system.

Figure (14) represents the comparison function block, which compares the current property vector (B) against reference property vector (A). The comparison result is then thresholded using the input threshold (T, corresponding to *error*) deciding match or unmatch condition. The logical matching vector (E) is then returned back to MATLAB to complete the operation of the object recognition.

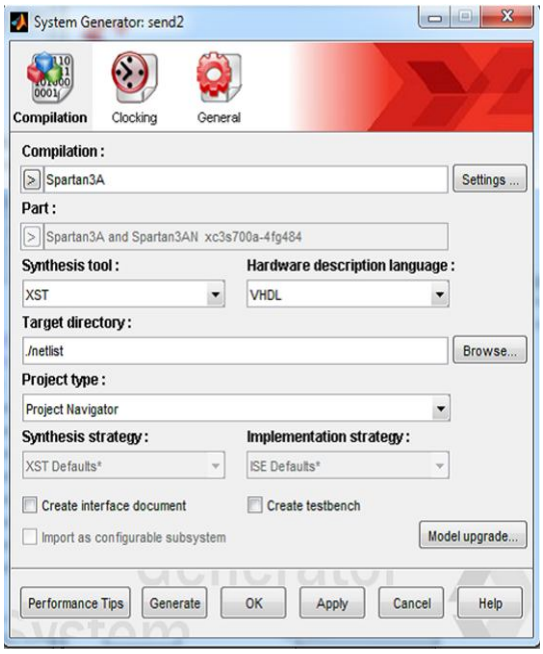


Figure (13): Adding hardware component

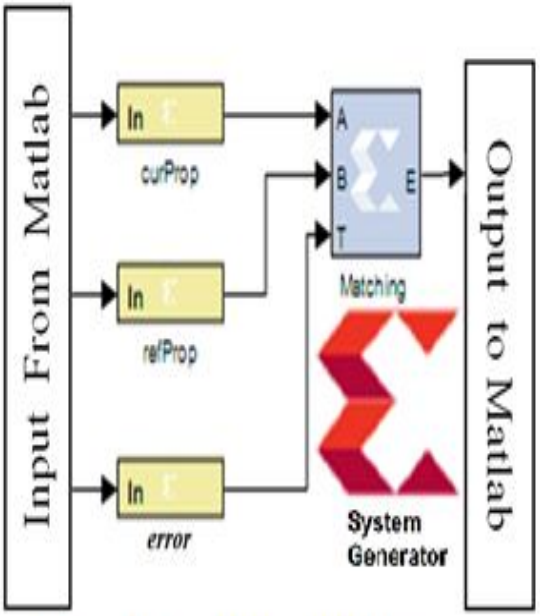


Figure (14): simulation and implimentation

9. Conclusions:

False recognition number has been reasonably low when the accepted property error-ratio, *error*, was below 0.04; hit and miss numbers were optimum when *error* equaled or more than (0.05).

Also, the noise testing showed the degradation of detection process when noise was more than 0.5 percent.

The processing time was proportional to the number of detected objects. There was an important case in frame 76, an object split into two ones. This phenomenon represents a weaknesses of the proposed system. The system considered one object as two ones because a large part of its middle area looked like the background. This case will be treated using more effective segmentation techniques in the future work.

The implementation by utilizing high speed, low cost of Xilinx-XC3S700A-FPGA and the accuracy of MATLAB, system can perform computationally expensive real-time tracking operations.

References

1- Carlos Cuevas, and Narciso García, "Improved background modeling for real-time spatio-temporal non-parametric moving object detection strategies", Elsevier, Image and Vision Computing, 31, page 616-630, 2013.

2- Walter G. Kropatsch and Horst Bischof, "Digital Image Analysis Selected Techniques and Applications", Springer-Verlag New York,2001.

3- Marcos Nieto, Luis Unzueta, Javier Barandiaran, Andoni Cortés, OihanaOtaegui and Pedro Sánchez, "Vehicle tracking and classification in challenging scenarios via slice sampling", EURASIP Journal on Advances in Signal Processing 2011.

4- Pritam Das, RanjitGhoshal, Dipak Kumar Kole, and Rabindranath Ghosh,"Measurement of Displacement and Velocity of a Moving Object from Real Time Video", International Journal of Computer Applications (0975 – 8887) Volume 49– No.13, July 2012.

5- Dashan Gao and Jie Zhou, "Adaptive Background Estimation for Real-time Traffic Monitoring", IEEE Intelligent Transportation Systems Conference Proceedings - Oakland , USA - August 25-29, 2001.
6- Ramin Ramezani, Plamen Angelov, and Xiaowei Zhou, "A Fast Approach to Novelty Detection in Video Streams using Recursive Density Estimation", IEEE 4th International Conference "Intelligent Systems", 2008.
7- Oge Marques, "Practical Image and Video Processing Using Matlab", Published by John Wiley & Sons, Inc., Hoboken, New Jersey, 2011.
8- Plamen Angelov, Ramin Ramezani and Xiaowei Zhou, Student Member, "Autonomous Novelty Detection and Object Tracking in Video Streams using Evolving Clustering and Takagi-Sugeno type Neuro-Fuzzy System", IEEE, International Joint Conference on Neural Networks 2008.
9- G. Prabhakar and B. Ramasubramanian, "An Efficient Approach for Real Time Tracking of Intruder and Abandoned Object in Video Surveillance System", International Journal of Computer Applications (0975 – 8887) Volume 54– No.17, September 2012.
10- K.Shanmugapriya and T.SreeVidhya," An efficient Detection Approch for Visual Survellance System using Morphology", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 2, Issue 12, December 2013.
11- Parameswaran Ramachandran and Diego Sorrentino, "Detection and Tracking of Moving Vehicles in a Parking Lot", ELEC 669 Computer Vision – Project Report, Dept. of ECE/ University of Victoria,Canada, April 2006.
12- Shih-Wei Sun, Yu-Chiang Frank Wang, Fay Huang and Hong-Yuan Mark Liao, " Moving foreground object detection via robust SIFT trajectories", Elsevier, J. Vis. Commun. Image R., 2012.
13- Su Liu, Alexandros Papakonstantinou, Hongjun Wang, Deming Chen, "Real-Time Object Tracking System on FPGAs", 2010.
14- Xiaqiong Yu, Xiangning Chen and Heng Zhang, "Accurate motion detection in dynamic scenes based on ego-motion estimation and optical flow segmentation combined method", IEEE, 2011.
15- Marcos D Zúñiga, François Brémond and Monique Thonnat,"Real-time reliability measure-driven multi-hypothesis tracking using 2D and 3D features", EURASIP Journal on Advances in Signal Processing 2011.
16- Bastian Leibe, Konrad Schindler, NicoCornelis, and Luc Van Gool, " Coupled Object Detection and Tracking from Static Cameras and Moving Vehicles", IEEE Transactions, Pattern Analysis and Machine Intelligence, Volume:30 , Issue: 10, 2008.
17- Ling Cai, Lei He, YirenXu, Yuming Zhao and Xin Yang "Multi-object detection and tracking by stereo vision",Elsevier, Pattern Recognition Vol.43, pag.4028–4041, (2010).
18- Alexander Barth and Uwe Franke, "Tracking Oncoming and Turning Vehicles at Intersections", 2010
19- Hamid Haidarian Shahri, Galileo Namata, Saket Navlakha, Amol Deshpande and Nick, "A Graph-Based Approach to Vehicle Tracking in Traffic Camera Video Streams", 4th International Workshop on Data Management for Sensor Networks (DMSN'07), Austria, 2007.
20- Julius Popoola and Aishy Amer, "Performance Evaluation for Tracking Algorithms Using Object Labels", Natural Science and Engineering Research Council (NSERC) of Canada, 2007.
21- KyungnamKima,Thanarat H. Chalidabhongse, David Harwooda, and Larry Davis, "Real-time foreground–background segmentation using codebook model" Elsevier, 2005.
22- Si Luo and Li Zhang "A Background Model Estimation Algorithm Based
on Analysis of Local Motion for Video Surveillance", IEEE, ICICS 2005.
23- Yi˘githanDedeo˘glu, "Moving Object Detection, Trackingand Classification for Smart Video Surveillance", master of science, 2004.
24- Robert T. Collins and Yanxi Liu, "On-Line Selection of Discriminative Tracking Features", IEEE ICCV03, 2003.
25- V. B. Jagdale, R. J. Vaidya, " High Definition Surveillance System Using Motion Detection Method based on FPGA DE-II 70 Board", International Journal of Engineering and Advanced Technology, ISSN: 2249 – 8958, Vol. 2, Issue-2, Dec. 2012.
26- Ashish Kumar Sahu, AbhaChoubey, " A Motion Detection Algorithm for Tracking of Real Time Video Surveillance", International Journal of Computer Architecture and Mobility, (ISSN 2319-9229) Vol. 1, Issue 6, 2013.
27- Rajvi Shah and P. J. Narayanan, "Interactive Video Manipulation Using Object Trajectories and Scene Backgrounds", IEEE Trans. On Circuits and System for Video Technology, Vol. 23, No. 9, Sept. 2013.

28- Jeisung Lee and Mignon Park, "An Adaptive Background Subtraction Method Based on Kernel Density Estimation", journal of sensors, Vol. 12, 2012.

29- Yasira Beevi C P1 and Dr. S. Natarajan, "An efficient Video Segmentation Algorithm with Real time Adaptive Threshold Technique",International Journal of Signal Processing, Image Processing and Pattern Recognition Vol. 2, No.4, December 2009.

30- M. Vatsa, R. Singh, and A. Noore, "Reducing the False Rejection Rate of Iris Recognition Using Textural and Topological Features", International Journal of Signal Processing Vol. 2, No. 2, 2005.

31- L. Snidaro and G.L. Foresti, "Real-time thresholding with Euler numbers", Elsevier, Pattern Recognition Letters 24, 2003.

32- Vladimir Y. Mariano et. al. "Performance Evaluation of object Detection Algorithm", 16th international conference proceeding on pattern recognition vol. 3, 2002.

33- Alexander R. Marschner, "An FPGA-based Target Acquisition System", Master of Science in Computer Engineering, Blacksburg, December 2007.

34- http://www.mathworks.com/products/matlab/.

35- http://www.mathworks.com/matlabcentral/answers/111010-how-to-link-system-generator-of-xilinx-with-matlab.