Implementation, Comparison and Literature Review of Spatio-temporal and Compressed domains Object detection.

Gokul Krishna Srinivasan [1 2]
[1]Master of Science, Electrical Engineering Department, University of Texas at Arlington
[2]Chief Technology Officer, Third Space Automation Oy

ABSTRACT:

Object detection in a moving video stream is playing a prominent role in every branch of science and research [1]. Objection detection or tracking is done by two different methods, namely, spatio-temporal domain and compressed domain. This project will deal with both the domains in order to bring out the advantages and disadvantages of each and every method in terms of complexity in computations, efficiency etc. Along with that, a detailed literature survey will also be done on the same topic.

Most image and video data are stored or transmitted after compression for efficiency. Processes like pattern detection and localization typically include the extra expense of decompressing the data since most image and video processing techniques require access to the original pixel values in the spatial domain. This project performs a statistical analysis of present object detection schemes in both spatio-temporal and compressed domains. The results of multiple object detection in spatio-temporal domain are compared to those of compressed domain object detection and various evaluation results are analyzed. The comparisons will be made in the context of complexity of algorithm, efficiency and application.

MOTIVATIONS:

Even though there are several research programs and papers that discuss object detection in both spatio-temporal [4 & 5] and compressed domains [1, 2, 3 & 6], there is not much work presented in evaluating the functionality of compressed domain object detection with that of a spatio-temporal detection. This project presents an evaluation of compressed domain object detection and spatio temporal object detection.

1. Literature Review

Spatio-temporal databases deal with applications where data types are characterized by both spatial and temporal semantics. Development and research in this area started decades ago, when management and manipulation of data, relating to both spatial and temporal changes, was recognized as an indispensable assignment. However, spatio-temporal data handling was not a straight forward task due to the complexity of the data structures requiring careful analysis in structuring the dimensions, together with the representation and manipulation of the data involved. Therefore, the earlier work in this area began from separate research in both temporal and spatial databases. This effort later became the basis for spatio-temporal database models. Since the integration of spatial and temporal database models into spatio-temporal database models, a number of new approaches have been proposed. At the same time, reviews of these works have classified and compared the existing spatio-temporal models. Currently,

domain experts are trying to achieve more effective integration of the spatial and temporal aspects providing practical, unified spatio-temporal data modeling, and clarifying the direction for further research and development. Standing at this point the contribution and contemporaneously the aim of this paper is to provide a complete literature review of existing spatio-temporal database models developed or suggested in recent decades and for the first time to critically compare and evaluate them in terms of some universal criteria, in order to identify the trend as well as the needs for further research in the area. The large volumes of visual data necessitate the use of compression techniques. Hence, the visual data in future multimedia databases is expected to be stored in the compressed form. In order to obviate the need to decompress the image data and apply pixel-domain indexing techniques, it is efficient to index the image/video in the compressed form**.** Compressed domain image/video indexing techniques based on compression parameters have been reported in the literature. These techniques have a lower cost for computing and storing the indices. Compressed domain indexing (CDI) techniques can be broadly classified into two categories: transform domain techniques, and spatial domain techniques. The transform domain techniques are generally based on DFT (discrete Fourier transform), KLT (Karhunen-Loeve transform), DCT, and Sub-bands/Wavelets. Spatial domain techniques include vector quantization (VQ) and fractals.

2. COMPRESSED DOMAIN OBJECT DETECTION:

The compressed domain approach exploits the encoded information like motion vectors, discrete cosine transform (DCT) [1] coefficients, and macroblock types which are generated as a compressed bit stream [1, 2, 3 & 6]. Compressed domain object detection can greatly reduce the computational complexity and make real-time or fast processing possible although the precision is not better than the spatial domain approach. The conventional compressed domain object detection algorithms uses motion vectors or DCT coefficients as resources in order to perform object detection and tracking [6]. These encoded data are not enough credible or insufficient to detect and track moving objects, but recent work uses an extra feature called vector-featured images that record moving regions and accumulate unmoving regions in which the moving objects are expected to exist after the current frame [6]. This method uses only motion vector estimate without any other information from the encoded bit stream. In the vector featured images there are five types of block regions [2]. The basic unit of a region in the vector-featured image is the same size as MPEG macroblocks. The five different blocks are reference block B_r, current block B_c, background block B_b, moving block B_m and unmoving block B_u. The algorithm consists of four different steps, namely: initial region extraction, moving region detection, unmoving region creation and updating and modification of vector featured regions [2]. The results obtained are comparable to real time spatial object detection algorithm. Figure 1 shows the initial region extracted from the motion vector values, in fig 2 the label f_{n-2}, f_{n-1}, and f_n represent the corresponding frame and each block in the frame represents a macro block motion vector value. The initial regions are formed by one to one mapping of the previous and the next corresponding frame.

1. INITIAL REGION EXTRACTION:

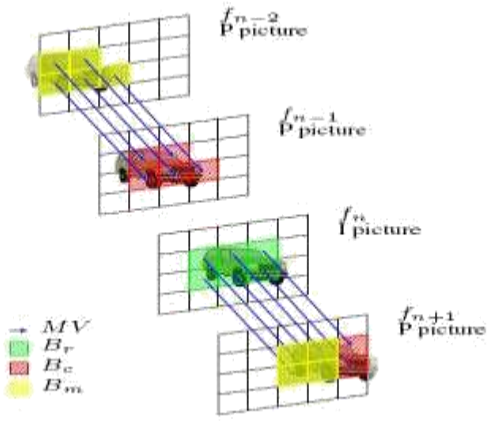

Fig.1: Each block in the figure represents a macro block motion vector value, [2].

2. CREATING AND UPDATING UNMOVING REGION:

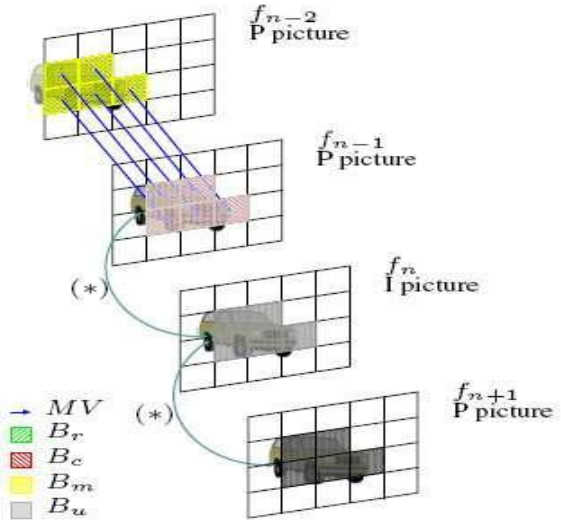

Fig.2: Extraction of moving and unmoving regions, [2].

3. Motion vector calculation:

Using the previous and the future frame as reference, the encoder finds the motion vectors for the forward and backward prediction frame. Each video sequence is divided into one or more group of pictures. The encoder outputs the motion vectors in the bit stream order. Only the motion vectors that are received in the decoder side are processed to find the moving object region.

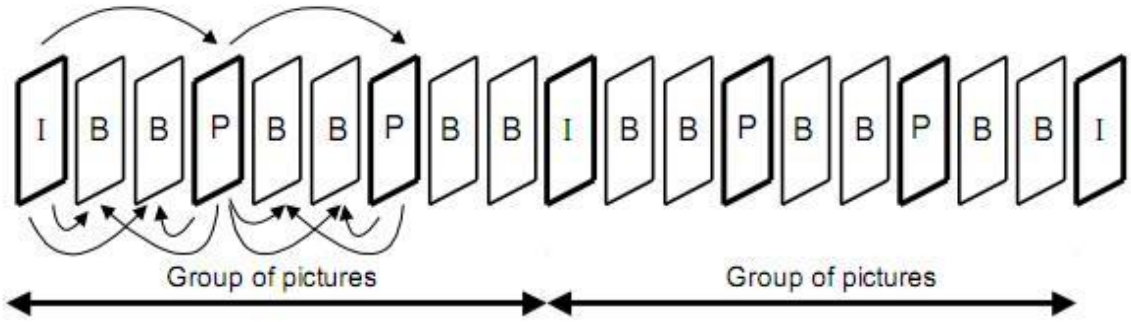

Fig 3: MPEG group of pictures – Display order [11].

Frames do not come into the decoder in the same order as they are displayed. The ouput frames from the encoder will be of the form I B B P B B P B B I B B P B B P B B I. Where I is the intra coded frame, P is the predicted frame and B is the bi-directionally predicted frame. First, reorder the incoming frames or slice from bit stream order to display order. To reorder the frames to the display order the following procedure is followed,

- If an I or P frame comes in put it in a temporary storage called future.
- I or P is left in the future until another I or P frame comes in, on the arrival of a new I or P frame, already present I or P frame is taken out from the temporary storage called future and is put in the display order and the newly arrived I or P is put into the temporary storage called future.
- All B frames are immediately put in the display order.
- At the end whatever frame is left in the temporary variable called future is put in the display order.

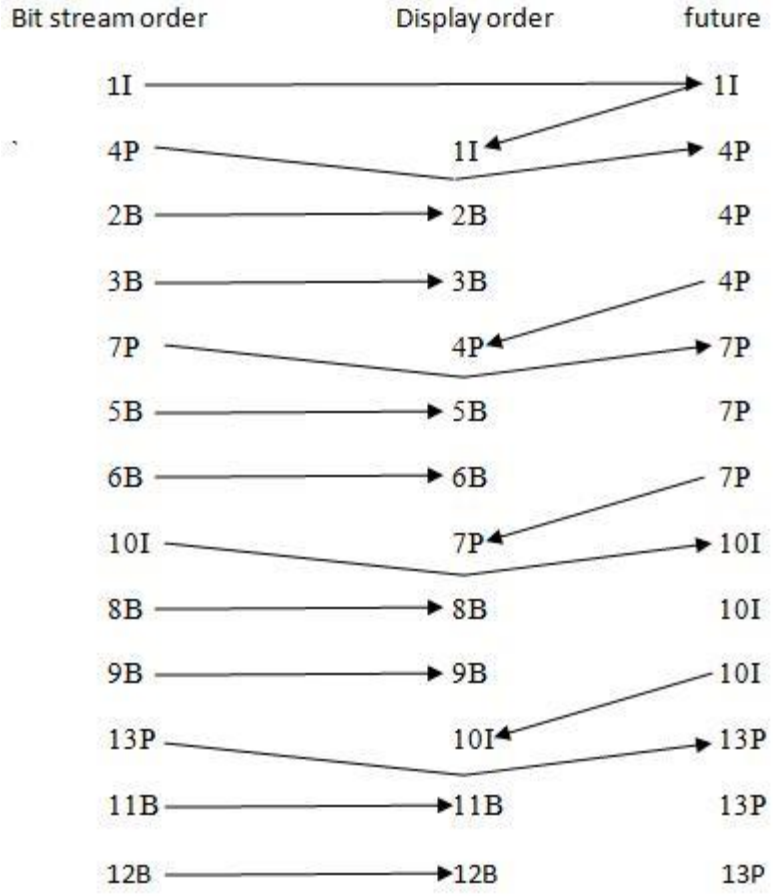

Fig 4: Conversion from bit stream order to display order [6].

4. Algorithm for storing the motion vectors in respective arrays

For ease of handling, the motion vectors are stored in a two dimensional arrays and the size of the array corresponds to the frame size in macro block (in this case 8x8 macro block was chosen). The forward prediction vectors and backward prediction vectors are stored in separate arrays. Each prediction vector in turn contains two more arrays to store the horizontal and vertical movement of vectors. To find the motion from one frame to another, a record of motion vectors of the previous frame has to be kept. Fig 5 shows the explanation of the algorithm in a flow chart. The process of inputting the motion vectors into correct arrays and reordering the frames into the display order were incorporated in the decoder.

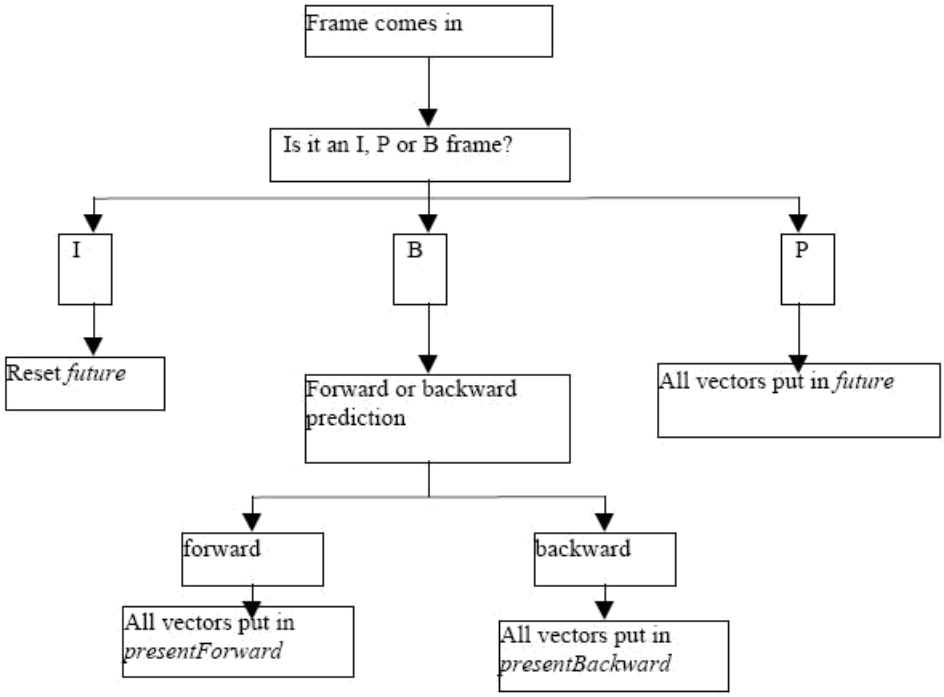

Fig 5: Flow chart explaining storage of motion vectors in respective arrays. [6].

The final output of this algorithm stores the motion vectors of successive frames in an array. For finding the motion from frame to frame, the present and previous frame motion vectors are subtracted. If an 'I' frame is encountered all the values in the array are set to zero. If a B frame is encountered, forward prediction and backward prediction vectors are subtracted separately and an average total motion is found in both the horizontal and vertical directions. The obtained motion vectors for each frame are written into a separate file each for horizontal and vertical motions.

5. Moving Region Detection:

A moving object generates non-zero motion vectors that appear continuously over multiple frames. The continuous appearances of motion vectors cause previous block and current block motion vectors to overlap, these regions are detected as moving regions Bm. When a current frame pointed to is an I-frame, then no motion vectors information will be there In the current frame and hence the contents of motion vector information from previous vector-featured image is copied to present frame, moving block is not created in these frames.

6. Unmoving region creation and updating:

The difficulty in detecting moving objects by using only motion vector information is that the probability of getting false positive (i.e) detecting a background as a moving object is high. To overcome this, methods such as connected component analysis and threshold value for motion vectors are implemented. A block which was having a motion vector in the previous frame may tend to a zero motion vector in the current frame. These regions are marked as unmoving regions as the moving object is expected to exist after current frame. When a moving object stops, zero motion vectors are generated, however just before the object stops, moving block regions will be created along the movement of the object whose current block has a zero motion vector. These regions whose previous motion vector value had a moving block related to the current zero motion vector is marked as unmoving block. The effect of the unmoving block is also crucial, there are two criterions that should be considered, one is that the unmoving regions should be considered correctly and a object which has moved in position after few frames should also be noted. For these issue unmoving block regions are monitored for some k frames, if a zero motion vector persists its brightness value is decreased gradually, if not it is marked as a moving block region.

7. GUI for annotating object locations:

The obtained results of moving object detection from both the methods are compared with the manually annotated hand location gives the coordinate location of the centre of the object in every frame. The user selects a video filename from the list of filenames present in the drop down box, the second edit box holds the start frame number and the third edit box holds the end frame number as entered by the user. On clicking the get frame button, a separate window opens which contains the image of the particular frame number that lies between the start and end frame. Using a marker tool the user annotates the hand location by which the coordinate locations are obtained. The hand locations obtained between the start and the end frame are stored in an array which forms the bench mark for evaluating the accuracy of moving object detection. Fig 6 shows the GUI that has been created for manually annotating hand locations.

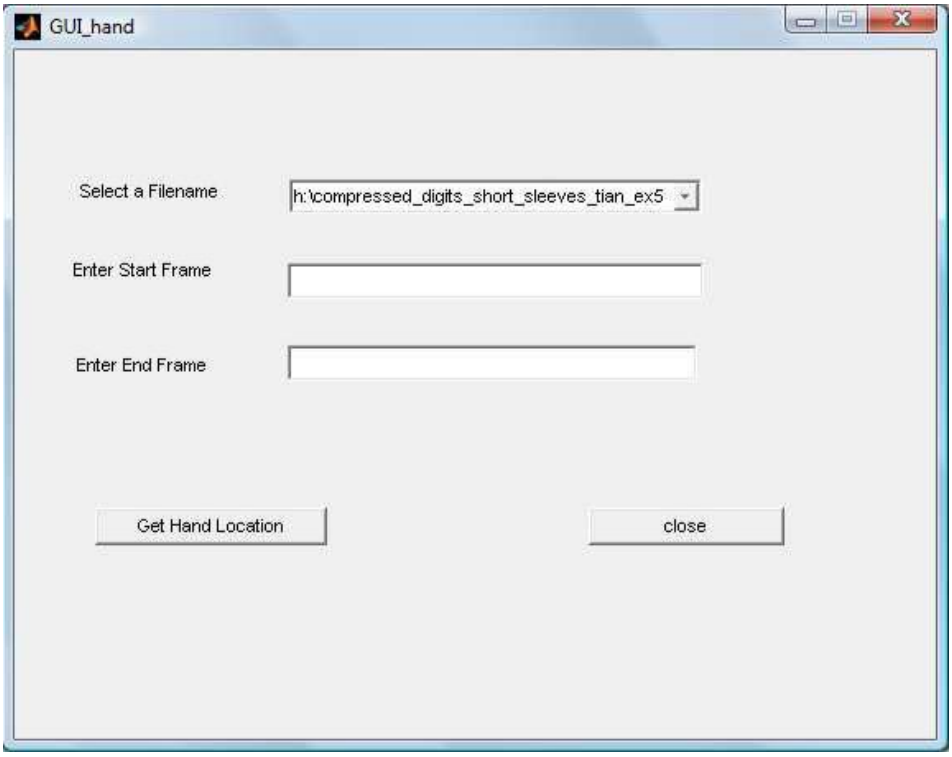

Fig 6: GUI for manually annotating hand locations

3. Spatio-Temporal Object Detection:

Spatio-temporal object detection is an integral part of many computer vision applications. A common approach is to perform background subtraction, which identifies moving objects from the portion of a video frame that differs significantly from the background. There are four main steps in a background subtraction algorithm [5], they are preprocessing, background modeling, foreground detection and data validation as shown in fig 7. Preprocessing step consists of a collection of simple image processing tasks that change the raw input video into a format that can be processed by subsequent steps. Steps like intensity adjustments, smoothening are handled in preprocessing stage. In real-time systems frame size reduction is also done to speed up the process. The block diagram of a background detection algorithm in spatial domain is shown in fig 7. Background modeling is at the heart of any background subtraction algorithm, background modeling is a process to obtain static image regions from a sequence of video. Background modeling is broadly classified into recursive and non-recursive techniques.

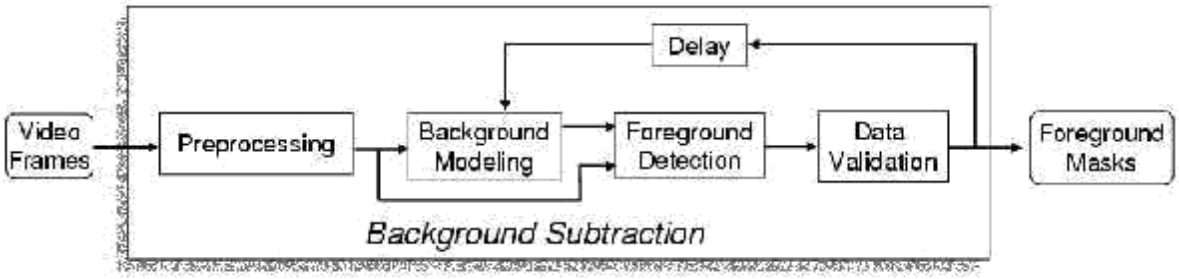

Fig 7: Block diagram of a spatio-temporal object detection [5].

Non-recursive technique uses a sliding window method and it stores a buffer of the previous L video frames, and estimates the background image based on the temporal variation of each pixel within the buffer. Recursive technique is a non-adaptive technique and does not use past input frame information. Foreground detection just compares the input frame with the background model and uses a threshold value as in binary classification. Data validation is the process of improving the candidate foreground mask based on information obtained from outside the background model. All the background models have three main limitations, first, they ignore any correlation between neighboring pixels; second, the rate of adaption may not match the moving speed of the foreground objects; and third, non-stationary pixels from moving leaves or shadow cast are easily mistaken as true foreground objects. These limitations of background model are reduced in foreground detection and the possibility of false detection is also reduced [5].

1. Parametric and non-parametric object detection:

Background modeling stage is at the heart of any background subtraction algorithm, having said that the recursive technique based background modeling uses a parametric and non-parametric model for object detection. These two models can be intuitively understood from their names which mean a parametric model requires some parametric estimates in order to perform object detection whereas a non-parametric model can perform object detection without requiring any parameters to estimate them. An example of parametric object detection is a simple Gaussian model which requires the mean and standard distribution estimate of the object being detected. Consider that a particular color object from an image has to be estimated. First the sub window that contains samples of particular color that has to be detected. Then find the mean and standard deviation from the sub window pixel data.

Steps to be performed for the parametric model of object detection,

(1) Estimate the mean and standard deviation of a particular color object to be detected from the sub window block which contains only particular color pixel values.
(2) Find, P_1(RGB/color), which forms the training set,
(3) Assume that colors are mutually independent, then,
P_1 (RGB/color)=P_1 (R/color)*P_1 (G/color)*P_1 (B/color)
(4) Each P_1 (R/skin), P_1 (G/skin) and P_1 (B/skin) are estimated through Gaussian probabilities.
(5) Gaussian PDF is given as,

$$F(x) = \frac{\exp[\frac{-(x-m)^2}{2\sigma_2}]}{\sigma\sqrt{2\pi}}$$

where,
F(x) → Functional Gaussian probability estimate of data x.
x → The sample data whose Gaussian probability has to be found.
m → The mean value of the object to be detected.
Sigma → Standard deviation.

Maximum likelihood estimate is then applied to the obtained Gaussian probability model to obtain the color object region from the video frame [5 & 7].

A non-parametric model of object detection uses minimum or no parametric estimate to perform object detection. An example of non-parametric model is histogram based object detection. For example, to detect green object, the each color image pixel is made up of overlapping red, green and blue intensity values, cancel out the green pixel intensity with that of the red pixel and also with the blue pixel intensity values, the formulation becomes [7],

$F(i,j)=2*I_g(i,j,2)-I_r(i,j,1)-I_b(i,j,3)$, where $1\leq i\leq N$, $1\leq j\leq M$

- N and M are row and column lengths of the image I(N,M).
- $I_g(i,j,2)$ is the index of the green pixel intensity and similarly for
- $I_b(I,j,3)$ is the index of the blue pixel intensity
- $I_r(I,j,1)$ is the index of the red pixel intensity
- F(i,j) is the final functional value of the green color density distribution.

After the estimate of background modeling has been performed a correlation filter is used to remove salt pepper regions and to obtain smooth surfaced object detection from the entire background image. The steps illustrated above detects object in a single frame, In order to perform moving object detection frame differencing has to performed and the obtained results has to be multiplied with the single frame object detection. Frame differencing is performed as follows,

Let frame(n) be current frame, frame(n-1) be previous frame and frame(n+1) be next frame, then

Frame_diff=Min((frame(n-1)-frame(n)),(frame(n+1)-frame(n))).

4. Experimental results - spatio-temporal moving object detection:

The spatio-temporal object detection algorithm along with its performance metrics was implemented in MATLAB. Two video sequences with different proximity of distance from camera were considered during the testing and training phase. For each video both single and multiple detection boxes performance was evaluated. The performance metrics such as localized output box count precision, average detected box area precision, area based recall for frame, area based precision for frame were evaluated and the tabulations and graphs are plotted. The training for detecting object in a video was performed by annotating or manually marking hand the locations using a GUI. The output of the training set contains the spatial co-ordinate locations of the center of detected object. The testing was performed using the color detection algorithm which also returns the spatial coordinate locations of detected object in each frame. Let G be the correctly classified object in a single frame found during testing and let D be the set of output boxes produced by the algorithm, N_g and N_d be their respective values in each frame. Testing was done on two video sequence close_detect and detect.

1. Localized output box count precision:

This metric count the number of output boxes that significantly covered the correctly classified object obtained from training. An output box D in each frame significantly covers the ground truth if a minimum proportion of its area overlaps with detection box G. Ground truth can be defined as the area under the actual moving object in each frame or can also be defined as a correct classification in each frame.

Loc_box_count = ∑ ( ) - ( )

box_prec(D) = { 1 | ∩ | >

Where,

Ug → is the union space of total number of pixels in the detection box.

|D ∩ | → is the pixel count of output detection box overlapping with the ground truth.

Overlap min is the minimum proportion of the output box's area that should be overlapped with the ground truth in order to say that output is precise.

2. **Average detected box area precision:**

This metric provides the average of detection boxes precision in covering the area that ground truth object covers in each frame.

Precision = ∑
Box precision (D) = |D ∩ |/|D| ( )/

Where,D= output detection boxes in each frame; Ug=Spatial union of ground truth detection

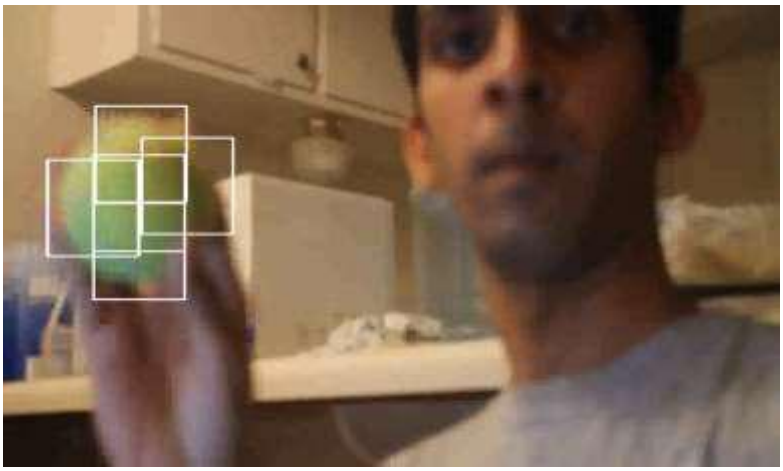

Figure 8: Hand movement detected for frame #100, 5 detection boxes of size 150X150

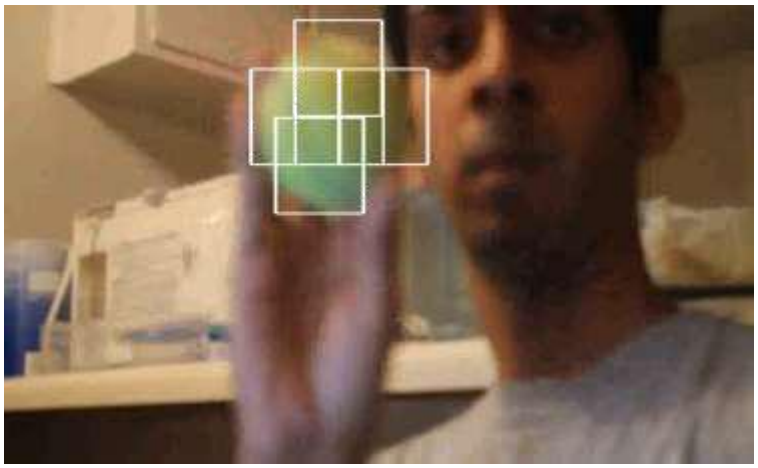

Figure 9: Hand movement detected for frame #115, 5 detection boxes of size 150X150

4. Experimental results - moving object detection using motion vectors:

The motion vector values that are obtained were result of an 8x8 macro block encoding, the frame size used in this project is 240x320, and hence the motion vector values obtained for each frame were of size 30x40. In order to represent the moving object detection using motion vectors, the direction vector plot of each frame is plotted as shown in fig 10. The constraints that to be considered in compressed domain object detection using motion vector estimate is that, if the test video has moving background objects other than object to be detected then the accuracy of detection will be less.

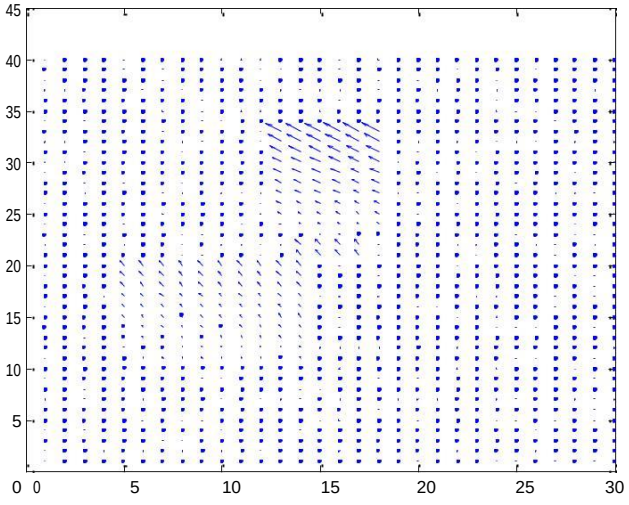

Figure 10: Motion vector values from frame #42 of close detect 1.

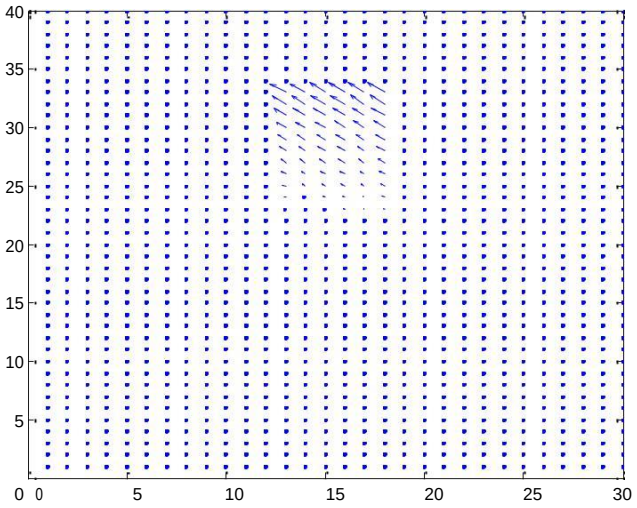

Figure 11: Threshold motion vector values for frame #115

Using connected component analysis and setting up a threshold value, the regions of maximum displacement is obtained, in this frame the maximum motion vector after setting the threshold is found to be at array index [33,15] which is [264, 120] pixel location in the original spatial frame coordinate location. The threshold value must be selected such a way that the important information from the motion vector should not be erased.

5. Discussions:

Having obtained the moving object detection from both the methods, the efficiency of detection of the moving objects has to be compared. The results obtained suggest that the accuracy of detection of moving objects is more efficient using the spatio-temporal object detection algorithm than using a compressed domain motion vector based moving object detection. But it should be noted that as the size of the moving object is larger, the performance of compressed domain motion vector based detection was better. The accuracy of detection which specifies the percentage of correct detection is tabulated in table 1. The results were obtained by considering the object detection from three frames of two different test video sequences.

Test video sequence	Accuracy of detection - spatio-temporal object detection	Accuracy of detection – motion vector based compressed domain object detection
Close detect	94%	79%
Distant detect	91%	68%

Table 1: Percentage of correct detection

Accuracy of detection= correctly classified object/(correct answers), as shown in fig 12.

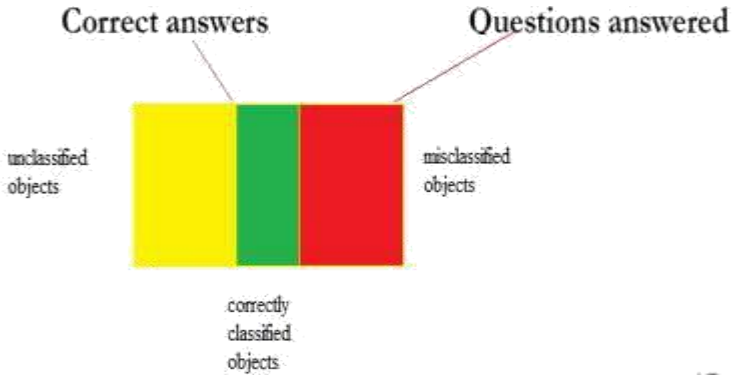

Fig 12: Accuracy of detection

6. Conclusions:

The performance of moving object detection is evaluated using two different techniques. The experimental results show that the accuracy of moving object detection in spatio-temporal domain is better than that of the compressed domain motion vector based moving object detection.

7. References:

1. Qiya Z and Zhicheng L, "Moving object detection algorithm for H.264/AVC compressed video stream", ISECS International Colloquium on Computing Communication, control and management, vol. 1, pp. 186-189, Sep, 2009.
2. Yokoyama T, Iwasaki T, and Watanabe T," Motion vector based moving object detection and tracking in the MPEG Compressed Domain", Seventh International Workshop on content based Multimedia Indexing, pp. 201-206, Digital Object Identifier: 10.1109/CBMI.2009.33, Aug, 2009
3. Kapotas K and Skodras A. N," Moving object detection in the H.264 compressed domain", International Conference on Imaging systems and techniques, pp. 325-328, , Digital Object Identifier: 10.1109/IST.2010.5548496, Aug, 2010
4. Sen-Ching S. C and Kamath C," Robust techniques for background subtraction in urban traffic video" Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, vol. 1, pp. 586-589, Jul 2004.
5. Elhabian S. Y, El-Sayed K. M," Moving object detection in spatial domain using background removal techniques- state of the art", Recent patents on computer science, Vol. 1, pp. 32-54, Apr, 2008.
6. Sukmarg O and Rao K. R," Fast object detection and segmentation in MPEG compressed domain", TENCON 2000, proceedings, vol. 3, pp. 364-368, Mar, 2000.
7. Thompson W. B and Ting-Chuen P," Detecting moving objects", International journal of computer vision, vol. 6, pp. 39-57, Jun, 1990.
8. JM software - http://iphome.hhi.de/suehring/tml/
9. Mariano V. Y., et al," Performance evaluation of object detection algorithms" International conference on pattern recognition, Vol.3, pp. 965 – 969, June 2002.