A Smart Camera Application: DSP-Based Human Detection and Tracking

V. Cheng and N. Kehtarnavaz
DSP Lab, Department of Electrical Engineering,
Texas A&M University, College Station, TX 77843
kehtar@ee.tamu.edu

ABSTRACT

As an application of smart cameras, this paper presents an algorithm and its real-time implementation for detecting and tracking multiple people moving in a surveillance scene. The algorithm is devised to work for low resolution situations and any camera angle. An adaptive background subtraction module is used to obtain candidate moving target areas. A body detection module is then activated to locate heads. The detection of moving target areas is enhanced by using a predictor. The algorithm is implemented on the TI TMS320C62 fixed-point DSP processor for real-time processing of video signals. This real-time capability allows field deployment for the purpose of collecting data such as number of people and their crossing frequency in a certain area. So far a processing time of nearly 56 msec per frame is achieved.

1. INTRODUCTION

With the increase in the use of surveillance cameras, the need for automatic processing of surveillance scenes is rapidly growing considering that manual monitoring is labor-intensive and not cost-effective. Smart cameras have started appearing in the market with on-board intelligent processing capabilities. As an application of smart cameras, this paper presents an algorithm and its real-time implementation on a high speed DSP processor to detect and track people moving in a surveillance scene. The algorithm is devised to work in real-time for low resolution images captured at any camera angle.

The selection of effective object features is the key issue in object detection/tracking. Most real-time tracking algorithms have been designed to track a single target having relatively high resolution in the image such as a person [1], a hand[2], or a face [3] under controlled lighting conditions.

An approach widely used is template matching. To cope with variations in the appearance of an object, an adaptive version of this approach has been reported in [4]. There are difficulties associated with this approach when variations are not slow and smooth, and there exists the possibility of targets occluding each other.

Shapes of moving objects have also been used as tracking features. If the background is stationary, the shape information can be easily retrieved via background subtraction and edge detection. In general, a mathematical model is constructed to describe the shape information. Baumberg[5] has used B-splines to model the silhouette of a pedestrian with 40 control points. Although the shape approach is effective for tracking a single moving target, it has difficulties when several targets such as people walk close to each other or in groups. At one moment one silhouette can contain several people and later get split into several separate silhouettes.

For object tracking, Kalman filtering is often used. However, Kalman filtering requires the density function for object features be unimodal. The presence of background clutter, self-occlusions, and complex dynamics during multiple objects tracking result in multi-modal density functions. One solution is to use non-parametric probability models such as done in

the condensation algorithm developed by Isard and Blake [3]. This tracking approach is computationally demanding and needs a large number of training samples. Intille, et al. [6] have demonstrated that tracking non-rigid multiple objects at low-resolution is a complicated task and requires using contextual information to drive the tracker.

The detection/tracking approach developed here attempts to satisfy two constraints at the same time: being fast enough to run in real-time on a DSP processor and being robust enough to work for any camera angle and low resolution images. This approach consists of three steps: motion detection, body detection, and tracking. A block diagram of the system is shown in Figure 1. Sections 2, 3 and 4 discuss the motion detection, the body detection, and the tracking steps, respectively. Section 5 includes the calibration process and Section 6 the real-time DSP implementation issues.

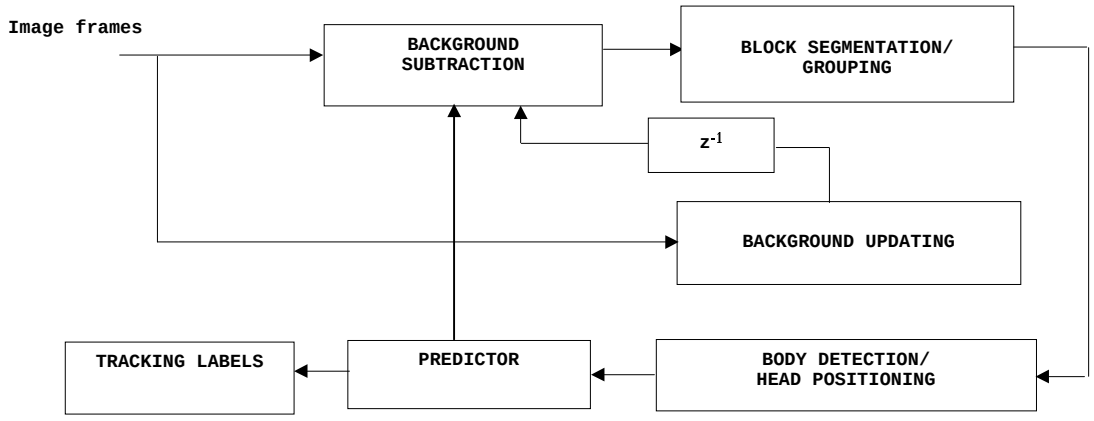


Figure 1: Detection/tracking block diagram

2. MOTION DETECTION

The first step involves identifying the location of activity or motion areas in the image corresponding to people. Since the camera is considered fixed, intensity differences between image frames and the background image can be used for this purpose. This approach is referred to as background subtraction in the literature. In general, those pixels that exhibit a difference value above a specified threshold value denote motion or activity. However, background subtraction is sensitive to image noise when it is done per pixel basis. To lower this sensitivity, it is appropriate to carry out the subtraction operation per block basis rather than per pixel basis. Each block is represented by the average intensity of all the pixels contained in it. In this manner, blockwise background subtraction would indicate the blocks different than the background as caused by moving objects. The block size is chosen according to the size of the moving people in the scene and is set during a calibration process to be discussed later. It should be noted that pixelwise background subtraction is still used to retrieve a finer description of people but this is done locally within an activity area.

This detection scheme works well under the assumption that lighting conditions do not alter. However, in a scene illuminated by outdoor lighting, this is not the case. The background needs to be updated to incorporate possible changes in luminance. It is possible to compute a background image by taking the average of image frames over a relatively long time or by taking the median of last few frames. This allows an effective mechanism to retain the permanent background information by eliminating temporary, moving objects. Rajkotwala and

Kehtarnavaz [7] have implemented an efficient way to perform background updating without storing all the image frames. The following equation is used in their update:

$$B(i,j)=(1-\alpha)*B(i,j)+\alpha*I(i,j),$$

where B denotes the background image within a block, I the current image frame within the same block, and (i,j) indicate the (col,row) index of a block. At each frame sample, the background incorporates a small percentage of the current frame. As a result, a stationary object is completely incorporated into the background image block B after $1/\alpha$ frames or after $1/(\alpha*R)$ seconds if R is the frame rate. The parameter α is set during the calibration process.

The above detection step is improved by making use of the predicted positions of activity blocks as provided by the tracking algorithm discussed later. This improvement is achieved by raising the detection level of those blocks predicted by the tracking algorithm. The binary image obtained from the background subtraction is then used to assign the same label to connected binary points.

3. BODY DETECTION / HEAD POSITIONING

Now that the activity or motion areas are detected, a feature that is not computationally intensive and is relatively unique to human silhouette needs to be extracted. The vertical major axis that goes from the head through the entire body is used for this purpose.

First, within each activity area, a pixelwise background subtraction is done to obtain a binary mask incorporating the pixels not belonging to the background. Then a vertical projection histogram is computed. Each bin of the histogram corresponds to the number of white pixels in the corresponding column of the binary mask. As shown in Figure 2, a local maximum in the histogram corresponds to the location of the major axis of a person's silhouette. This body detection approach is particularly useful when there are multiple people in the activity area as shown in Figure 3.

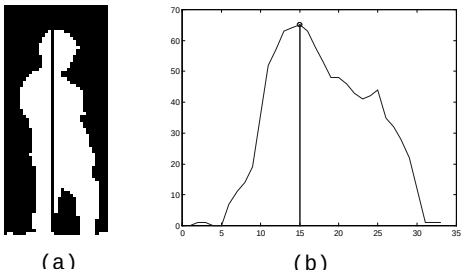


Figure 2: (a) Binary mask of a person's silhouette with its major axis. (b) Corresponding vertical projection histogram, the peak corresponds to the major axis of the silhouette.

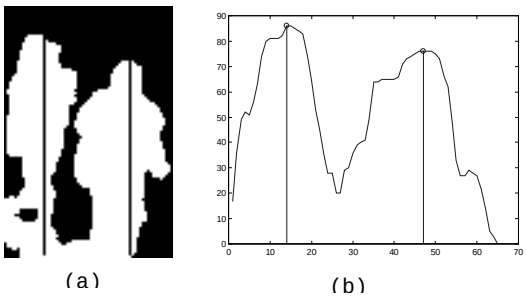


Figure 3: (a) Binary mask of an activity area containing 2 persons. (b) The corresponding histogram with two detected major axes.

To cope with the presence of image noise causing false peaks, the histogram is smoothed prior to the extremum search. Duda and Hart in [8] give an efficient mechanism for both smoothing and finding the maxima by a single scan of the histogram values. The technique, called hysteresis, is parameterized by the width of the smoothing window. The smoothing parameter depends on the distance between the camera and the field of view or the size of the moving people. This parameter is set during the calibration process.

Considering a major axis gives the location of a human body, the position of the head can be deduced from it. The location of the major axis already gives the x-coordinate

of the head. The y-coordinate is estimated to be on the major axis a few pixels (half a person's head) from the first non-zero value. The size of the head is set during the calibration process.

4. TRACKING ALGORITHM

For tracking purposes, the following state vector V is attached to each detected head:

$V=[x,y,v_x,v_y,vm_x,vm_y]$

x: x position of the head
y: y position of the head
v_x: velocity of x position
v_y: velocity of x position
vm_x: average velocity of x position over N frames
vm_y: average velocity of y position over N frames

x and y position components are available immediately after the head detection step whereas the velocity components are computed two frames after the first appearance of the head. The tracking effect is made visible by attaching a number label to each head in the scene. Thus, a head being tracked would keep the same number label along consecutive frames until it exits the field of view. A new person entering the field of view is assigned a new unused number label.

The way each head is tracked in the field of view is by performing a simple prediction. The basic idea is to observe, predict and then update the observations according to the prediction. In the updating stage, each object is relabeled by using the prediction made based on the previous frame. For instance an object labeled "0" at the observation stage is relabeled "1" during the update stage if it is very close to the predicted position of the object "1" in the previous frame. The update stage is crucial to the efficiency of the tracking algorithm.

By including some contextual information, different situations such as crosspaths can be handled. The contextual information considered include: a target that has been successfully tracked for several frames cannot disappear in the middle of the scene unless it is close to the image boundaries. Also, the direction of velocity cannot be reversed from frame to frame.

The field of view represents a closed world context, while some people may enter into it, others may walk out or still be inside. For the tracking to be effective, one must take this into account. Hence, new objects can only appear in delimited regions called "**source**". The source for examples can be the four image boundaries or a specific region of interest within the image. In the same manner, objects cannot disappear anywhere but only if they fall into a region named "**well**". Source and well can be represented by the same region. Given these constraints, any change in the number of targets in the field of view corresponds to the number of incoming objects minus the number of exiting objects. The update is done as follows. Each object is assigned a state that mainly reflects how confidently it is to be tracked. Thus the position of each object is predicted according to the following possible states:

- **Newcomer**: It denotes the characteristic of an object that just entered into the field of view and lies in the source region. No prediction is done for it since no velocity is available. In the next frame, the nearest object is assigned the same label.
- **Traceable**: An object reaches this state when it keeps the same label during the first two frames it is in the field of view. This state naturally follows the **Newcomer** state. The instantaneous velocity is then computed using the two previous positions and its next position is predicted.
- **Predicted**: Usually one frame after an object becomes **traceable**, it reaches the **predicted** state, often at the third frame. In this state, the object is stable and its label should not change. An object keeps the **Predicted** state until its exit. The mean velocity is computed over the previous frames. A simple mechanism is used to cope with the occlusions of an object until it reaches the outbound of the field of view. The update algorithm attempts to find the closest match in the current frame. First, the Euclidian distance is used for the match. Then, once the closest object in the current frame is found, the new instantaneous velocity is

computed and its direction is compared to the direction of the mean velocity. If they are in opposite directions, then the object is discarded and the process is repeated for the next closest object. This allows coping with the problem of crosspaths and occlusions. For each predicted head of the past frame, there should be a match in the present frame. If no match is found, it is likely the head is occluded and a new object is created at the predicted position.

- **Out**: When the predicted position falls into a "**well**", the object is assigned the **out** status. It will not be tracked any more and its label is made available to the next **newcomer**.

If for a particular head in the present frame, no match is found among the predicted or traceable heads of the past frame, the person is assigned a newcomer state. However, false positive detection can still occur in the source region because the detection is not perfect for all situations, resulting in the presence of an incorrect newcomer state. To prevent such "newcomers" from being upgraded to the "traceable" state, the algorithm is designed to delete them if the Euclidian distance between the initial position and the closest present match is higher than a certain value, set during the calibration process. Likewise, people entering the "source" region can be missed. Particularly when people enter in groups, there might be several heads missed. One solution to overcome this is to enlarge the "source" region. But this could diminish its effectiveness. Therefore, objects appearing outside the source that have not been assigned to any predicted object are still kept in the system memory but are assigned a new state "ghost". The system would then decide later if those "ghosts" can be upgraded to a "traceable" state by checking if their trajectory is consistent or their velocity does not vary much over time.

This tracking scheme allows multiple objects to be tracked when their trajectories remain almost linear or when the change in their directions happens smoothly. Occlusions in general do not affect performance as long as they last for a short time. Another issue is that the background may not be uniform and a moving object may go through regions where the background luminance is very close to its own. The consequence of this is that the object may fall below the detection threshold resulting in a missed head. One way to cope with this is to use the prediction information to temporarily lower the detection threshold for regions likely to contain the next position.

5. SYSTEM CALIBRATION

Some parameters used by the algorithm depend on image resolution and camera distance. These parameters include block size, width and height of an individual, threshold used for background subtraction, and head detection parameters. These parameters are expected to be set during an initial calibration process.

The first parameter as part of the calibration is the coefficient α used for the background update. This parameter reflects how fast current frames are incorporated into the background. In outdoor scenes, the brightness naturally evolves during the day. Hence, α should be set to accommodate for this. One way to do this is to capture a steady scene until the overall brightness change causes 50 % of the blocks to be detected. The time T elapsed can then be used to compute α=R/T, where R is the frame rate.

The next stage of the calibration consists of selecting the block size and the threshold for background subtraction. Block size must be large enough in order for image noise not to get detected. For instance, if the scene contains trees, the block size should be set to a size to avoid detecting the motion of leaves. This part of the calibration is done visually by examining detected blocks. The operator increases the size of the block until image noises disappear. On the other hand, if the block size is set to be larger than heads, the head detection would be less effective. One must make sure that a head is covered by

several blocks. Note that the detection of shadows cannot be suppressed by increasing the block size since shadows and people have more or less the same dimensions. However, if the threshold for background subtraction is set high enough, shadows can be prevented from getting picked up. Care must be taken not to set a very high threshold value since such a threshold may degrade the detection in background regions with poor contrast.

The calibration is continued by collecting statistics on the size of a single moving person. This information is used thereafter by the body detector to parameterize the histogram smoothing window. Basically for this stage of calibration, a person is asked to walk in the field of view of the camera for the system to collect statistics which include mean and variance of the width and height of the bounding rectangles. In order to prevent the data from getting corrupted by other situations, the person should walk in background regions having good contrast.

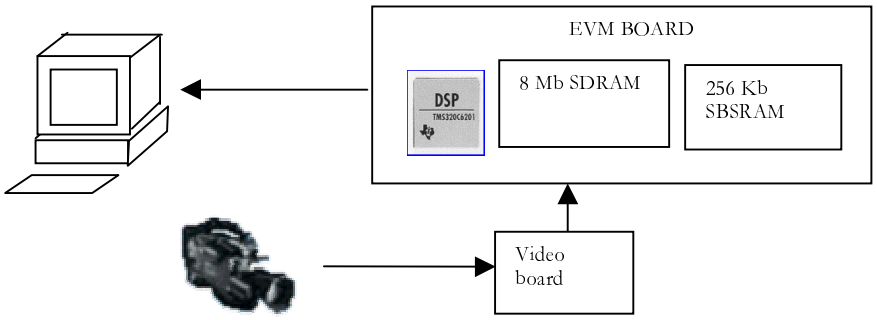


Fig 4: Hardware setup

6. REAL-TIME DSP IMPLEMENTATION

The above algorithms have been implemented on the TI TMS320C6201 DSP processor as part of the C6x evaluation module (EVM) board running at 167 Mhz. A videoboard provided by TI is connected to the EVM board for video data acquisition. This video daughterboard acquires image frames from an analog input video signal and stores digitized image frames on the on-board memory of the EVM, as shown in Figure 4. The amount of on-board memory is 256 Kilo-bytes of SBSRAM running at a maximum speed of 133Mhz and 8 Mega-bytes of SDRAM running at one-half the CPU clock rate. As can be seen, the entire system is slowed down by its memory access. However, there is 64 Kilo-bytes of on-chip program space called internal program memory (IPM) and 64 Kilo-bytes of on-chip data memory capable of running at the full speed. Here, IPM is used as cache memory to speed up the execution.

In order to increase the throughput of the above algorithms, steps are taken to make sure that these algorithms are properly optimized for the C6x VLIW architecture. Although the C-compiler optimizer is used to do the optimizations, some routines have been hand coded in linear assembly in order to improve the throughput. A host program running on an intel-based PC host is used to communicate with the EVM. It is used to display images in real-time and to send commands to the algorithms running on the EVM. Along with the videoboard, a software interface provided by TI, called Network Camera Operating System (NCOS), is used to communicate with the videoboard. NCOS is based on the real-time SPOX operating system. The host program running on the PC uses different libraries provided by the EVM API functions to transfer data to and from the EVM. Most of the algorithms are implemented in C. However, it was found that the block averaging routine (which accumulates all the pixels contained in a block to produce their average) was the most time consuming routine so it was rewritten in assembly.

The C6x has several DMA channels that allow background memory transfers without

interfering with the CPU operation. This feature has been used here for moving frames from the input video buffer to the processing buffer and also for initializing memory. In order to compute velocity vectors, the time between frames is measured by one of the C6x timer whose counter is incremented every 4 cycles.

The processing time depends on the number of people present in the field of view. If many people are in the field of view, the time required for region segmentation increases accordingly. As shown in Table 1, 65% of the processing time is dedicated to receiving the image data from the videoboard and to sending the result to the PC host. This is due to the fact that the board used is a developmental board and many of the I/O routines are not yet optimized. Besides in an actual smart camera device, the videoboard will be able to output the image buffer to a monitor, hence eliminating the need for the program to send the image to a PC host. Given the I/O constraints, currently a processing rate of 4 to 7 frames per second is achieved depending on the number and size of people in the field of view. Easing the I/O constraints can double this processing rate.

Figures 4-a through 4-l illustrate sample frames for different camera angles in a lobby.

7. CONCLUSION

With their rapid growth in their processing speed, the latest DSP processors allow computational intensive video analysis algorithms to be implemented in real-time as indicated by the computationally demanding detection/tracking algorithms presented here. Such a human detection/tracking system is suited for security tasks including surveillance and public area monitoring. For example, if a number of smart cameras is placed in a network, a person can be tracked inside a large area or within a building. This system can also be used to collect data such as number of people, and rate of people walking in a certain area.

ACKNOWLEDGEMENTS

This research has been funded in part by the TI sponsored DSP Program at Texas A&M University. The authors would like to thank Dr. Leonardo Estevez from Texas Instruments for providing the videoboard for this project and for his beneficial comments.

REFERENCES

[1] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time Tracking of the Human Body. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1997, vol. 197, pp. 780-785.

[2] M Isard and A. Blake: Condensation – Conditional Density Propagation for Visual Tracking. *In International Journal on Computer Vision,* 1998, vol. 29, pp. 5-28.

[3] Richard J. Qian, M. Ibrahim Sezan and Kristine E. Matthews: A Robust Real-time Face Tracking Algorithm *In Proceedings of International Conference on Image Processing*, 1998, vol. 1, pp. 131-135.

[4] A.J. Lipton, H. Fujiyoshi, R.S. Patil: Moving Target Classification and Tracking from Real-time Video. *In Proceeding of Fourth IEEE Workshop on Applications of Computer Vision*, Oct 1998, pp. 8-14.

[5] A M Baumberg. An Efficient Method For Contour Tracking using Active Shape Models. *In Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, 1994, pp. 194-199.

[6] S. Intille, J. Davis, A. Bobick, Real-Time Closed-World Tracking, *In Proceedings of the Fifth International Conference on Computer Vision,* 1995, pp. 672-678.

[7] F. Rajkotwala and N. Real-Time Vision-based Detection of Waiting Pedestrians, *In Real-Time Imaging,* 1997, pp. 433-440.

[8] R. Duda and P. Hart: Local Extrema of Figure Boundary, *In Pattern Classification and Scene Analysis. Wiley-Interscience Publication,* 1973, pp. 354-355

.

Image transfer from video board	54 ms	33.3 %
Block averaging	31 ms	19.2 %
Image subtracting, background updating	3 ms	1.85%
Block segmentation, human body detection, head positioning	22 ms	13.6 %
Tracking: predict, observe and update	0.005 ms	0.003%
Display label information	0.32 ms	0.2 %
Sending image to the PC host	51 ms	32 %
Total time	162 ms = 6.2 frames/s	

Table 1: Timing table

(a)

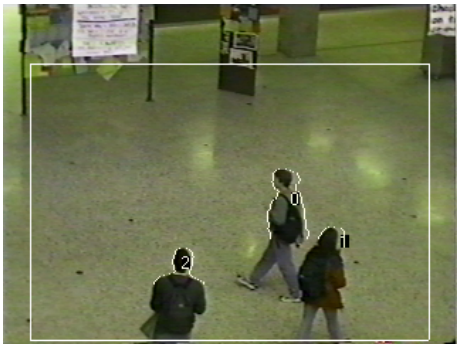
(e)

(i)

(b)

(f)

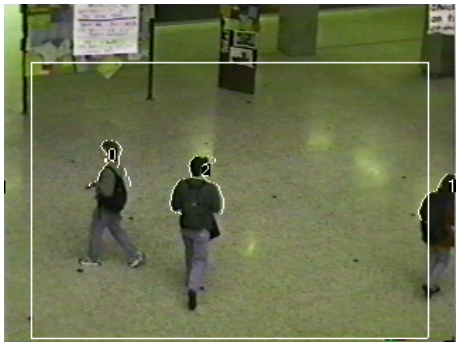
(j)



(c)

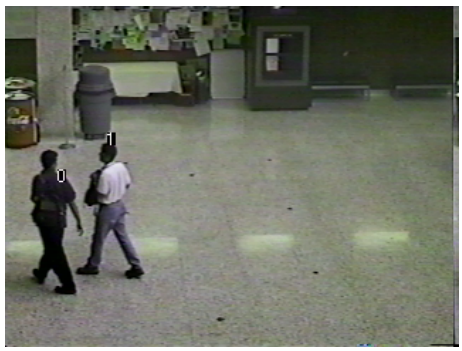
(g)

(k)



(d)

(h)



(l)

Fig 4.a – 4.d Lobby 1
Three people are walking in the field of view. In frame (c), #0 and #1 persons walk close to each other (their shape merges) but the system is able to keep track.

Fig 4.e – 4.h Lobby 2
A group of four people are crossing the lobby. It can be noticed that one person in the group is not detected due to occlusion.

Fig 4.i – 4.l Lobby 3
Two people walking together and being detected.