Abandoned Object Detection Using Multi-Layer Motion Detection

Simon Denman, Sridha Sridharan, Vinod Chandran
Image and Video Research Laboratory
Queensland University of Technology
GPO Box 2434, Brisbane 4001, Australia
{s.denman, s.sridharan, v.chandran}@qut.edu.au

Abstract

Abandoned object detection (AOD) systems are required to run in high traffic situations, with high levels of occlusion. Systems rely on background segmentation techniques to locate abandoned objects, by detecting areas of motion that have stopped. This is often achieved by using a medium term motion detection routine to detect long term changes in the background. When AOD systems are integrated into person tracking system, this often results in two separate motion detectors being used to handle the different requirements. We propose a motion detection system that is capable of detecting medium term motion as well as regular motion. Multiple layers of medium term (static) motion can be detected and segmented. We demonstrate the performance of this motion detection system and as part of an abandoned object detection system.

1 Introduction

Abandoned object detection (AOD) is the task of locating objects that are left in a scene. Often these objects are quite small (compared to the people at least) and are frequently occluded by other people or vehicles moving about the scene. AOD systems are typically derived from foreground segmentation algorithms. Medium term motion information (pixels that are not part of the background, but are not moving) is used to find regions containing abandoned objects. Relying on motion detection leaves the systems vulnerable to problems caused by lighting fluctuations, shadows and varied contract levels across the scene.

Systems such as those proposed by Sacchi et al [12] and Stringa et al [14] use long term change detection to locate pixels of interest. Pixels are examined for a consistent change in their gray level over several frames, to filter out changes caused by noise. Resultant pixels are filtered, grouped into objects and classified as either an abandoned object, person, lighting effects or structural changes (i.e. moving of a chair) by analysing the bounding box over time. Foresti et al [6] proposed using a long-term change detection algorithm. This is able to absorb slow environmental changes such as gradual lighting changes, but is still sensitive to objects being added to the scene.

Martinez-del-Rincon et al [11] proposed a system that used a double background subtraction method capable of detecting short term abandoned objects [8]. Long term (background only) and short term (background with recently stopped objects) models are combined with the current frame to locate static regions. These regions are fed into an accumulator, where once a fixed time is reached the object is classified as static. For the object to be classified as abandoned luggage, size and shape requirements must be met.

Recently, many AOD systems have been implemented as a sub-system of an object tracking system. This allows motion that is caused by moving objects to be ruled out when performing AOD, and allows the person responsible for the abandoned object (the owner) to be tracked and (possibly) identified. Spengler et al [13] proposed a person tracker and a blob based detection system to locate abandoned objects. After person tracking is performed, remaining unexplained foreground regions are extracted provided they meet size constraints. These candidates (described by their centroid, bounding box and colour model) are observed for a short period of time (1-5 seconds) to filter out spurious objects. If, after the observation period there is a sufficiently high probability that the detected object represents an abandoned object, an alarm is raised. Gule et al [7] combined a moving object detector and a stationary object detector (both based on foreground segmentation results) to locate abandoned objects and their owners. The moving object detector analyses tracked objects for splits to try and identify the drop-off

events, and the resultant objects are matched against those detected by the stationary object detector.

Occlusions can be a major problem for abandoned object detection systems, as they are often required to be able to function in busy environments such as transport hubs where the objects may be frequently occluded. To help maintain the objects through occlusions, Li et al [10] proposed a system that builds templates for stationary objects to allow them to be tracked through complex occlusions. Other approaches such as [2, 9] have been designed to work in a multi-camera environment, which can aid tracking and reduce the effect of occlusions. Auvinet et al [2] proposed a system which merged the results of motion detection into the ground plane of a four camera network. This allows all motion in the scene to be viewed from an overhead perspective, and helps to overcome occlusions. An abandoned object is detected when a fork in the tracking occurs (i.e. one object splits into two), and one of the resulting objects is immobile. Krahnstoever et al [9] uses detectors based on foreground segmentation results to detect and classify objects such as people and luggage.

We propose an abandoned object detection system that utilises a mutli-layer motion detection system, thus eliminating the need for multiple motion detectors. The proposed algorithm is able to deal with lighting changes, and uses a variable threshold to cope with varied contrast levels across the scene. The use of a multi-layer motion detection algorithm also allows occlusions to be partially handled via the foreground segmentation. We demonstrate this motion detection algorithm and describe an abandoned object detection system that utilises this system, and demonstrate it using the PETS 2006 database [1].

2 Multi-Layer Motion Detection

2.1 Existing Technique

An efficient method of foreground segmentation that is robust and adapts to lighting changes was proposed by Butler [3] based on modeling of pixel attributes in multi-modal distributions and pixel clustering. The technique was extended in [4] to incorporate optical flow and improve performance. In this work, a one-frame history of each pixel was stored in the form of an index of the matching cluster for each pixel. The method is further extended into a multi-layer framework here using such motion information.

Let $f(x, y, t)$ be a frame sequence, where x, y is in $[0, N-1]$ and t is in $[0, T]$. Let $P(x, y, t')$ be a pixel in the frame at time t'. Pixels are tracked with their motion and colour history over time interval δt, and have data stored in a set of K clusters, $C(x, y, t, 1..K) = (y_1, y_2, Cb, Cr, w)$, which represent a multi-modal PDF. Input images are in Y'CbCr 4:2:2 format. Pixels are paired to create a cluster which consists of two luminance values (y_1 and y_2), a blue chrominance value (Cb), and red chrominance value (Cr) to describe the colour; and a weight, w. Clusters are ordered from highest to lowest weight; and the current matching cluster, $C(x, y, t, m)$ (where m is the index of the matching cluster in the range $1..K$), for each pixel is stored, giving an approximation of the image.

For each (x, y, t) the algorithm makes a decision assigning it to one of the sets (background, or a motion layer) by matching $P(x, y, t)$ to $C(x, y, t, k)$, where k is an index in the range 1 to K. Clusters are matched to incoming pixels by finding the highest weighted cluster which satisfies,

$$|P(y_1) - C(k)(y_1)| + |P(y_2) - C(k)(y_2)| < LumThr, \quad (1)$$

$$|P(Cb) - C(k)(Cb)| + |P(Cr) - C(k)(Cr)| < ChrThr, \quad (2)$$

where $P = P(x, y, t)$ and $C(k) = C(x, y, t, k)$. The centroid of the matching cluster is adjusted to reflect the current pixel colour, and the weights of all clusters in the pixels group are adjusted to reflect the new state,

$$w_k = w_k + \frac{1}{L}(M_k - w_k), \quad (3)$$

where w_k is the weight of the being adjusted; L is the inverse of the traditional learning rate, α; and M_k is 1 for the matching cluster and 0 for all others. If there is no match, then the lowest weighted cluster is replaced with a new cluster representing the incoming pixels. Clusters are gradually adjusted, allowing the system to adapt to changes in the background.

Based on the accumulated pixel information, the frame can be classified into foreground,

$$fgnd = \forall(x, y, t)\ where \sum_{i=0}^{m} C(x, y, t, i)(w) < T(x, y, t), \quad (4)$$

where $T(x, y, t)$ is the foreground/background threshold; and background. The foreground can be further split into moving and temporarily static objects. We define a static region as an area of motion that has entered the scene and stopped moving, and an active region as an area of motion that is currently moving. The separation of these regions is explained in Section 2.2.

2.2 Static Layers

To discriminate between active and static foreground, we need to compare against the last cluster at a given pixel, and any static foreground objects that are present there.

When $C(x,y,t,m) = C(x,y,t-1,m)$, $P(x,y,t)$ has a static layer, $S(z)$, initialised, where z is the depth of the layer. Each layer has a counter, c, and a colour, (y_1, y_2, Cb, Cr) associated with it. For subsequent frames where $C(x,y,t,m) = C(x,y,t-1,m)$, $P(x,y,t).S(z).c$ is incremented, otherwise it is decremented. Static pixels can be defined as,

$$\forall (x,y,t) \in fgnd\, where \quad P(x,y,t).S(z).c >= \delta t. \tag{5}$$

Static pixels can be further organised into layers depending on when the pixel appears. Layers can be built one on top of the other, as new objects appear and come to a stop atop an existing static layer. Layers remain until the observed cluster is matched to either a lower layer, or the background.

The potential number of static layers depends on the number of clusters at the pixel. We propose varying the number of clusters at a pixel to improve system efficiency while still allowing areas that are complex and have many modes to be modeled effectively. For simple parts of the background (i.e. sky) there will be a very limited number of background modes possible, and therefore there will not be need for the same number of modes in higher volume areas. The number of clusters at a given pixel is evaluated whenever a new mode is detected. At this time, any clusters that have a weight less than half the initial cluster weight, and are not a static cluster, are removed and a new cluster is created to represent the new mode.

The algorithm for detecting and updating the static layers for a single pixel is outlined in Figure 1.

Each static layer is monitored by a counter which is updated each time step, and used to determine the state of the layer (i.e. static, to be removed). Counters are incremented when the layer is detected, and decremented only when a lower level static layer (or background) is detected. When a higher level static layer (or active layer) is detected counters are unchanged as the static layer may be hidden below. Counters are decremented gradually to provide error tolerance for incorrect cluster matching, or noise. The decrement rate depends on the scene, with more challenging scenes requiring a slower decrement rate due to the increased chance of an erroneous cluster match. Layers are removed when the counter reaches zero, and counters are capped to guarantee that a layer can be removed in a set number of frames.

The algorithm has some limitations in that it can only detect overlaps when at least one of the overlapping objects is static. It is also not possible to determine when a lower level static object leaves while higher level static objects remains, or when a lower level object moves in behind a higher level object, due to the relevant pixels being obscured.

2.3 Feedback

It is important to allow changes to occur in the background model as the scene varies, but we must also prevent foreground objects of interest being incorporated into the background. As it is not practical for the motion detector to make these decisions, we propose a method where by an external process can make these decisions.

The inverse of the weight adjustment algorithm can be used to prevent the object from being incorporated into the background model, by effectively stopping all weight updates so that objects of interest remain in the foreground,

$$w_k = \frac{(Lw_k - M_k)}{L-1}. \tag{6}$$

where w_k is the weight of the cluster being adjusted; L is the inverse of the learning rate (lower values will result in background changes being incorporated faster); and M_k is 1 for the matching cluster and 0 for all others.

2.4 Lighting Compensation

In surveillance situations, lighting levels can change rapidly resulting in large amounts of erroneous motion. To prevent this we propose incorporating simple adjustment to the luminance threshold to compensate for lighting changes.

Lighting changes, such as those caused by the sun moving behind clouds, can be expected to cause uniform changes across all (or at least large parts of) a scene. For each frame, we calculate the weighted average of luminance changes,

$$Lum_{offset}(t) = \frac{\sum Lum_{Diff}(x,y,t) \times C(x,y,t,m).w}{\sum C(x,y,t,m).w}. \tag{7}$$

The use of weighted sum allows pixels that are only recently created, and so potentially created partially under the present lighting conditions to be weighted less highly. Provided this value is within a percentage threshold of the previous luminance offset, it is

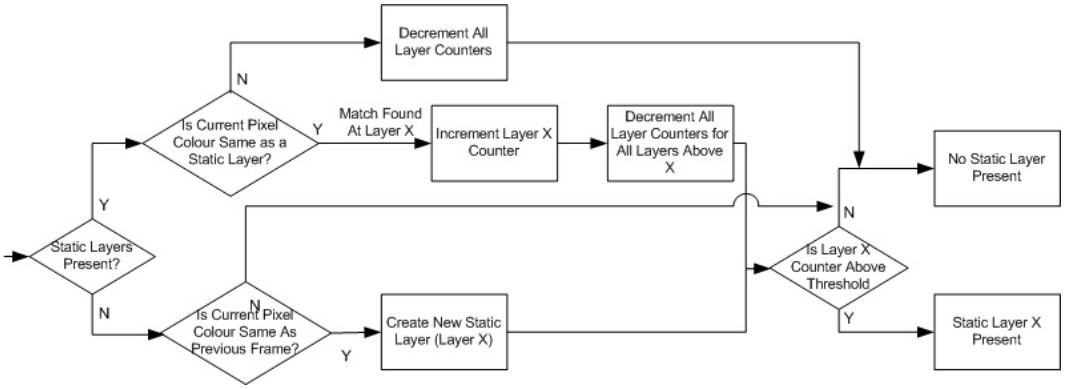


Figure 1. Static Layer Matching Flowchart - If the pixel already has static layers, we compare against these. If there are no layers, or no matches to existing layers, we check to see if there is possibly a new static layer forming (last two frames have the same colour at the pixel).

accepted and used for the next frame,

$$\alpha <= \frac{Lum_{offset}(t)}{Lum_{offset}(t-1)} <= \frac{1}{\alpha}, \tag{8}$$

where α is the change threshold for the luminance offset and is in the range [0..1]. If the change in the luminance threshold is outside of this limit, it indicates a very rapid lighting change has occurred, or a large object has entered the area. In this situation, the weighted standard deviation of the luminance offset is calculated, and if this is beneath a threshold, the lighting change is accepted. If it is outside the threshold, we do not. The luminance offset is incorporated into the match equation by adding the offset to the existing luminance threshold. This results in the matching equations for incoming luminance pixels to cluster becoming,

$$(-Lum_{Thr} + Lum_{offset}) < (P(y_1) - C(k)(y_1)), \tag{9}$$
$$(P(y_2) - C(k)(y_2)) < (Lum_{Thr} + Lum_{offset}), \tag{10}$$

where P is the pixel and $C(k)$ is the cluster that is being matched.

To improve performance, we apply this process on a region level. The image is broken into a grid and the lighting variation at each grid square is considered separately. This allows different materials and their reflective properties, or regions that cast self shadows to be taken into consideration separately. In situations where colour lighting is present, the same approach could be applied to the chrominance threshold to compensate.

2.5 Shadow Detection

Shadows can result in motion being detected where there is none. As such, it is important to recognise shadows and ensure that they are not recorded as motion. Shadows can be characterised by the fact that they alter the luminance component of the objects colour, but have minimal effect on the chrominance. We add shadow detection to the algorithm by adding additional constraints when matching the incoming pixels to the clusters,

$$0 < (C(k)(y_1) - P(y_1)) + (C(k)(y_2) - P(y_2)) < S_{Thr}, \tag{11}$$
$$|P(Cb) - C(k)(Cb)| + |P(Cr) - C(k)(Cr)| < (Chr_{Thr}/S). \tag{12}$$

If there is a positive difference in the luminance, less than the prescribed shadow threshold, S_{Thr}, and only a small difference in the chrominance (determined by dividing the chrominance threshold, Chr_{Thr}, by an integer S) we have a shadow and motion is not detected at P. Shadows need to be handled differently elsewhere in the system. When adjusting the lighting model, and the variable threshold, we disregard pixels that are in shadow as the difference incurred when matching has been significantly altered by the presence of a shadow. As it is not possible to accurately estimate what effect the shadow has had, we disregard it.

2.6 Variable Threshold

A variable threshold is added to the motion detection to aid the system in handling different lighting conditions within the same scene (i.e. shadow, sunlight,

artificial light). Whenever a match is made between an incoming pixel and a cluster, two differences are calculated. In situations where there is no noise (and no motion), these values should be 0. The threshold, $T(x, y, t)$, at $P(x, y, t)$ is based on the standard deviation of these differences, $\sigma(x, y, t)_{Lum}$ and $\sigma(x, y, t)_{Chr}$ over time. We offset the differences by any lighting adjustment to ensure that the lighting changes are not modeled twice.

We assume that the only source of error in a correct match is sensor noise, and that the noise forms a Gaussian distribution with a mean of 0. The standard deviations are multiplied by three to obtain the appropriate thresholds. As we have assumed that the noise is a Gaussian distribution, this will ensure 99% of noise is within the thresholds,

$$T_{Lum}(x, y, t) = 3\sqrt{\sigma_{Lum}(x, y, t)}, \quad (13)$$

$$T_{Chr}(x, y, t) = 3\sqrt{\sigma_{Chr}(x, y, t)}. \quad (14)$$

To avoid sudden changes in the threshold caused by a high level of sensor noise, or by other errors such as an incorrect match, we subject it to the same learning rate as the cluster centroids,

$$\sigma_{Lum}(x, y, t) = \frac{L-1}{L}\sigma_{Lum}(x, y, t-1) + \frac{1}{L}\sigma_{Lum}(x, y, t) \quad (15)$$

When the model is first created, luminance and chrominance variances are initialised to $\sigma_{Lum_{init}}$ and $\sigma_{Chr_{init}}$. If the pixel does not match any existing background mode (i.e. a new colour), then these same initial values are used as the variance for that frame. The threshold is bounds checked (upper and lower) to ensure that excessive levels of motion or noise do not result in the threshold becoming too high, and that high levels of inactivity do not result in it becoming too sensitive.

3 Abandoned Object Detection

The abandoned object detector processes on a pixel level to locate abandoned objects. The process builds directly on the results produced by the static layer motion detection. A time stamp and the pixel's colour is recorded for all pixels. At each time step, the pixels are updated. An accumulator image is constructed in the same manner as static timers are used for the static motion image. The static layer number (depth) and colour information are used to determine matching abandoned pixels in the accumulator image. By using the static layer image, we allow multiple abandoned pixels to be present at a given location in the image, allowing overlapping abandoned objects to be detected and segmented correctly (see Figure 2).

Once an abandoned pixel's counter reaches a threshold, it is added to the abandoned objects. The list of objects is searched for an object that is eight-connected to the newly abandoned pixel to add the pixel to. If no such object can be found, a new abandoned object is created. Merging and splitting of objects occurs at the end of each processing loop, to account for newly abandoned pixel joining objects.

3.1 Fusion with Tracking System

The abandoned object detection system is integrated into an existing motion detection based person tracking system [5]. The tracking system detects people using a combination of motion detection, optical flow and colour. A condensation filter is used to track the people (see Figure 3).

After each frame, the motion that has not been assigned to people (i.e. is unaccounted for) is fed to the abandoned object detector. By integrating the abandoned object detection into a tracking system it allows the person that dropped the luggage to be detected and tracked. We assume that the person closest to the abandoned object when it is first detected is the owner.

4 Results

4.1 Motion Detection

Testing was conducted using a 10,000 frame sequence of real world data acquired at a public passenger drop off area. Twenty frames which illustrated various effects such as lighting variation, shadows, temporarily stopped objects and overlapping objects were hand segmented for comparison (it is not practice to hand segment the entire sequence). Performance was measured in terms of false negatives (FN, motion present in ground truth but not detected) and false positives (FP, motion detected but not present in ground truth). The algorithms overall performance was compared to Butler's [3] (see Table 1). Incorrect detection of the motion type results in a FN and a FP being recorded for the appropriate motion types (i.e. active foreground detected when static's expected - FN for static, FP for active; static detected in layer two expected in layer one - FN and FP for static). We measure the performance of the algorithm at classifying active motion, static motion and shadows, to provide an indication of the performance of each component. Shadow detection was measured purely in terms of false positives, as we

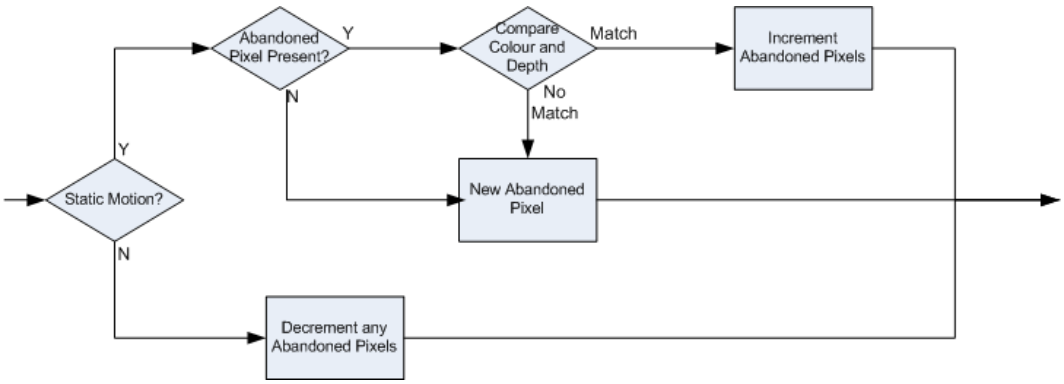


Figure 2. Abandoned Object Detection Process for each Static Pixel - If a static pixel is present, attempt to update any existing AOD pixels that exist at this location. If a match cannot be found, or no AOD pixel exists, create a new one. If there is no static motion, decrement any abandoned pixel counters.

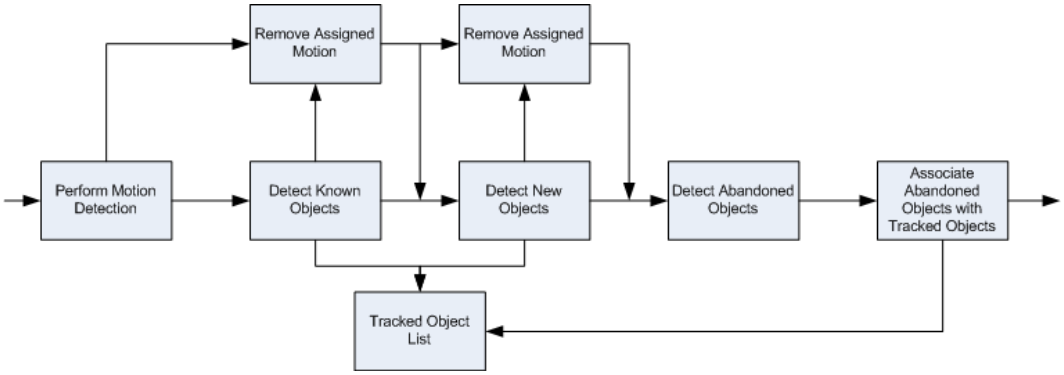


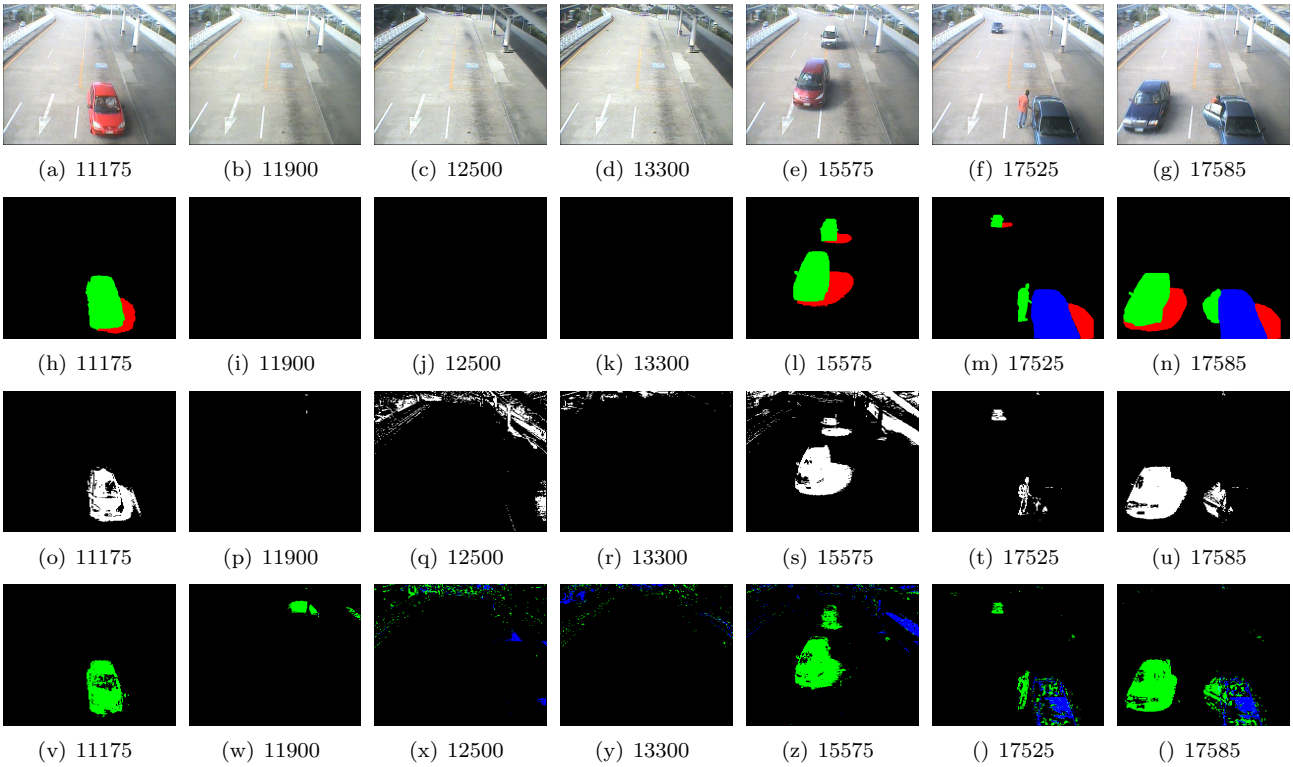
Figure 3. Tracking System - Motion detection is used to detect objects in two stages; detect known (previously detected) objects, followed by detecting any new objects. The remaining motion (which does not belong to people) must belong to any abandoned objects and is used to update the abandoned object detector. The system then attempts to determine which abandoned objects belong to which people.

expect no motion to be detected at a shadow (i.e. errors only occurs when shadows are detected as motion). A simple object detector was applied to the output of our algorithm to locate large foreground objects and apply feedback to the region they occupy. No morphological operations were applied to the output of either system.

As Table 1 and Figure 4 illustrate the system performs well and is able to discern between static and active foreground objects, as well as cope with lighting changes (see frames 12,300 and 13,300 in Figure 4) and shadows. However, the system does struggle to deal with lighting various where the background is widely varied, due to the different textures in the region (i.e. the area around the rails on the left edge of the image,

Table 1. Motion Detection Results

	Our Algorithm		Butler's Algorithm[3]	
	FN	FP	FN	FP
Active Motion	25.40%	2.33%	N/A	N/A
Shadow Motion	N/A	49.34%	N/A	64.95%
Static Motion	55.73%	1.18%	N/A	N/A
Total Motion	38.58%	3.40%	55.49%	8.46%



(a) 11175 (b) 11900 (c) 12500 (d) 13300 (e) 15575 (f) 17525 (g) 17585

(h) 11175 (i) 11900 (j) 12500 (k) 13300 (l) 15575 (m) 17525 (n) 17585

(o) 11175 (p) 11900 (q) 12500 (r) 13300 (s) 15575 (t) 17525 (u) 17585

(v) 11175 (w) 11900 (x) 12500 (y) 13300 (z) 15575 () 17525 () 17585

Figure 4. Multi-layer segmentation results - Top row are the original images; second row is the ground truth; third is the output from Butler [3]; the fourth row is the output from our algorithm; green indicates active motion, blue static motion, red in the ground truth images indicates shadow (which we expect to be detected as no motion in the bottom row). The captions for each image indicate the frame number.

see frame 13,300 and 15,575). The shadow detection can also effect the motion detection when dark objects enter, such as the windscreen and windows of the car in frames 17,400 and 17,525. Despite the limitations of proposed changes however, they result in a significant improvement in performance, clearly reducing the rate of false positives and false negatives when compared to [3].

4.2 Abandoned Object Detection

The abandoned object detection is tested using the PETS 2006 database[1]. Whilst this database is captured using a multi-camera setup, we only consider a single camera situation. Person tracking and abandoned object detection (see Section 3) are performed on the sequences. Results are illustrated in Figures 5 to 8.

As Figures 5, 6 and 7 show, the system is able to track people and abandoned objects, and associate the abandoned objects with their owners. In the case of Figure 6, only the base of the abandoned object is detected at first as the owner is partially occluding the object. As the person moves away, the remainder of the object is detected and flagged. Figure 8 illustrates the systems ability to maintain the location of the abandoned object during occlusions. The abandoned object is occlude several times, yet remains correctly detected. The use of feedback into the motion detection allows the abandoned object to be held out of the background and remain detected.

5 Conclusions and Future Work

We have described a multi-layer motion detection system that can be applied to the problem of abandoned object detection. We have demonstrated the ability of the motion detection system to perform in challenging real-life conditions and as part of an abandoned object detection system. The use of this approach removes the need for multiple motion detectors

(a) 1818 (b) 1868 (c) 1948 (d) 2098 (e) 2113

(f) 1818 (g) 1868 (h) 1948 (i) 2098 (j) 2113

Figure 5. Abandoned Object Detection Results 1 - Top row shows input, bottom row shows the system output. The yellow shaded area indicates an abandoned object, with the shaded yellow person indicating the owner of the object. As the owner moves away from the object, an alarm is raised, indicated by the owner and object being shaded red.

(a) 1820 (b) 1925 (c) 2005 (d) 2035 (e) 2135

(f) 1820 (g) 1925 (h) 2005 (i) 2035 (j) 2135

Figure 6. Abandoned Object Detection Results 2 - Top row shows input, bottom row shows the system output. The yellow shaded area indicates an abandoned object, with the shaded yellow person indicating the owner of the object. As the owner moves away from the object, an alarm is raised, indicated by the owner and object being shaded red. Once the owner is no longer visible, the abandoned object is shaded purple.

to perform abandoned object detection. By detecting mutliple layers of motion, and allowing overlaps when lower layers are occluded, occlusions can be handeled effectivley.

Future work will involve expanding the system to work in a multi-camera network, and expanding the tracking systems to better utilise the mutli-layer motion detection. The motion detection will also be expanded to allow the detection of overlaps in the active layers (i.e. one person walking in front of another), and a variable learning rate.

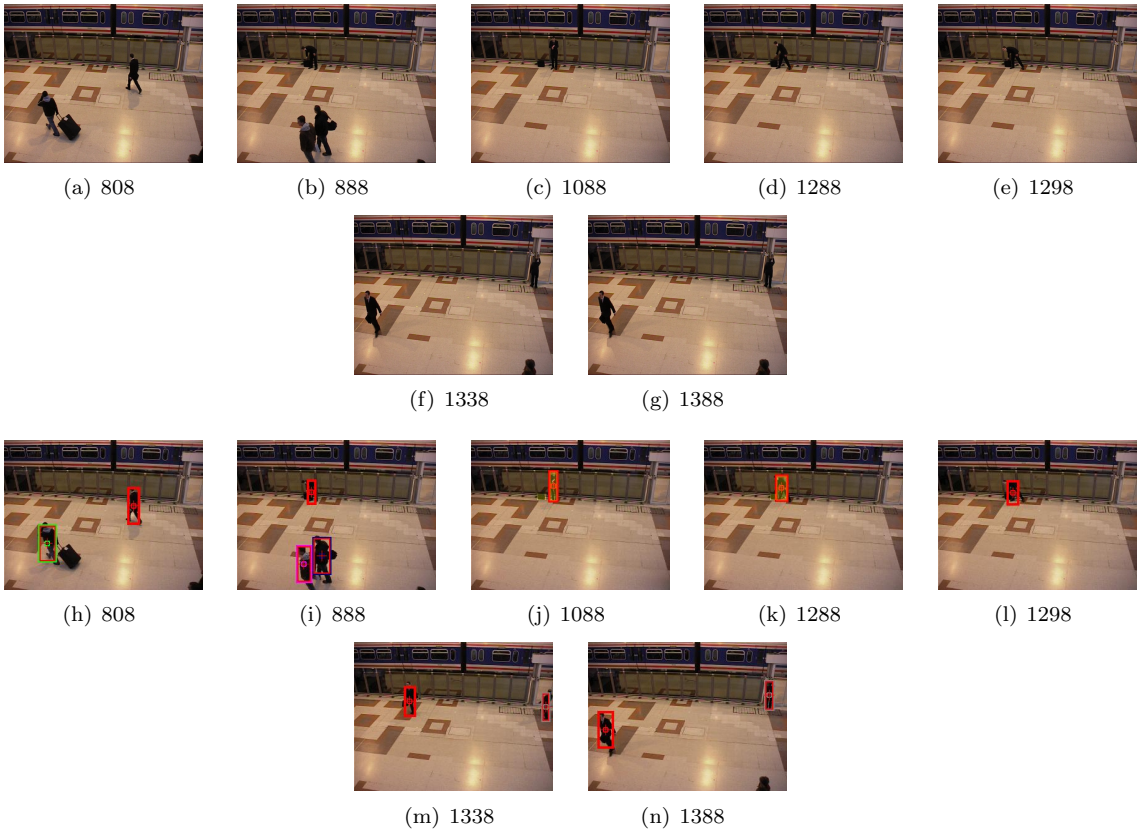
Acknowledgments

This project was supported by the Australian Government Department of the Prime Minister and Cabinet

References

[1] Ninth ieee international workshop on performance evaluation of tracking and surveillance. 2006.

[2] E. Auvinet, E. Grossmann, C. Rougier, M. Dahmane, and J. Meunier. Left-luggage detection using homographies and simple heuristics. In *IEEE International Workshop on PETS, New York, June 18, 2006*, pages 51–58, New York, 2006.



(a) 808 (b) 888 (c) 1088 (d) 1288 (e) 1298

(f) 1338 (g) 1388

(h) 808 (i) 888 (j) 1088 (k) 1288 (l) 1298

(m) 1338 (n) 1388

Figure 7. Abandoned Object Detection Results 3 - Top two rows shows input, bottom two rows show the system output. The yellow shaded area indicates an abandoned object, with the shaded yellow person indicating the owner of the object. The owner places the object beside where they are standing, and an abandoned object is detected. The owner then picks up the object and walks away, and the abandoned object is no longer detected.

[3] D. Butler, S. Sridharan, and V. M. Bove Jr. Real-time adaptive background segmentation. In *ICASSP '03*, 2003.

[4] S. Denman, V. Chandran, and S. Sridharan. Adaptive optical flow for person tracking. In *Digital Image Computing: Techniques and Applications*, Cairns, Australia, 2005.

[5] S. Denman, V. Chandran, and S. Sridharan. A multi-class tracker using a scalable condensation filter. In *Advanced Video and Signal Based Surveillance*, Sydney, 2006.

[6] G. Foresti, L. Marcenaro, and C. Regazzoni. Automatic detection and indexing of video-event shots for surveillance applications. *Multimedia, IEEE Transactions on*, 4(4):459–471, 2002.

[7] S. Guler and M. K. Farrow. Abandoned object detection in crowded places. In *IEEE International Workshop on PETS, New York, June 18, 2006*, pages 99–106, New York, 2006.

[8] E. Herrero, C. Orrite, and J. Senar. Detected motion classification with a double-background and a neighborhood-based difference. *Pattern Recognition Letters*, 24:2079–2092, 2003.

[9] N. Krahnstoever, P. Tu, T. Sebastian, A. Perera, and R. Collins. Multi-view detection and tracking of travelers and luggage in mass transit environments. In *IEEE International Workshop on PETS, New York, June 18, 2006*, pages 67–74, New York, 2006.

[10] L. Li, R. Luo, R. Ma, W. Huang, and K. Leman. Evaluation of an ivs system for abandoned object detection on pets 2006 datasets. In *IEEE International Workshop on PETS, New York, June 18, 2006*, pages 91–98, New York, 2006.

[11] J. Martnez-del Rincn, J. E. Herrero-Jaraba, J. R. Gmez, and C. Orrite-Uruuela. Automatic left luggage detection and tracking using multi-camera ukf. In *IEEE International Workshop on PETS, New York, June 18, 2006*, pages 59–66, New York, 2006.

[12] C. Sacchi and C. Regazzoni. A distributed surveillance system for detection of abandoned objects in unmanned railway environments. *Vehicular Technology, IEEE Transactions on*, 49(5):2013–2026, 2000.

(a) 1003 (b) 1403 (c) 1490 (d) 1570 (e) 1663

(f) 1694 (g) 1883

(h) 1003 (i) 1403 (j) 1490 (k) 1570 (l) 1663

(m) 1694 (n) 1883

Figure 8. Abandoned Object Detection Results 4 - Top two rows shows input, bottom two rows show the system output. The yellow shaded area indicates an abandoned object, with the shaded yellow person indicating the owner of the object. Person tracking results are not shown in this sequence to make the abandoned object detection during occlusion clearer. After the owner drops the luggage, there are several instances of other people occluding the object, in each case the abandoned object remains detected.

[13] M. Spengler and B. Schiele. Automatic detection and tracking of abandoned objects. In *VSPETS*, 2003.

[14] E. Stringa and C. Regazzoni. Real-time video-shot detection for scene surveillance applications. *Image Processing, IEEE Transactions on*, 9(1):69–79, 2000.