2009 Advanced Video and Signal Based Surveillance

An abandoned object detection system based on dual background segmentation

A. Singh, S. Sawan, M. Hanmandlu
Department of Electrical Engineering
I.I.T. Delhi
Delhi, India
abhinavkumar.singh@mail2.iitd.ac.in

V.K. Madasu, B.C. Lovell
School of ITEE
NICTA and The University of Queensland
Brisbane, Australia
v.madasu@uq.edu.au

***Abstract*— An abandoned object detection system is presented and evaluated using benchmark datasets. The detection is based on a simple mathematical model and works efficiently at QVGA resolution at which most CCTV cameras operate. The pre-processing involves a dual-time background subtraction algorithm which dynamically updates two sets of background, one after a very short interval (less than half a second) and the other after a relatively longer duration. The framework of the proposed algorithm is based on the Approximate Median model. An algorithm for tracking of abandoned objects even under occlusion is also proposed. Results show that the system is robust to variations in lighting conditions and the number of people in the scene. In addition, the system is simple and computationally less intensive as it avoids the use of expensive filters while achieving better detection results.**

Keywords- video surveillance, left baggage detection, background segmentation, tracking

I. Introduction

Recent years have seen a stark rise in terrorist attacks on crowded public places such as airports, train stations and subways, nightclubs, shopping malls, markets, etc. Many surveillance tools have been employed in the fight against terror. Although video surveillance systems have been in operation for the past two decades, the analysis of the CCTV footage has seldom ventured out of the hands of human operators. Recent studies [1-3] have brought into fore the limits to human effectiveness in analyzing and processing crowded scenes, particularly in video surveillance systems consisting of multiple cameras.

The advent of smart cameras with higher processing capabilities has now made it possible to design systems which can possibly detect suspicious behaviors (in general) and abandoned objects (in particular). A number of algorithms [5, 7, 8] have been suggested in the recent past to deal with the problem of abandoned-object-detection. Due to their dependence on complex probabilistic mathematics, most of these algorithms have failed to perform satisfactorily in real time scenarios. In addition, the other difficulty of detecting an abandoned object under occlusion adds to the overall complexity. Some proposed algorithms [4-5] have dealt with partial occlusion (by moving people) but complete or prolonged occlusion (by another object) has not yet been tackled. Furthermore, the background subtraction methods employed in the above methods are either computationally intensive or lack dynamically updating features.

In this paper, we present an abandoned object detection system based on a simplistic and intuitive mathematical model which works efficiently at QVGA resolution which is the industry standard for most CCTV cameras. The proposed system consists of a novel self-adaptive dual background subtraction technique based on the Approximate Median model [6] framework. Algorithms for tracking abandoned objects with or without occlusion are also included.

A. System Overview

The overall system (see Fig. 1) is modular in nature and consists of four disparate blocks with each block acting as a discreet processing unit making it easy to modify any block, provided the input and output data types remain compatible with the connecting blocks. The 4 blocks are: Data extraction and conversion unit; Background subtraction module; Still object tracking and occlusion detection block and Alarm raising and display of result unit

A live video stream is initially segmented into individual images from which a region of interest is extracted and converted to 3D intensity matrices (height * width * intensity value of each pixel). These matrices are then fed as input to the Background Subtraction module.

II. Background Segmentation

Numerous background subtraction methods are available in the literature. The most popular being the ones based on Gaussian mixture models, the first of which was proposed by Friedman and Russell [12] and then modified by several authors [13-14] to suit their specific needs. In this work, a new background subtraction technique based on the Approximate Median algorithm is developed. This method is adaptive, dynamic, non-probabilistic and intuitive in nature. Like the majority of other methods (for ex. [6]), we also use pixel color/intensity information for background processing. But instead of having one reference frame, we maintain two different reference frames for self adaptability resulting in less computation due to non-inclusion of any complex mathematics. Moving crowd/objects, lighting changes and unnecessary details like shadows, reflections on floors and walls are filtered off efficiently with only stationary objects remaining in the scene, thus leaving us with the prime motive of 'detecting abandoned objects'. Moreover, having two backgrounds has an added advantage that the user can adjust the time interval between the update of reference background frames to suit different needs and environments.

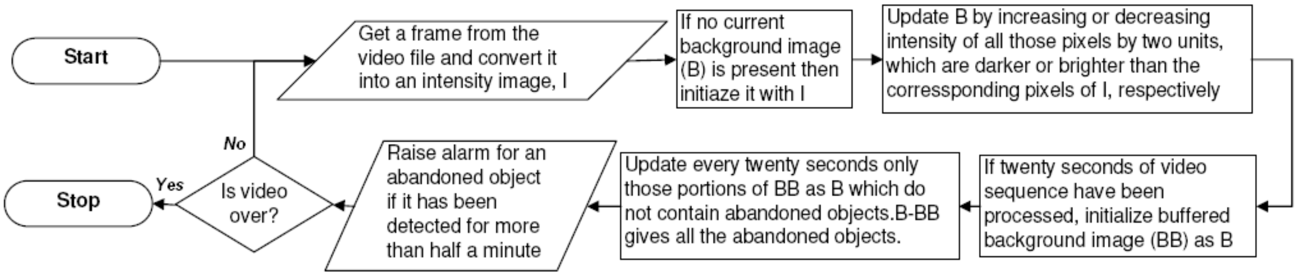

Start
Get a frame from the video file and convert it into an intensity image, I
If no current background image (B) is present then initiaze it with I
Update B by increasing or decreasing intensity of all those pixels by two units, which are darker or brighter than the corressponding pixels of I, respectively
If twenty seconds of video sequence have been processed, initialize buffered background image (BB) as B
Update every twenty seconds only those portions of BB as B which do not contain abandoned objects.B-BB gives all the abandoned objects.
Raise alarm for an abandoned object if it has been detected for more than half a minute
Is video over?
No
Yes
Stop

Figure 1: Flowchart of the overall system operation

A. Algorithm

The proposed algorithm to separate background and foreground in the incoming image is based on the 'Approximate Median Model' [6]. However, our technique requires two reference background images, namely, 'Current Background' and 'Buffered Background'. This technique of storing two backgrounds can be considered as a dual background method. One of the interesting features of this technique is that both the backgrounds are updated dynamically. The first one is updated frequently while the second one has a slower update rate.

The first frame of the incoming video is initialized as 'Current Background'. Subsequently, the intensity of each pixel of this current background is compared with the corresponding pixel of the next frame (after every 0.4 seconds). If it is less, then the intensity of that pixel of current background is incremented by one unit, otherwise it is decremented by one unit. In case of equality, the pixel intensities remain unchanged. This way, even if the foreground is changing at a fast pace, it will not affect the background but if the foreground is stationary, it gradually merges into the background.

Since we are interested in all those objects which are stationary for a long period of time (and thus have gradually merged into the background), we maintain another set of background images called 'Buffered Background'. Here, all those pixels which do not belong to the prospective abandoned objects set are made equal to that of 'Current Background'. This is done at an interval of every 20 seconds.

Difference of the two backgrounds is represented as a binary image with the white portion representing foreground (blobs).

B. Illustration

The Dual Background technique is illustrated in Fig. 2. Frame 2A shows all the objects that are detected. Frame 2D shows the current background, which is updated every half a second. The longer a person or object stays in A, stronger its impression is imprinted on D. Frame 2E shows the buffered background, which is updated every half a minute, and does not contain abandoned object(s). Hence the difference of 2D and 2E gives the position of abandoned objects, which is highlighted in frame 2C, after the object has been left abandoned for a long enough time. Frame 2B shows the foreground which comes from difference of 2A and 2E.

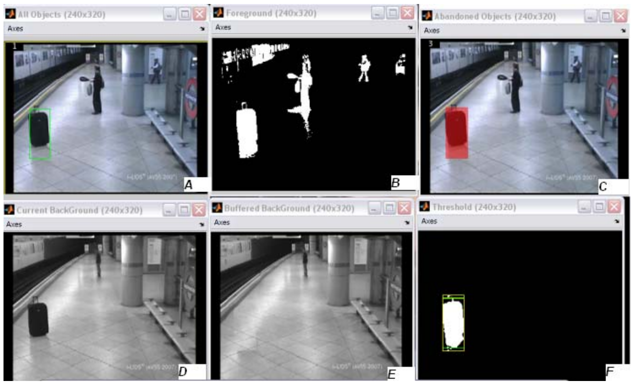

All Objects (240x320)
Foreground (240x320)
Abandoned Objects (240x320)
Current BackGround (240x320)
Buffered BackGround (240x320)
Threshold (240x320)
A
B
C
D
E
F

Figure 2: Dual Background Segmentation

III. Object Detection

In this module, we divide the binary image from the previous unit into a number of legitimate blobs (rectangular regions enclosing continuous regions of foreground). Once the blobs and their various properties like area, centroid position etc. have been generated, we apply the tracking algorithms.

A. Mathematical Model

Let us suppose that after blob analysis we get 'N' number of blobs, each with its enclosing region 'R_n (t, l, h, w)', its area 'A_n' and centroid 'C_n(i,j)'.

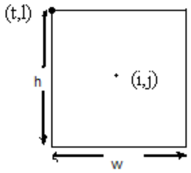


Figure 3: A typical blob

where,

t is the top position value of pixel; *l* is the left position value of pixel; *h* gives the height of the blob; and *w* is the width of the blob; and $1 \leq n \leq N$

Let 'T' be the set of tracked blobs such that,

$$T = [B_n : B_n = \{R_n(t,l,h,w), A_n, C_n(i,j), t_n, m_n\}]$$
$$\forall 1 \leq n \leq M$$

where,

M is the number of tracked blobs; t_n is the number of frames for which the blob has been tracked and m_n is the number of consecutive frames for which the blob being tracked previously has been not detected

Let us call the present set of blobs which we get after analysis of the present frame as ‘P’ and its objects as b_n , which are N in number. Then the set of blobs is:

$$T = \left[b_n : b_n = \{r_n(t,l,h,w), a_n, c_n(i,j)\}\right]$$
$$\forall 1 \le n \le N$$

B. *Blob detection*

The blob analysis takes as an input a binary image, applies an algorithm similar to the one described in [11] and returns various properties of the detected blobs like bounding box, area, centroid position etc. A simplified version of the algorithm is as follows:

1. Create a region counter.
2. Scan the image from left to right and from top to bottom.
3. For every pixel check the north and west pixel (4-connectivity) or the northeast, north, northwest, and west pixel (8-connectivity) for a intensity value of 1 in the binary image (termed as criterion of blob analysis)
4. If none of the neighbors fit the criterion then assign to region value of the region counter. Increment region counter.
5. If only one neighbor fits the criterion, assign pixel to that region.
6. If multiple neighbors match and are all members of the same region, assign pixel to their region.
7. If multiple neighbors match and are members of different regions, assign pixel to one of the regions and indicate that all of these regions are the equivalent.
8. Scan image again, assigning all equivalent regions the same region value.

C. *Tracking*

The next process in object detection is tracking the different blobs so as to find which blobs correspond to abandoned objects. The first step in this process is to create a set, *Track*, whose elements have three variables: *blob-Properties*, *hitCount* and *missCount*. The next step is to analyze the incoming image for all the blobs. If the area change and the centroid position change, as compared to any of the elements of the set *Track* are below a threshold value, we increment *hitCount* and reinitialize *missCount* with a zero; otherwise we create a new element in the Track-set, initializing the blob-properties variable with the properties of incoming blob and *hitCount* and *missCount* are initialized to zero. We then run a loop through all the elements of the set. If the *hitCount* goes above a user defined threshold value, an alarm is triggered. If the *missCount* goes above a threshold, we delete the element from the set. These two steps are repeated until there are no incoming images.

Pseudo Code for Tracking

```
Take area, centroid, bounding boxes (bbox) and total
number of blobs (n) as input from Blob Analysis block. Let
Track=empty set of vectors of type t where t=(area,
centroid, bbox, hitcount, misscount,active,occluded)

m= Track.Size
For i=1 to n
 c=0
 For j=1 to m
   If (percentage background in Track[j].area<50)
   Then Track[j].occluded=true
    End
    If (|area[i]–Track[j].area|/area[i] <.05 and |centroid[i]–
      Track[j].centroid|/centroid[i]<.05)
    Then Track[j].active =true, c=1,
     break from loop
   End
  End
 If c=0
  Then k=Track.size++,
          Track[k].area = area[i]; Track[k].centroid =
centroid[i];
          Track[k].bbox = bbox[i]; Track[k].hitcount = 1;
          Track[k]. active = true;
  End
End
m= Track.Size
For j=1 to m
 If (Track[j].active==true)
 Then
       Track[j].hitcount=Track[j].hitcount+1;
       Track[j].miscount=0;
  If (Track[j].hitcount > 4)
    Don't update pixels of Track[j].bbox in buffered
background
  End
   If (Track[j].hitcount > 40)
     Then raise alarm for Track[j]
   End
  If (Track[j].active==false and misscount >3)
     Then delete Track[j]
   End
 End
Update the buffered background
```

D. *Occlusion Detection and Tracking*

A tracked blob is considered to be occluded if its major region (say 80 %) is covered by foreground and it should continue to be tracked if either it is occluded or its area and centroid is matched with any of the blobs of set P.

An alarm is raised if t_n > threshold. The blob is removed from T if m_n>3. This idea is similar to the method used in [7] and [8] for occlusion detection, but instead of keeping track of two different foregrounds, we propose the following modification.

Let us assume that a particular portion of the frame containing the blob which is being tracked (i.e. present in the 'Track' set of blobs) is now occluded. Due to this occlusion, the blob signifying that particular object won't be included in the present set of Track. Mathematically,

$$t_n = t_n + 1 \text{ and } m_n = 0$$

if

$$\frac{R_n}{A_n} > 0.8$$

$$\text{or } \frac{\left\|C_n(i,j) - c_k(i,j)\right\|^2}{A_n} < 0.1 \text{ and } \frac{\left\|A_n - a_k\right\|}{A_n} < 0.1$$

$$\vee 1 \le k \le m$$

else

$$m_n = m_n + 1$$

Following are the possibilities in the new frame of the blob that was being tracked up to previous frame:-

- Object is removed from the location. In this case, the blob area representing the object should contain background pixels.
- There may be a new object at the same location.
- There is a new object which completely or partially occludes the old object.

An exception to the above cases is when a tracked object is removed while being occluded or another object of similar size is placed in camera's line of view. To deal with occlusion we have added the following two steps to the tracking algorithm:

Step 1: Calculate the number of pixels of buffered background which are same as that of current background for that element of set Track which has suddenly stopped being tracked (due to occlusion, or removal from scene). If it's below a threshold value, say 50 percent, and the hitCount is above a threshold value (making sure the blob has been tracked long enough), we label this element of the set Track as occluded.

Step 2: Go to step 2 of the tracking algorithm. A blob labeled occluded remains in the Track set; i.e. its hitCount is incremented and missCount is reinitialized.

Rest of the tracking algorithm remains same.

E. *Alarm and Display*

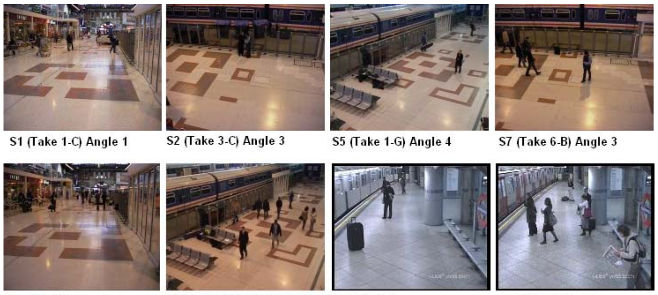
We use the Raise-alarm flag from previous units and highlight that part of the video for which the alarm has been raised. We also display the binary-image (without background) video so that the operator can fine tune the value of D for shadow and reflection subtraction.

IV. Results and Analysis

All algorithms described above are applied on standard benchmark datasets for obtaining experimental results.

A. *Datasets*

The experiments were computed on Intel Core 2 Duo processor with 1 GB RAM. Every video was scaled down to QVGA resolution (320x240) and 10 fps frame rate before further analysis. Five different situations involving different crowd densities and various sizes of objects were selected from the PETS 2006 dataset and two different situations were analyzed from AVSS 2007 i-LIDS Abandoned Baggage Training dataset, one with a low crowd density while the other had a medium crowd density. Both datasets involved surveillance feed from metro stations, snapshots of which are given below.



S1 (Take 1-C) Angle 1

S2 (Take 3-C) Angle 3

S5 (Take 1-G) Angle 4

S7 (Take 6-B) Angle 3

S6 (Take 3-H) Angle 1

S7 (Take 6-B) Angle 4

AVSS AB Easy

AVSS AB Medium

Figure 4: Snapshot of PETS & AVSS datasets

The PETS dataset was recorded at a metro station and each scene involved a person with a bag who loiters for a while before leaving the bag unattended. The details of videos which were analyzed are as follows:

- Dataset S1 (Take 1-C) : 1 person, 1 luggage item, difficulty 1/5
- Dataset S2 (Take 3-C) : 2 people, 1 luggage item, difficulty 3/5
- Dataset S5 (Take 1-G) : 1 person, 1 luggage item, difficulty 2/5
- Dataset S6 (Take 3-H) : 2 people, 1 luggage item, difficulty 3/5

Although ground truth for all the videos was available, it was not utilized as our model dynamically updates the background and is therefore not scene specific. Each scene of PETS dataset was recorded from four different angles, resulting in 20 different videos from PETS dataset and two videos from AVSS dataset.

B. *Setup*

The minimum time for which the object remains stationary and abandoned, before the alarm is raised, can be varied and should be ideally 2 to 3 minutes, but since the total time for which a video in our dataset runs is less than 3 minutes, we have kept the minimum time as 30 seconds or 300 frames (30s @10 fps). Once the object starts getting tracked, anything which occludes it is taken care by our algorithm. Hence the tracking time remains constant at around 30 seconds, in any situation for any stationary object before the alarm is raised.

Time taken for the alarm to be stopped after the object is removed from the site, depends on how soon the impression of the object on the current background disappears and hence the difference between the intensities of buffered background and current background pixels becomes insignificant. This time depends on current-background-update-rate and how different the object is from the things which are behind it in the scene. Since this time depends on the object and scene textures, it varies from five to ten seconds.

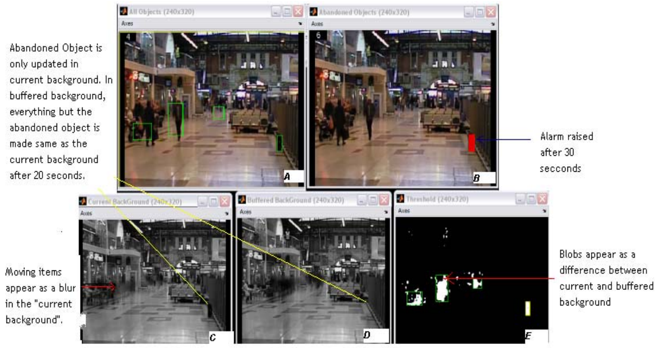

All Objects (240x320)
Abandoned Objects (240x320)
Abandoned Object is only updated in current background. In buffered background, everything but the abandoned object is made same as the current background after 20 seconds.
Alarm raised after 30 secconds
Current BackGround (240x320)
Buffered BackGround (240x320)
Threshold (240x320)
Moving items appear as a blur in the "current background".
Blobs appear as a difference between current and buffered background
A
B
C
D
E

Figure 5: Abandoned object detection process

False alarm is raised or the object is not detected in only three cases. Each case and the reason for failure are given as follows:

- Object gets camouflaged by the background and fails detection (n_o): 2 videos
- Object is correctly detected but a very still person also gets detected (n_p): 1 video
- Object is correctly detected, no person is detected but an unwanted blob is also incorrectly detected as an abandoned object (n_b): 3 videos

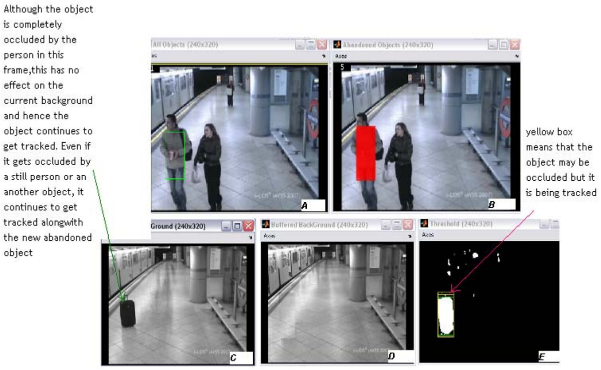

Although the object is completely occluded by the person in this frame,this has no effect on the current background and hence the object continues to get tracked. Even if it gets occluded by a still person or an another object, it continues to get tracked alongwith the new abandoned object
All Objects (240x320)
Abandoned Objects (240x320)
yellow box means that the object may be occluded but it is being tracked
Ground (240x320)
Buffered BackGround (240x320)
Threshold (240x320)
A
B
C
D
E

Figure 6: Occluded abandoned object detection

Although, the system performance can be measured via common metrics such as the ROC curves, we have defined the success rate of our algorithm using a score which is equivalent to the ratio of the total number of videos analyzed to the number of successfully analyzed videos. Mathematically,

$$S = \frac{n}{n - \left(n_o + n_p \times 0.5 + n_b \times 0.25\right)}$$

We define the parameter for measuring successful videos in the above fashion because our algorithm completely fails if no object is detected, it partially (50% failure) succeeds if object is detected but along with it a still person is also detected, and it fails very little (25% failure) if the object is correctly detected, no person is detected but an unwanted blob is incorrectly detected as an abandoned object.

The overall results are illustrated in Table 1. A total of 22 videos were analyzed from both the PETS and AVSS datasets. There were two complete failures as the system was unable to detect the abandoned objects in the video frames. There was one partial failure in which a still person was also detected as an abandoned object. In addition, there were three videos in which unwanted blobs were sometimes classified as abandoned objects. By applying the formula for the performance score as explained above, we achieved a success rate of 85.2%. The results are comparable to the methods presented in the literature and are significant because of lesser computation and faster processing.

C. *Discussion of Results*

Based on the results and analysis, we can conclude that low to medium density crowd has no effect on processing speed or accuracy of the model. In a high density scenario, there is a possibility that the object is prone to be hidden from camera view for most of the time or in other words it is camouflaged by the background leading to a failure in detection. Another achievement of this model was that difference in lighting conditions had almost negligible effect on the operating performance. This can be attributed to the use of Dynamic Background technique. The system will thus work perfectly in an open environment (under sunlight) too. Additionally, shadow effects and reflection of light from bright objects do not pose any problems.

The algorithm works in real time at QVGA resolution and 10 fps frame rate, and has a high success rate of 85%. Even decreasing the frame rate to as low as 3 or 4 fps has insignificant effects on the accuracy of the model. Processing speed is inversely proportional to the square of resolution of the video for a given aspect ratio and also inversely proportional to the frame rate of the video.

The model can detect any number of abandoned objects in a given video sequence. Although speed is compromised with an increase in the number of objects to be detected but such cases are rare to encounter. Some noticeable limitations of the model are that a completely immovable person gets mistaken for an abandoned object.

Also, the object must be in clear view of the camera for at least five seconds, otherwise it gets merged into the background.

V. CONCLUSIONS

This paper presented an abandoned object detection system based on a dual background segmentation scheme. The background segmentation is adaptive in nature and based on the Approximate Median Model. It consists of two types of reference backgrounds, Current and Buffered background, each with a different time interval. Blob analysis is done on the segmented background and a dynamic tracking algorithm is devised for tracking the blobs even under occlusion. Detection results show that the system is robust to variations in lighting conditions and the number of people in the scene. In addition, the system is simple and computationally less intensive as it avoids the use of expensive filters while achieving better detection results.

REFERENCES

[1] C. Sears and Z. Pylyshyn, “Multiple Object Tracking and Attentional Processing”, Canadian Journal of Experimental Psychology, vol. 54, 2000, pp. 1-14.

[2] P. Cavanaugh and G. Alvarez, “Tracking Multiple Targets with Multifocal Attention”, Trends in Cognitive Sciences, vol. 9(7), 2005, pp. 349-354.

[3] M. Bhargava, C-C. Chen, M.S. Ryoo, and J.K. Aggarwal, “Detection of Abandoned Objects in Crowded Environments”, in Proceedings of IEEE Conference on Advanced Video and Signal Based Surveillance, 2007, pp. 271 - 276

[4] G.L. Foresti, L. Marcenaro, and C.S. Regazzoni, “Automatic Detection and Indexing of Video-Event Shots for Surveillance Applications”, vol. 4, 2002, pp. 459 - 471

[5] R. Mathew, Z. Yu and J. Zhang, “Detecting New Stable Objects in Surveillance Video” in Proceedings of the IEEE 7th Workshop on Multimedia Signal Processing, 2005, pp. 1 – 4.

[6] N.J.B. McFarlane and C.P. Schofield, “Segmentation and tracking of piglets in images”, Machine Vision and Applications, vol. 8, 1995, pp. 187-193.

[7] N. Bird, S. Atev, N. Caramelli, R. Martin, O. Masoud and N. Papanikolopoulos, “Real Time, Online Detection of Abandoned Objects in Public Areas”, in Proceedings of IEEE International Conference on Robotics and Automation, 2006, pp. 3775 – 3780.

[8] F. Porikli, Y. Ivanov, and T. Haga, “Robust Abandoned Object Detection Using Dual Foregrounds”, Eurasip Journal on Advances in Signal Processing, vol. 2008, 2008.

[9] J.O. Aguilar, “Omnidirectional Vision Tracking with Particle Filter”, in Proceedings of 18th International Conference on Pattern Recognition, vol. 3, 2006, pp. 1115 – 1118.

[10] I. Haritaoglu, D. Harwood, L. S. David, “W4: Real-time surveillance of people and their activities”, IEEE Trans. Pattern Anal. Mach. Intelligence, vol. 22 (8), 2000, pp. 809–830.

[11] F. Chang, C-J. Chen, and C-J. Lu, “A linear time Component-Labeling Algorithm using Contour Tracing Technique”, Computer Vision and Image Understanding, vol. 93, 2004, pp. 206-220.

[12] N. Friedman and S. Russell, “Image segmentation in video sequences: a probabilistic approach”, in Proceedings of 13th Annual Conference on Uncertainty in Artificial Intelligence, 1997, pp. 175-181.

[13] C. Stauffer, and W. Grimson, “Adaptive background models for real-time tracking”, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 246-252, 1999.

[14] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland, “Pfinder:Real-time tracking of the human body”, IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 7, pp. 780–785, Jul. 1997.

TABLE I. SUMMARY OF RESULTS

Video Nomenclature	**Time Taken for Object to be detected after being left abandoned** (in seconds)				**Time Taken for alarm to be removed after the object is removed from the site** (in seconds)			
	Angle1	Angle2	Angle3	Angle4	Angle1	Angle2	Angle3	Angle4
S1 (Take 1-C)	30	30	30	30	6	5	6	6
S2 (Take 3-C)	31	31	31	31	7	7	7	7
S5 (Take 1-G)	Failed	Failed	30	30	Failed	Failed	9	10
S6 (Take 3-H)	29	29	29	29	9	9	9	9
S7 (Take 6-B)	30	30	30	30	6	5	5	5
AVSS (low density)	28	_	_	_	6	_	_	_
AVSS (medium density)	30	_	_	_	7	_	_	_