Design and implementation of novel image segmentation and BLOB detection algorithm for real-time video surveillance using DaVinci processor

Badri Narayana Patro
Dept. of Electrical Engineering
Indian Institute of Technology Bombay, India
Email: badriindia.1@gmail.com

***Abstract*—A video surveillance system is primarily designed to track key objects, or people exhibiting suspicious behavior, as they move from one position to another and record it for possible future use. The critical parts of an object tracking algorithm are object segmentation, image clusters detection, and identification and tracking of these image clusters. The major roadblocks of the tracking algorithm arise due to abrupt object shape, ambiguity in number and size of objects, background and illumination changes, noise in images, contour sliding, occlusions and real-time processing.**

This paper will explain a solution of the object tracking problem, in 3 stages: In the first stage, design a novel object segmentation and background subtraction algorithm, These algorithm will take care of salt pepper noise, and changes in scene illumination. In the second stage, solve the abrupt object shape problems, objects size and count various objects present , using image clusters detected and identified by the BLOBs (Binary Large OBjects) in the image frame. In the third stage, design a centroid based tracking method, to improve robustness w.r.t occlusion and contour sliding.

A variety of optimizations, both at algorithm level and code level, are applied to the video surveillance algorithm. At code level optimization mechanisms significantly reduce memory access, memory occupancy and improved operation execution speed. Object tracking happens in real-time consuming 30 frames per second(fps) and is robust to occlusion, contour sliding, background and illumination changes. Execution time for different blocks of this object tracking algorithm were estimated and the accuracy of the detection was verified using the debugger and the profiler, which will provided by the TI(Texas Instrument) Code Composer Studio (CCS). We demonstrate that this algorithm, with code and algorithm level optimization on TIs DaVinci multimedia processor (TMS320DM6437), provides at least two times speedup and is able to track a moving object in real-time as compared to without optimization.

***Keywords*—Centroid Based, Segmentation, BLOB, Tracking, Optimization, Background Subtraction DaVinci Processor.**

I. Introduction

Surveillance systems are used for monitoring, tracking of objects and screening of activities in public places such as banks, in order to ensure security. Various aspects like screening objects and people, biometric identification and video surveillance, maintaining the database of threats and blackmails etc., which are used for monitoring the activity.

In this paper, we present our approach in each stage, such as video object tracking approach based on background subtraction, image segmentation [1], blob detection and identification, and center of mass based tracking, where these techniques were implemented on TMS320DM6437. Our approach consists of improvements and novel ideas in each stage of the object tracking algorithm, such as, threshold based segmentation, centroid based tracking and a novel idea for blob detection and identification. These individual improvements combine together to improve video object tracking, and provide fast, accurate and good video object tracking services. Apart from presenting new and improved algorithms in the different stages of the object tracking algorithm, also present optimizations at the source code level. These consist of using the inbuilt DSP instructions, the DSP pipeline technique, and Temporary Variables, to achieve speed optimization and code optimization (based on the characteristics of C64x+ DSP core). Consequently, this algorithm can be applied to multiple moving and still objects in the case of a fixed camera. Different steps for video object tracking are shown in Figure 1.

Surveillance camera is used to capture input video data. This input video frame is used to create a YUV-stream, after correcting for bad pixels, color, lens distortion etc., from the RGB stream. This input stream is then down sampled and followed by resizing to 480x720, and low light is adjusted using adjust auto focus, white balance and exposer. This front end processing is collectively referred to as the Image Pipe.

The most computation intensive part of tracking algorithm is background subtraction, which is based on difference between a previous image or background image[3] and the current image. There are various methods for background subtraction[11], and each algorithm has its own merits and demerits. However, algorithms based on the difference of images have problems in the following cases: First case, when still objects included in the tracking task exist. Second, when more than one moving objects are present in the same frame. Third, when the camera is moving. Fourth, when occlusion of objects occurs. By considering real-time performance and computational speed optimized method for background subtraction is implemented in this paper which tries to avoid most of the effects. The occlusion effect can be solved by using adjacency pixel value and center of mass algorithm for object tracking.

Image segmentation process is identifying components of homogeneous regions in the image. This algorithm is used to extract various information of a particular object like persons, car. Primer there are three image Segmentation algorithm[10],

978-1-4799-3080-7/14/$31.00

such as 1. threshold based technique, 2. boundary based technique, 3. region based technique.. The basic threshold technique is the process of reducing the gray levels in the image and is based on pixel intensity level. problem in threshold [10] technique requires additional filtering background noise and clustering. It is very difficult to distinguish intensity gradient, i.e., variance in contrast in case of edge based technique. problem in region growing is to to find regions in case of over stringent[10], blurred regions and merging of the regions. A novel approach of image segmentation algorithm, in order to extract all objects information in the input image, is implemented in this paper which is optimized and overcome most of the problems.

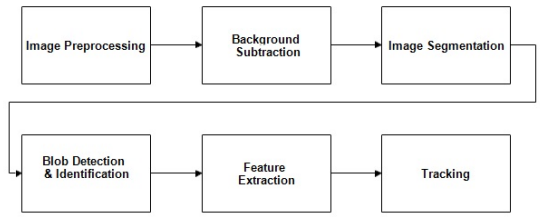


Fig. 1: Object tracking for visual surveillance system

The main problem of object tracking in continuous video stream is Binary Large OBjects (BLOBs) detection and identification[4]. Various approaches developed for blob detection can be grouped into following categories: scale-space analysis, matched filters or template matching, watershed detection and sub-pixel precise blob detection which will have their advantage and disadvantages. In this paper we will discusses about various method for blob detection and implement best method for BLOB Detection.

In case of BLOB identification process those pixel value which is greater than threshold value are checked for their adjacency pixels[4].Each blob is enclosed by rectangle which is elastic in nature, i.e., it can stretch in vertical, and horizontal directions until the whole blob is enclosed in a rectangle box. The process is then repeated for all blobs that are present in the image. This paper is based on a BLOB detection system with adjacency pixels, a Bounding Box and a Center-of-Mass based approach for tracking the computation of the center points using DSP processor. Also in this paper will discuss about design problems and performance gain. The statistical feature[1] of each blob such as, the approximate location of the center gravity, the size of the rectangular enclosure, the actual size or pixel count of the blob, and volume bounded by the membership function value are calculated.

The major problem for detection and tracking applications is occlusion. In Kalman[8] filter, observed that tracking of moving objects are partially occluded. Therefore the objects need to be detected without occlusion or the model of the occluding objects had to be taken into account as well. The use of centroids introduces a competitive learning mechanisms in the tracking algorithm which lead to improved robustness with respect to occlusion and contour sliding[6].

Boundary Box and Center of mass are two approaches for estimation of center points of the enclosed BLOB's. The problem in the boundary box[4] method is that center position is strongly depends upon pixel presents at the blobs border and Flicking at boundary due to detection of pixels based on Threshold value. In order to achieve a higher precision for the center point computation the application of a Center-of-Mass (CoM) based method has been selected. For the CoM based method the BLOB detection module needs to store more information about the identified pixels. It covers the design and implementation of a tracking solution for the estimated BLOB center points in hardware and in software. Those two approaches are compared with respect to precision[6] and performance.

Video surveillance systems require high resolution video, large bandwidth and higher computational speed at low cost and power. DaVinci devices are suitable for surveillance applications as they provide, ASIC-like low cost and power for the complex processing, programmable DSPs with high performance. The DaVinci processor DM6437 is a fixed-point DSP based on the third generation high performance, advanced VelociTI[12] very-long-instruction-word (VLIW) structure of clock rate 700 MHz and faces digital multimedia applications. It also provide function accelerators in the video processing[10] subsystems(VPSS) for common video processing tasks such as Image pipeline, encoding, decoding and display.

II. Design Statement

From the Literature Review we can derive our problem statement which can consider following points

- To have a better a background subtraction model for object tracking.
- To develop a better object segmentation algorithm with a good image filtering and edge detection which will improve computation time and complexity for real time applications.
- Design an better image clusters detection and identification program which is not discussed in the most of literature.
- Design simple, accurate and optimized tracking method to improved robustness w.r.t occlusion and contour sliding.
- Optimization of all algorithms for program flow designed, memories access time and memory uses.

III. Algorithm Design For Object Tracking

It proposes a novel algorithm, which consist of five stages: In the first stage, input captured stream has to be pre-processed for down-sampling, resizing and to creates a YUV-stream after correcting for bad pixels, color, lens distortion. Second stage, segregate the moving objects from the background form each frame using back ground subtraction method all objects present in the image can be detected irrespective of they are moving or not. Third stage, with the help of image segmentation using crisp fuzzier, smooth filter and median filter, the subtracted image is filtered out and free from salt paper noise. In the fourth stage, the segmented image is processed for detecting and identifying the BLOBs present, which is going to be tracked. In the fifth stage, the object tracking is carried out

by feature extraction and center of mass calculation in feature space of the BLOB detection results of successive frames. The major steps for object tracking are as shown in Figure 2.

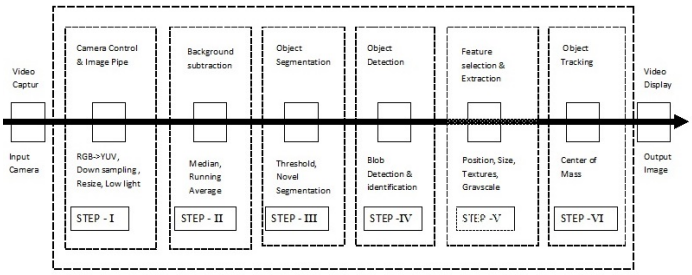


Fig. 2: TRACKING BLOCK DIAGRAM

The major steps for object tracking are such as:A. Image Preprocessing, B. Background subtraction, C. Object segmentation, D. Blob Detection and Identification, E. Feature Extraction and Selection, F. Object tracking.

A. Camera control and Image Pipe

The goal of pre-processing is to process the captured input stream and apply basic image processing logic to change row input video into a specific format by removing noise using smoothing and unwanted object by using morphological operation. Here captured frame is processed using down sampling, resizing, preview engine and H3A in order to creates a YUV-stream. The role of Image Preview[5] is for transforming row uncompressed image and video data from camera into YUV422 for compression or display. Capture Image stream is converted from RGB to YUV after correcting for bad pixels, color,dark frame, lens distortion, luminance enhancement and chrominance suppression. Job of resizer is to make up sample for digital zoom using bi-cubic Interpolation and down sampling to reduce image size. Auto focus (AF), auto white balance(AWB), and auto exposure (AE), collectively known as H3A is applied on the raw image data.

B. Background subtraction

The main motivation for the background subtraction is to detect all the foreground objects in a frame sequence from static camera. The rationale in this approach is that of detecting the moving objects from the difference between the current frame and a reference frame, often called the "background image", or "background model". Four major steps in background subtraction method are

- Pre-processing : Noise filtering
- Background Modeling: Mean filter or Median filter or frame difference.
- Foreground detection: Detection Based on 2 frames or 3 frames or Gaussian model.
- Data validation

In order to detect the foreground objects, the difference between the current frame and an image of the scene's static background is compared with a threshold. The background subtraction equation is expressed as:

$$|Frame(t) - Background(t)| > Th \tag{1}$$

$$Background(x, y, t) = Image(x, y, t-1) \tag{2a}$$

$$|Image(x, y, t) - Image(x, y, t-1)| > Th \tag{2b}$$

There are some problems associated with this method[11] are occlusion handling problem i.e. overlapping of moving blobs, camera oscillations, lighting condition, shadow detection and illumination changes. There are various other simple approaches aiming to, maximize speed, limit the memory requirements, to achieve the highest possible accuracy under any possible circumstances. These approaches include, running gaussian average, temporal median filter, mixture of gaussians, kernel density estimation (KDE), co-occurrence of image variations, eigen backgrounds. By considering real-time performance computational speed background subtraction algorithm is implemented in this paper using running average difference method as the background model and foreground detection using 2 frame detection. the back ground modeling equation as follows:

$$B(i+1) = \alpha * F(i) + (1-\alpha) * B(i) \tag{3a}$$

$$|I(x, y, t) - B(x, y, t)| > Th \tag{3b}$$

where α, the learning rate, is typically 0.05.

C. Object segmentation

In image segmentation process,an image is segmented into various regions that have same characteristics and then extracting the interested regions. The basic aspects in image segmentation include threshold, clustering, edge detection and region growing. This paper describe about a novel image segment algorithm as follows:

1) Novel image segmentation approach:: The novel image segmentation algorithm proposed uses crisp fuzzier, smooth filter and median filter in the order as shown in figure below.

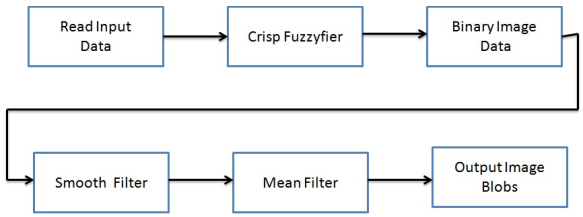


Fig. 3: Image segmentation Steps

In this approach, crisp fuzzier is used for finding out the relevant gray value information and the output data of crisp fuzzier is then processed in order to eliminate isolated points and noise.Enhancing the segmented result using morphological operations. Elimination of isolated points and noise removal is done by using a binary smoothing filter and a median filter respectively.

Steps for Image Segmentation approach are as follows:

1) The pixel data from the input frame and is passed through a crisp fuzzyfier. The job of crisp fuzzyfier is to first select a pixel value , say p, if the selected pixel value p is in the range of PL and PH (PL $\leq$ p $\leq$ PH), then the pixel data is assigned a value of 1.0. Else the pixel data is assign to zero. where PL and PH are the lower and upper limits of the pixel values. This process will replete until all data are processed.
2) The image pixel data obtained in step-1 is processed for smoothing filter which is used to a. remove isolated points, b. fill in small holes, c. fill in small notches in edge segments.
3) The binary image data obtained in step-2 is filtered using a median filter. The median filter is a nonlinear digital filtering technique, in which the center pixel value is replaced with the median of all the pixel values in the filter window. However, the performance of Gaussian is much better than median blur for high levels of noise, but it is used to remove salt and pepper noise (impulsive noise) in the blobs and separate out those points whose intensity is very different form their neighbors and make a blob of all pixel value whose intensity is nearly equal to its neighbors. Finally the output of the median filter consists of clusters or blobs.

D. Object Detection and Identification

The aim of the BLOB detection is to determine the center point of the Blobs in the current frame encoded in XY-coordinates. A BLOB consists of white pixels while the background pixels are black. To simplify the problem, an upper bound for the number of Blobs to detect has been defined. There are Two simple criteria to decide if a pixel belongs to a BLOB:1) Is brightness of the pixel greater than threshold? 2) Is pixel adjacent to pixels of a detected BLOB? .Dependent on the viewing angle of the user and the speed of his motion,the intensity and shape of the Blobs can vary. if the user speed of motion is less than the actual frame rate and resolution of the camera which will eliminate blurring effect.

In case of BLOB identification process those pixel value which is greater than threshold value are checked for their adjacency pixels. To combine detected pixels to Blobs a test of pixel adjacency needs to be performed. There are two common methods to evaluate adjacent Pixels that is 4-pixel and 8-pixel neighborhood as shown in fig.4 . 4-pixel neighborhood checks for adjacent pixel only on vertical and horizontal axis of the the current pixel. 8-pixel neighborhood check for adjacent pixel on vertical, horizontal and diagonal axis of the the current pixel. In this paper,to realize BLOB detection 8 pixel neighborhood is used for adjacency check due to 8-pixel neighborhood is more reliable than 4-pixel neighborhood.

The BLOBs with blur depends on the direction of the objects movement. If the motion of the object goes along the horizontal or vertical axis of the screen, the BLOB can show an elliptical shape. In the detection approach, only the elliptical shape has been taken into account.

E. Feature Extraction of Blob

In this step geometric and radiometric feature of the blob are calculated. Boundary length, Blob area, color of each blob, shape, Distance between the blob, Geometric moments that is co-ordinate of center of blob, and higher order moments are the following significant features of blobs which are going to extract.

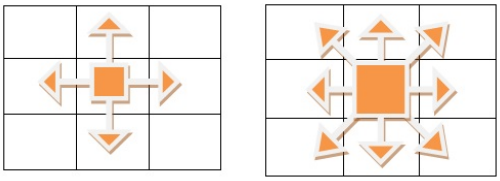


Fig. 4: 4 and 8 Connected Neighborhoods

F. Object Tracking: Center-Of-Mass method

After Blobs detection and feature selection method, the next step is computation of their center points of Blobs. Boundary Box and Center of mass are two approaches for estimation of center points of the enclosed BLOBs. In Boundary Box approaches, the center-point of the blob can be estimated with the help of minimum and maximum coordinates of the pixels on the horizontal and vertical axis.

The center_positions of the BLOBs can be computed using following formula :

$$X\_position = \frac{maxX\_position + minX\_position}{2} \quad (4a)$$

$$Y\_position = \frac{maxY\_position + minY\_position}{2} \quad (4b)$$

The computational complexity of mathematical operations like addition and subtraction are not so high. Problem is division operation, that is Divide by 2 can be achieve by using bit shift operator. The problems in boundary box algorithms are follows: 1. Center position is strongly depends upon pixel presents at the blob's border. 2. Flicking at the boundary due to detection of pixels based on Threshold value. When Blobs in motion, this Flickering effects become stronger and stronger and cause motion Blur in the blobs shape. In order to reduce flickering effect **center-of-mass method(CoM** is used for the computation of the blob center point.

In this method, for the computation of the center point All pixels of the detected blob are taken into account. The formula for finding position value of the detected pixels as weighted sum and calculates an averaged center coordinate. BLOBs center_position is follows

$$X\_position = \frac{\sum(X\_position\_of\_all\_BLOB\_pixels)}{number\_of\_all\_BLOB\_pixels} \quad (5a)$$

$$Y\_position = \frac{\sum(Y\_position\_of\_all\_BLOB\_pixels)}{number\_of\_all\_BLOB\_pixels} \quad (5b)$$

Above algorithm helps to find the center position of a BLOB w. r. t. its mass. But it does not use the information

about the brightness of the pixel, which belongs to the BLOB. In order to increases the precision of the center-position of a blob, CoM algorithm is modified by replacing pixel position with the product of the pixel position and brightness of the pixels is multiplied as weights to pixel position. CoM approach shifts the estimated center position of the BLOB more to the region concentrated with pixels with the highest brightness.

$$center\_position = \frac{\sum(pixel\_position * pixel\_brightness)}{\sum(pixel\_brightness)} \tag{6}$$

$$X\_position = \frac{\sum(X\_pixel\_position * pixel\_brightness)}{\sum(pixel\_brightness)} \tag{7a}$$

$$Y\_position = \frac{\sum(Y\_pixel\_position * pixel\_brightness)}{\sum(pixel\_brightness)} \tag{7b}$$

Since video frame is gray scale, flickering depends upon threshold value and the color gradient. In order to avoid a similar flickering in the computed center-position of the blob, it is recommended to running average(a running summation of the all pixel values during the detection phase followed by division of the number of pixels). CoM and Boundary Box are giving similar results for Blobs with perfect circular shape. But for the Blobs with motion blur, the modified Center-of-Mass based approach was closer to the expected optimal result.

IV. Implementation of Object Tracking Algorithm

A. Software Design Flow

DaVinci Software Architecture consist of three layers that are Signal Processing Layer (SPL),Input Output Layer (IOL) and Application Layer(APL). SPL take care of all the processing functions or algorithms that run on the device. Similarly, all the input and output, that is all the peripheral drivers are grouped into another layer that is Input-Output Layer (IOL).Also IOL generates buffers for input or output data buffers which reside in shared memory. The IOL drivers are integrated into an Operating System such as Linux OS or DSP/BIAS. IOL contains Video Processing Subsystem(VPSS) device driver used for video capturing and displaying. The third layer is the Application Layer which interacts with IOL and SPL. Also APL makes calls to IOL for data input and output, and to SPL for processing. The APL generates a master thread which is the highest priority level thread such as a video thread or an audio thread. The job of master thread is to handles the opening of I/O resources (through EPSI API), the creation of processing algorithm instances (through VISA API), as well as the freeing of these resources.

In case of video playback loop, video capture driver reads data from a input video port and starts storing into a input memory buffer. An interrupt is generated by the IOL to the APL, When this input buffer is full and the fully filled input buffer pointer is passed to the APL. Now APL generates an interrupt and passes that buffer pointer to the SPL. Now SPL layer processes this input buffer data and when data processing completes, it generates an interrupt back to the APL and transfers the pointer of the output buffer that it created. The APL passes this output buffer pointer to the IOL and also APL gives instruction to IOL for display output buffer using display driver. As we know that overhead passing the buffer pointers is negligible because here, only pointers to the memory location are passed while the buffers remain in that memory location.

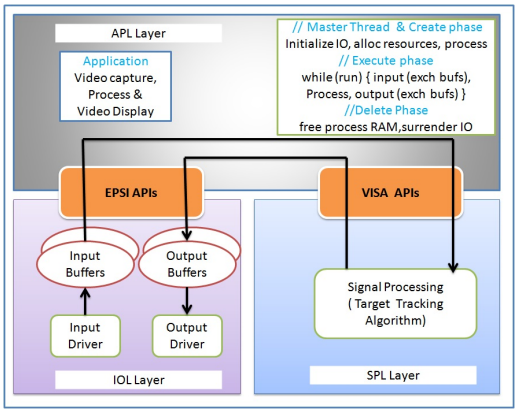


Fig. 5: Software Processor Flow

B. Sequence Diagram and API Design Flow

The control flow of the video tracking application is shown in Figure 7. The following is the sequence of operations: Initialize video capture device using VPFE_open() and VPBE_open(). Configure the video capture device using this API.The FVID handle is then used to configure the video input (composite, component, s-video), video standard (ntsc, pal, secam, auto), video file format (UYVY,YUYV, YUV420, etc.).

VPSS_getbuffer ,The VPSS driver has a queue of buffers for capturing video frames. Whenever a buffer is filled up, it is moved to the back of the queue.The application can obtain a buffer using FVID_dequeue() API. If a buffer is dequeued from the VPFE queue, the buffer depth at the VPFE driver decreases.

FVID_exchange() API removes a buffer from the VPFE buffer queue, takes a buffer from application and adds it buffer to the VPFE buffer queue.The buffer dequeued from the VPFE buffer queue is returned to the application as shown in Figure.6 . Close all VPFE and VPBF device using VPFE_close() and VPBE_close().

C. Object Tracking Function Flow

The steps involved in video surveillance tracking system are as shown in Figure. 8 as follows:

- Capture video sequence by CCD camera.
- Pre-process of the video frame free form noise.
- Segregating input image frame sequences from the input video frame.

API	EPSI APIs	DSP/BIOS	Function
Open: Initializes device	VPFE_open	FVID_create ()	Open a handle to VPFE device
		FVID_control ()	Configure the VPFE parameters Set the video input - Composite/S-Video / Component Set the video standard - NTSC/PAL/SECAM/AUTO Set the video format - UYVY/YUYV/YUV420/..
		FVID_alloc ()	After configuring the device, the function requests for allocation of buffers from the VPFE driver. These buffers are queued at the VPFE driver.
		FVID_queue ()	An application can obtain a buffer using FVID_dequeue() API. If a buffer is dequeued from the VPFE queue, the buffer depth at the VPFE driver decreases. Hence, it is usual for the application to queue one of its free buffers to the VPFE buffer queue through FVID_queue() call.
		FVID_dequeue ()	
Control: Used to configure device settings	VPFE_control	FVID_control ()	Configure the VPFE parameters
GetBuffer: Get the next ready buffer from queue	VPFE_getBuffer	FVID_exchange ()	This API removes a buffer from the VPFE buffer queue, takes a buffer from application and adds it buffer to the VPFE buffer queue. The buffer dequeued from the VPFE buffer queue is returned to the application
Close:Uninitialize the device	VPFE_close	FVID_free ()	The buffers allocated at the VPFE driver are freed using the FVID_free() API.
		FVID_delete ()	After FVID_free ,The device is uninitialized using FVID_delete() API

Fig. 6: EPSI APIs for DSP/BIOS

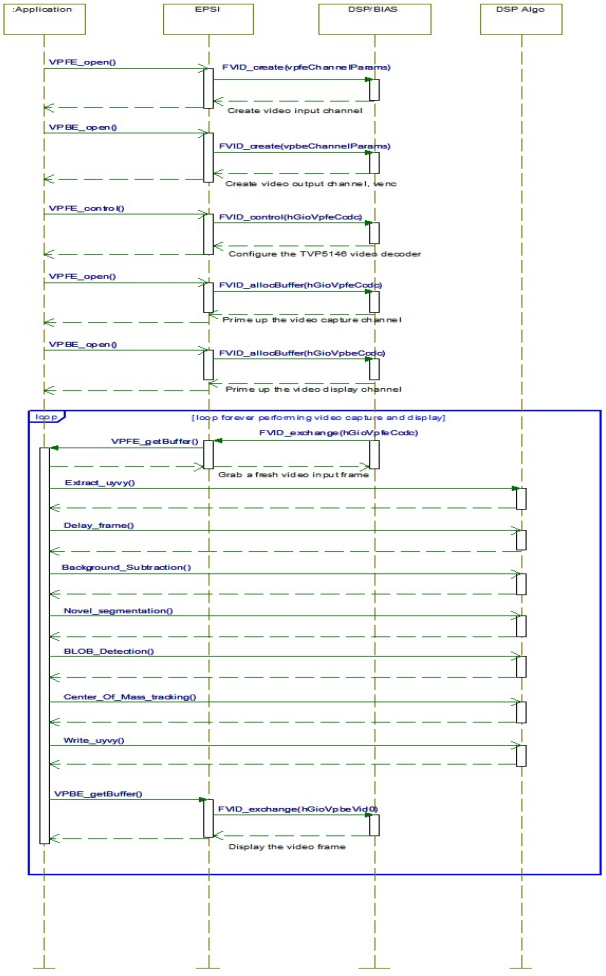


Fig. 7: Sequential Flow diagram object tracking algorithm

- Convert image from one domain to another domain and Down sample it.
- Delay Image frame for background subtraction.
- Finding of the moving objects for consecutive frames using background subtraction.
- Segmenting the regions where there is movement using fuzzyfier, mean and median filter operation.
- Blob detection using adjacency pixel value method.
- extracting features and marking objects tracking using edge or boundary box method.
- Tracking of particular object using center of mass method.

Steps	Procedure for tracking Alog	Implementation of tracking On DM6437
Step -1	Create video input and Output channel	FVID_create("/VPFE0",...) FVID_create("/VPBE0"...)
Step -2	Configure TVP5146 video decoder	FVID_control(VIDEO_DECODER_CONF)
Step -3	Allocate input and output frame buffers	FVID_allocBuffer(hGioVpfeCcdc, & frame BuffTable[i])
Step -4	Prime up the video display channel Grab first buffer from input queue	FVID_queue(hGioVpbeVid0, frameBuffTable[3])
		FVID_dequeue(hGioVpfeCcdc, &frameBuffPtr)
Step-5	Loop forever performing video capture and display	While(1){
Step-5 Part-I	Grab a fresh video input frame	FVID_exchange (hGioVpfeCcdc, &frameBuffPtr)
Step-5 Part -II	Tracking Algorithm	Extract_uyvy()
		Image Pre-Processing()
		Delay_frame()
		Backgound_Subtraction()
		Novel_Segmentation()
		BLOB_Detection ()
		Center_of_Mass_tracking()
		Write_uyvy()
Step-5 Part-III	Display the video frame	FVID_exchange(hGioVpbeVid0, &frameBuffPtr) }

Fig. 8: Steps for implement object tracking

D. Optimization techniques

1) Algorithm Optimization:

- Accuracy and efficient way for object segment using crisp fuzzyfier, smooth and median filter.
- Weighted Average method for Back ground subtraction.
- Boundary box matching for BLOB detection
- Center of mass based tracking.

2) Code Optimization:

- Changing Data type of Variable: using short data type instead of word data type.
- Speed Optimization is done using Temporary Variable which will avoid repeated computations
- Inbuilt Instruction: using Intrinsic operators like ADD2, SUB2,MPY, LDW and STW.
- Change Compiler Setting: -pm , -mt and -on
- DSP Pipeline Concept : using MUST_ITERATE

V. Profiling and Debugging Results

A. Profiling results

Profiling is used to measure code performance and make sure for efficient use of the DSP targets. Profiling is used on

TABLE I: Object tracking profiler data

Function name	Cycle Without OPT	With TempVar	With Inbuilt
Extract_uyvy_fun	1340201	1326798.99	1299995
Copy_frame_fun	94419	93474.81	91586
Frame_subtract_fun	28340201	28056798.99	27489995
Segmentation_fun	48849160	48360668.4	47383685
Blob_detection_fun	25124071	24872830.29	24370349
Centriod_loop_fun	15644100	15487659	15174777
Write_uyvy_fun	1234020	1221679.8	1196999
Total Cycle	120626172	119419910	117007387

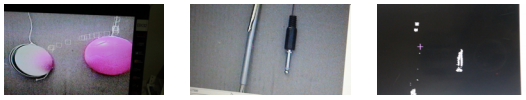
TABLE II: Object tracking profiler data

Function name	Cycle Without OPT	With Chagedatatype	With pipeline
Extract_uyvy_fun	1340201	1031955	1246387
Copy_frame_fun	94419	72703	87810
Frame_subtract_fun	28340201	21821955	26356387
Segmentation_fun	48849160	37613853	45429719
Blob_detection_fun	25124071	19345535	23365386
Centriod_loop_fun	15644100	12045957	14549013
Write_uyvy_fun	1234020	950195	1147639
Total Cycle	120626172	92882153	112182340

different function of Object tracking and the time taken to execute each function is measured by considering its inclusive and exclusive cycle, access count and processor clock frequency as shown in profiler data table I and II.

B. Debugging results

Debugging results of object tracking is shown in the following Figure 9 and Figure 10. Result of the background subtraction with filtering, segmentation, Blob detection and center of mass tracking is shown in Figures.



(a) (b) (c)

Figure 9: Debugging results : (a)Two ball tracking, (b)Two object tracking, and (c)Background subtraction with filtering

(a) (b) (c)

Figure 10: Debugging results : (a) Segmentation with filtering, (b)Blob Detection (c)Center of mass based tracking

VI. Conclusion

An object tracking algorithm is developed and implemented on DaVinci Processor (DM6437), using weighted average based on background subtraction, BLOBs detection using adjacent pixel value, identification using boundary box algorithm, and tracking of the object based on center of mass method. Presence of crisp fuzzier, smoothing and median filters processes play a major role in processing pixel data in image segmentation, which solved the salt pepper noise. Background and illumination changes problem is solved using weighted average based on background subtraction. Number of object, size of object and abrupt object shape problems of tracking are solved using BLOBs detection using adjacent pixel value, identification using boundary box algorithm. Tracking of the object is based on center of mass method, which improves robustness w.r.t occlusion and contour sliding. These approaches were optimized, both in code level and also in the algorithm. Based on profiling results, this object tracking approach as implemented on DaVinci processor is efficient and much faster for real time scenarios.

Acknowledgment

I express my sincere thanks and deep sense of gratitude to my supervisor Prof. V Rajbabu for his invaluable guidance, inspiration, unremitting support,encouragement and for his stimulating suggestions during experiment. I acknowledge with thanks to Mr. Vijay, Mr. Alok, Mr. Sarat and My family who have directly or indirectly co-operate me throughout my research activity.

References

[1] Hinz. S., ‘‘FAST AND SUBPIXEL PRECISE BLOB DETECTION AND ATTRIBUTION”,Technical University Munich, Germany.

[2] Uy. D. L.,‘‘An algorithm for image clusters detection and identification based on color for an autonomous mobile robot”, Research report submitted to Hampton university, Verginia, 1994

[3] Patro. B. N. ,‘‘Real-Time Video and Image Processing for Video Surveillance using DaVinci Processor”, Master of Technology, Indian Institute of Technology Bombay, Mumbai,India, June 2012.

[4] Bochem. A., Herpers. R., and Kent. K. B., ‘‘Hardware Acceleration of BLOB Detection for Image Processing”, Third International Conference on Advances in Circuits, 2010, pp. 28-33.

[5] Pawate. B. I. , ‘‘Developing Embedded Software using DaVinci&OMAP Technology” Margon & Claypool, 2009.

[6] Nascimento. J. C., Abrantes. A. J., Marques. J. S., ‘‘AN ALGORITHM FOR CENTROID -BASEDTRACKING OF MOVING OBJECT. ”, Lisboa, portugal, October 1994.

[7] Xiang. H. I., Xiao-ning. F.,Guo-qiang. H. , ‘‘An Optimization for Distance Estimation in Real-Time with DM6437 ”, 4th International Conference on Signal Processing Systems,2012.

[8] Frank.T., Haag.M., Kollnig.H., and Nagel. H.-H., ‘‘Tracking of occluded vehicles in traffic scenes.”, European Conference on Computer Vision-Volume II, pages 485494, London, UK, 1996.

[9] Agarwal. S., Mishra. S., ‘‘A STUDY OF MULTIPLE HUMAN TRACKING FOR VISUAL.”, IJAET,ISSN: 2231-1963,Nov. 2012.

[10] Hedberg. H. , ‘‘A Survey of Various Image Segmentation Techniques.”, Dept. of Electroscience, Lund University, Sweden.

[11] Piccardi. M. , ‘‘Background subtraction techniques: a review.”, IEEE International Conference on Systems, Man and Cybernetic, 2004.

[12] Texas Instrument Inc., ‘‘TMS320C6000 Optimizing Compiler v7.0 User’s Guide.”, Spru187Q, 2010.2.