Dropped Object Detection in Crowded Scenes

by

Deepti Bhatnagar
B.Tech., Indian Institute of Technology Delhi (2007)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science
in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009

Author .....
Department of Electrical Engineering and Computer Science
May 22, 2009

Certified by .....
Prof. W. Eric L. Grimson
Bernard Gordon Professor of Medical Engineering
Thesis Supervisor

Accepted by .....
Terry Orlando
Chairman, Department Committee on Graduate Students

Dropped Object Detection in Crowded Scenes

by

Deepti Bhatnagar

B.Tech., Indian Institute of Technology Delhi (2007)

Submitted to the Department of Electrical Engineering and Computer Science
on May 22, 2009, in partial fulfillment of the
requirements for the degree of
Master of Science
in Electrical Engineering and Computer Science

Abstract

In the last decade, the topic of automated surveillance has become very important in the computer vision community. Especially important is the protection of critical transportation places and infrastructure like airport and railway stations. As a step in that direction, we consider the problem of detecting abandoned objects in a crowded scene. Assuming that the scene is being captured through a mid-field static camera, our approach consists of segmenting the foreground from the background and then using a change analyzer to detect any objects which meet certain criteria.

In this thesis, we describe a background model and a method of bootstrapping that model in the presence of foreign objects in the foreground. We then use a Markov Random Field formulation to segment the foreground in image frames sampled periodically from the video camera. We use a change analyzer to detect foreground blobs that remain static through the scene and based on certain rules decide if the blob could be a potentially abandoned object.

Thesis Supervisor: Prof. W. Eric L. Grimson
Title: Bernard Gordon Professor of Medical Engineering

Acknowledgments

I would like to thank my advisor Prof. Eric Grimson for his guidance, direction and encouragement. I am grateful to him for his patience, especially during my initial years at grad school. It is an honor and great pleasure to be advised by him.

This work greatly benefited from the ideas and discussions with all the people in CSAIL and APP. I thank them all for the many insightful discussions.

Words cannot express my feelings towards my parents, Mrs. Preeti Bhatnagar and Mr. Pradeep Bhatnagar, for their love, encouragement and sacrifice. This thesis is dedicated to them. I express a feeling of bliss for having a loving sister like Ms. Shikha Bhatnagar. I would also like to thank all my friends and family for their unconditional love, support and encouragement.

Contents

List of Figures

Chapter 1

Introduction

Computer Vision is an active area of research, with the goal of developing techniques that can enable artificial systems to better understand the world in an automated way.

To achieve this goal, we need to extract meaningful information and patterns from the images and video clips given to us. The image data itself can take many forms, such as views from multiple cameras or multi-dimensional or multi-modal data. Moreover, we should be able to leverage that information to aid us in creation of systems that control processes, detect events, organize information, and model interactions.

Some of the interesting problems in which the computer vision community has been traditionally interested include object recognition, scene reconstruction, motion tracking, image restoration, and activity analysis in videos.

Activity analysis is the term given to the broad set of techniques that are used to analyze the contents of a video clip to recover activities of interest. An activity consists of an agent performing a series of actions in a specific temporal order.

The area of activity analysis is extremely important in terms of its scope and potential impact. If we can solve the problem, it would have significant implications on the way we organize and search for information. It can significantly benefit us in ways ranging from social science research (like crowd behavior analysis) to automated monitoring in surveillance videos. We have such large amounts of information

captured in videos that could prove to be extremely helpful to mankind if we can interpret it and represent it in ways that are useful.

In this chapter we will motivate the problem of activity analysis and define it more clearly. We will illustrate the challenges that lie along the way and provide some preliminary examples.

1.1 Activity Analysis

Even though the potential applications of activity analysis are growing very fast, the concept of activity still remains mostly elusive for artificial systems. There are several reasons as to why that could be the case.

First is the problem of definition. Many activities are very subjectively defined. For example, consider a surveillance setting that does theft detection or tries to classify anti-social activities. There is no clear cut definition of what constitutes a theft, or an anti-social activity. The classification is difficult even for humans, let alone artificial systems.

Second is the problem of representation. The concept of an activity comes naturally to humans. But for a computer, it is very hard to represent an activity in any meaningful format. To be able to abstract an activity, a computer needs to be fully aware of the agents, the actions they are performing and in some cases, their intentions as well. It is difficult to represent this information in terms of rules or observable variables. This is because an occurrence in itself does not mean anything, it has to be taken into account with all the contextual information. Even then, the notion of activity is highly subjective and cannot be easily quantified.

Third is the problem of implementation. Even if we define an activity precisely and devise an appropriate representation, it is still very difficult to detect its occurrence in every case.

There are several reasons as to why looking for instances of an activities in a video clip is hard. To understand the actors in a clip, we need to analyze the image frames at every instant of time and discern all the different objects that are present

in it. Furthermore, we need to recover the actions by analyzing how the objects are changing over the course of time and how they are interacting with each other.

Extracting the objects present in a scene, technically called object segmentation, is a computationally hard task and remains an unsolved problem. Although, there have been many advances in the last few years, a universal solution continues to elude us. And unless we make significant progress in object segmentation and classification, developing a generic activity analyzer seems improbable.

Having concluded that a generic activity analyzer is unlikely in the near-future, the question that faces us is whether we can develop techniques that can let us analyze specific types of activities in a predefined setting without fully solving the object segmentation and classification problem.

The prerequisite for such a system would be a technique that does not deal with objects as the basic building block but rather operates at a much lower pixel level and considers an entity as a set of pixels. The problem of abandoned object detection with which we are dealing helps to illustrate this approach to activity analysis.

1.2 Abandoned Object Detection in Crowded Scenes

1.2.1 Motivation

In the last decade the topic of automated surveillance has become very important in the field of activity analysis. Within the realm of automated surveillance, much emphasis is being laid on the protection of transportation sites and infrastructure like airport and railway stations. These places are the lifeline of the economy and thus particularly prone to attacks. It is difficult to monitor these places manually because of two main reasons. First, it would be very labor intensive and thus a very expensive proposition. Second, it is not humanly possible to continuously monitor a scene for an extended period of time, as it requires a lot of concentration. Therefore, as a step in that direction, we need an automated system that can assist the security personnel in their surveillance tasks. Since a common threat to any infrastructure

establishment is through a bomb placed in an abandoned bag, we look at the problem of detecting potentially abandoned objects in the scene.

1.2.2 Other Applications

Even though the problem at which we are looking is very specific in nature, the techniques are fairly general and can be reused in completely different domains to solve other interesting problems.

Examples of areas where these techniques can be employed include retail settings, social science research and human-computer interaction. In retail, it could be revolutionary if the activity of the consumers could be analyzed to better understand consumer behavior and improve product placement in the stores. It can also be used to manage queues more efficiently. Similarly, for social science research, activity analysis could yield a significant amount of information on how species tend to behave under different set of circumstances. For example, it can be used to understand such diverse topics as human behavior in crowds, or the behavior of monkeys or bees under different control conditions. Also, activity analysis could be utilized in human-computer interaction to better understand how humans interact with computer systems and how they could be improved through the use of gesture-based interfaces that rely on certain types of hand-gestures for real-time interaction.

1.2.3 Discussion

There are several ways to interpret the problem of abandoned object detection in crowded scenes. There is still no consensus in the literature on what constitutes an abandoned object. Every major work on this problem has made a different set of assumptions and worked from there.

The set of assumptions that we make are as follows:

- We define an abandoned object as a non-living static entity that is part of the foreground and has been present in the scene for an amount of time greater than a predetermined threshold.

- A foreground object is defined as an entity that is not a part of the background model. It could result from a new object entering the scene or a sudden change in the background.

- There are no vehicles present in the scene. This assumption is realistic for the environments in which we are interested (like airports, train stations etc.).

- The background changes constantly over time.

- The video will be a mid-field domain. The domain of an imaging system is classified based on the resolution at which the objects and its sub-parts are imaged. In a near-field domain, the objects are high resolution and the object sub-parts are clearly visible. In a far-field domain, the object is not clearly visible and no sub-parts are visible. Mid-field lies between the two, the objects as a whole are clearly visible but some of the sub-parts may be just barely visible.

As a preliminary discussion, an obvious approach to solve this problem would be to segment the objects present in every frame and track them over the course of the video clip. Then based on some set of predefined rules, we can classify an object as abandoned and raise a red flag. There are several flaws with this approach. First and foremost, object segmentation is still a hard unsolved problem. Especially in video clips, which tend to have predominantly low resolution images, it is harder still. Adding to the problem is the fact that we expect the scenes to be densely crowded. Second, even though tracking in uncluttered scenes is comparatively reliable, it may simply not be worth the computational effort when we are trying to detect abandoned objects. Therefore, we need a new perspective on the problem and an entirely different approach to solve it. Moreover, we would like to steer clear of hard problems like object segmentation and recognition as far as possible.

1.3 Problem Statement

Given a video sequence captured by a static uncalibrated camera in a mid-field setting, our objective is:

- To develop a reliable system that detects abandoned objects in a crowded scene. As discussed earlier, we define an abandoned object as an entity which is absolutely static in the scene for more than a time period T and the perceived owner of the object is not present within a radius of r.
- To propose a systematic method for segmenting the foreground and background in the scene based on a comprehensive background model.
- To make the background model adaptive, so the system adapts to changes which are persistent and does not have to be restarted periodically.
- To create a system which penalizes false negatives more than false positives.
- To use techniques which are robust in the presence of dense crowds, like at airports, train stations, retail stores etc.

1.4 Outline of our Approach

We use a pixel based approach to solve the problem. Our solution is based on a foreground segmentation and statistical analysis technique. We use a statistical background model based on the variation in brightness distortion and chromaticity distortion of every pixel. We model the foreground segmentation problem as a labeling problem and then use a Markov Random Field setup to tackle it.

The segmented foreground is fed to a change analyzer that tries to detect sets of pixel in the image which are consistently being labeled as foreground and to ascertain if that set has been static for some predetermined amount of time ($\sim T$).

As an example, consider the background scene image shown in Fig. 1-1. For a sample frame shown in Fig. 1-2, we obtain a segmented foreground shown in Fig.

Figure 1-1: Background image for sample dataset S7 from PETS-2007

1-3. Feeding it to the change analyzer we obtain the classification shown in Fig. 1-4 where the red box indicates a potentially abandoned object, a blue box indicates a person or a group of people standing in the scene, and a green box indicates people moving through the scene. Fig. 1-5 depicts more such examples for different data sets.

We update our background model at every time step to constantly adapt it to the changing background conditions, at the same time making sure that any abandoned objects are not immediately incorporated into the background.

We try to do this without any priors over the position, orientation, scale or size of the dropped object, in order to make our approach as generic as possible. To classify the entities as living and non-living, we use a Support Vector Machine (SVM). We determine how much an entity has deviated from its position over a certain time period and use that information for classification. According to our analysis, living entities tend to deviate slightly over a period of time. This is because it is difficult for them

Figure 1-2: Frame from sample dataset S7

to stand absolutely still and even while standing at a place, they show some motion on the fringes of the silhouette. This motion can be captured by cameras in mid-field views. Therefore, we compute the difference of two frames over a predetermined time period and its Discrete Cosine Transform (DCT) over the region of interest. DCT for living entities will have higher coefficients than the non-living ones.

Our approach is based on the principle of minimal evaluation, which says that we should compute a result only if it is required. We will illustrate the use of this principle with the help of two examples. First, when we look for static objects in a scene, we don't track the objects or store their trajectory. This is because all that information is unnecessary if all we are trying to do is single out static objects.

Second, when we classify entities as living or non-living, we do not attempt to determine their exact identity. We are interested in whether the object is living, like a human or a pet, or if the object is non-living like a suitcase or a stroller. We are not interested in object recognition as long as we can make the distinction between

Figure 1-3: Extracted foreground using our technique

living and non-living entities.

The reason we use the principle of minimal evaluation is to prevent the errors in these standard algorithms from accumulating. Computer vision is full of hard problems like segmentation, tracking and object recognition which are not completely solved yet. If we use the object segmentation, classification and tracking approach to abandoned object detection, the error rate in each of those stages will accumulate. Therefore, it is best in general to avoid a technique that involves solving too many hard problems. It is much more useful to build specific components which are more reliable to the problem at hand.

Having said that, abandoned object detection is not an easy problem and remains particularly challenging for crowded scenes. In dense environments, the objects are usually too close together and there is a lot of occlusion. For that reason, it is difficult to segment the objects or to be able to classify them. Also, it is difficult to track the objects reliably as they move through the scene.

Figure 1-4: Abandoned object detection in the sample frame

1.4.1 Our Contributions

We make four main contributions to the problem of abandoned object detection in crowded scenes.

- A novel approach to initializing and updating the background model in the presence of foreground objects.
- A modified energy minimization formulation of the problem of foreground segmentation.
- An efficient pixel based approach to identify whether a foreground blob belongs to an abandoned object.
- A system that maintains minimal state information. We do not need to store large amounts of data in the form of trajectories. This makes the system more scalable by reducing the storage requirements significantly.

Figure 1-5: Results for other sample datasets from PETS-2006

1.5 Organization of the Thesis

The thesis is organized as follows. Chapter 2 gives an overview of the previous work related to the problem. The scene background model and foreground segmentation technique which forms the basis of our system is described in Chapter 3. Chapter 4 presents our approach for detecting abandoned objects in the scene. Experimental results and analysis are presented in Chapter 5. Chapter 6 summarizes the contributions of our work.

Chapter 2

Abandoned Object Detection: A Review

This chapter presents an overview of the past work that forms the basis for most of the research on the topic of abandoned object detection in crowded scenes. It also presents various approaches that have been adopted by researchers in the past as an attempt to solve the problem.

Abandoned object detection belongs broadly to the area of activity monitoring in unconstrained settings. In the recent years, significant amounts of work has been done in the field of activity monitoring. Some of that work has been more specific in nature and dealt with a particular subproblem, while others have been more generic.

In this chapter, we will first look at some of the past work in the broad category of activity monitoring. We will then review the topic of object tracking, which forms the basis for most research in this area. Finally, we will look at some of the work on the specific topic of abandoned object detection in crowded scenes.

2.1 Related Work: Activity Monitoring in Unconstrained Settings

Most of the work in the field of activity monitoring involves automated flagging of unusual activity in videos. It either entails flagging of individual scenes in a clip or flagging of the entire clip as unusual. Most of that work involves learning from classified examples. Many approaches have been adopted for dealing with the task.

The first category of papers does not involve any tracking of individual objects but rather deals with the entire clip as one entity, which is marked as usual or unusual in entirety [34, 33, 32]. They use motion feature vectors to characterize a clip. Some of these techniques use clustering algorithms to divide the clips into clusters and treat the outliers as unusual. They form an interesting approach but are quite restrictive for use in surveillance settings because of their inability to account for individual actions.

The second and most common category of papers involves tracking of individual objects in the video and then using statistical analysis to detect anything unusual. They segment the foreground into distinct objects and track each of those objects over time to extract their trajectories. Then using a suitable clustering algorithm, they cluster the trajectories into different clusters. For any new trajectory, they flag it as unusual if it doesn't belong to any of the clusters. Some of these papers also use appearance information to classify the objects into classes like humans, vehicles etc.

To cluster the trajectories, a metric is required to determine the inter-trajectory distance.Various metrics have been proposed in the literature including Euclidean distance [8], Hausdorff distance [16, 31] and hidden markov model based distances [23]. Various clustering techniques have also been proposed including spectral clustering [19], clustering using graph cuts [27] and agglomerative and divisive clustering [2, 18].

Although, trajectory clustering is a very common and well studied approach, there are three main problems that we would like to point out. First, despite all the recent advances in the area of object tracking, it remains unreliable in densely crowded scenes. Second, the trajectory of an object is an insufficient indicator of what consti-

tutes unusual in a surveillance setting. An activity could be unusual from a security perspective without having an unusual track and thus, a trajectory does not capture all information about the underlying object. Third, clustering remains ineffective and difficult if we do not know the number of clusters a priori. There are some clustering techniques that do not demand to know the number of clusters; however, these non-parametric techniques are computationally very expensive.

The third category of papers involve a probabilistic approach to activity monitoring. [20] uses Bayesian framework to determine isolated tracks in a video. [21] uses Coupled Hidden Markov Models to model interactions between two objects. There has also been a rise of non-parametric Bayesian approaches like Dirichlet process models in recent years. They use a framework like time dependent Dirichlet processes to model the time series data. For example, [7] uses Dirichlet process to solve the problem of data association for multi-target tracking in the presence of an unknown number of targets.

2.2 Related Work: Object Tracking

Since object tracking forms the basis of most research in the area of activity monitoring, it is useful to understand the evolution of tracking.

In the 1990's, some key advances were made by researchers like Stauffer and Grimson [28] and Haritoaglu [12] in the area of far-field tracking. Since then, a fair amount of work has been done to improve reliability of tracking algorithms in challenging conditions, with lighting changes and dynamic backgrounds

Most tracking methods are based on the technique of background subtraction. They formulate a background model and then compare it with the current frame to find regions of dissimilarity. Then based on some criteria they extract foreground blobs from the scene and then cluster or segment the blobs to form distinct objects [22, 26, 29, 30].

In sparse settings, these techniques work quite well to detect the objects present along with their associated tracks, but they are not as useful in densely crowded

scenes. Tracking in the presence of dense crowds still remains unreliable.

2.3 Related Work: Abandoned Object Detection

Abandoned object detection is a fairly recent problem in the domain of activity monitoring. This is because surveillance became an important topic after the increase of terrorism in different parts of the world and visual surveillance became one of the most active research topics in computer vision.

Increasing concern about security has caused tremendous growth in the number of security cameras present at every important location like airport or railway stations. With so many security cameras doing surveillance at each place, it is extremely labor intensive to have individuals monitor them, which might be an expensive proposition. Besides of this, individuals might get tired or distracted easily and might miss something which is dangerous yet inconspicuous. Hence, it is very useful to have automated systems that can assist the security personnel in their surveillance tasks.

There are three different ways that we can classify the existing research on this topic.

- First is the distinction based on how the scene is being monitored using cameras i.e. single vs. multiple camera setting. In a single camera setting, there is only one camera that monitors a given location, while in a multiple-camera setting there could be multiple cameras monitoring the same location. In case there is more than one camera, the technique can be classified based on whether the multiple cameras are calibrated or not.

- The second distinction is based on the level of granularity of the basic constituents-pixel level approach vs. object level approach. A pixel level approach does some computation for every pixel and then aggregates the results, while the object level approach detects the objects present in the scene and then does some computation for them.

- The third distinction is based on the algorithm being used to identify aban-

doned objects. In general, all algorithms for abandoned object detection can be grouped into three categories. First is the set of algorithms based on motion detection- this approach consists of estimating motion for all foreground pixels/objects present in a scene and then determining the abandoned ones based on certain rules. Second is the set of algorithms based on object segmentation and classification- the scene is segmented into objects which are classified into preexisting classes (like people, luggage) and then they are observed over time to identify the static ones. The third set of techniques involves tracking based analytics- the foreground pixels/objects are tracked over time and their tracks analyzed to detect abandoned objects.

We will now discuss some specific papers that attempt to detect abandoned luggage in a crowded setting. [5] deals with a multi-camera setup which detects moving objects using background subtraction. All the information is merged in the ground plane of the public space floor and a heuristic is used to track objects. It detects static luggage and an alarm is triggered when the associated person moves too far. [15] again deals with a multi-camera setup which uses a tracking algorithm based on unscented Kalman filter. It employs a blob object detection system which identifies static objects. It combines the information using homographic transformations and uses a height estimator to distinguish between humans and luggage. [17] tracks objects present in the scene using a trans-dimensional MCMC tracker and analyzes the output in a detection process to identify the static ones. [6] uses a tracking module which tracks blobs and humans using speed, direction and distance between objects. It uses a Bayesian inference technique to classify abandoned luggage. [11] uses a background subtraction based tracker. It looks for pixel regions in the scene which constantly deviate from the background and flags them as potential abandoned objects. [4] uses a multi-camera object tracking framework. It does object detection using adaptive foreground segmentation based on Gaussian mixtures tracking. It performs centrally in the ground plane, by homographic transformation of the single camera detection results. It does fusion analysis of the object's trajectories, detects stationary objects and splits and merges in the trajectories. [9] does foreground ex-

traction using Markov random fields. It uses a blob tracker as an attention mechanism for finding tracks of interest, extends them temporally using the meanshift algorithm. It uses weak human and luggage models, based primarily on characteristics like size. [24] has a pixel-based solution that employs dual foregrounds. It models background as multi-layer, multi-variate Gaussian distribution which is adapted using bayesian update mechanism. It fuses multiple cameras on ground plane.

2.4 Where This Thesis Fits In

The approach that we are adopting can broadly be classified as a single camera, pixel level approach. But the algorithm we use to identify abandoned objects is based on analyzing how each pixel changes over time.

Like most other approaches, we build a background model and perform foreground separation for every video frame. But since optical flow computation, object segmentation and tracking are unreliable in densely crowded scenes, we adopt a pixel-based solution.

We determine a cohesive set of pixels which are consistently being classified as foreground and analyze how it has been changing over time to determine if the pixels could belong to a potentially dropped object. The advantage of our approach is that it works well under fairly crowded scenarios.

Chapter 3

Foreground Extraction Using Energy Minimization

This chapter presents the processing steps that are required to extract the foreground from the background in each frame of the video clip. The chapter is divided into two main parts. The first part describes the different components of the background model and the algorithm to initialize them, while the second part describes the algorithm to extract the foreground from the background using an energy minimization technique. We also describe how the results from the foreground extraction can be used to update the background model.

3.1 Background Model

There has been a significant amount of work on background models in computer vision. Most models for background that have been proposed in the literature answer two broad questions. The first question is whether the background model is considered static or dynamic, and accordingly, is it allowed to change over time. The second deals with whether the background is assumed to be a single image which is unknown and has to be estimated using some technique, or is it assumed to be a statistical distribution over an image.

In cases where the background is assumed to be a single image, every pixel in the

background is assumed to have a unique correct value which has to estimated using some technique like maximum likelihood or energy minimization. Whereas in cases where the background is assumed to have a statistical distribution over an image, every pixel has its own distribution which has to be estimated using some statistical measures.

Before we delve into our background model, we would like to outline two characteristics of the scenes with which we are dealing, so as give more insight into how we answer those questions. First, the image frames that we use to construct our background model are extracted from videos and as a result, their resolution is not very high. Second, the application that we are targeting, i.e. abandoned object detection in crowded scenes, may run for an extended period of time. Over such a long period, there may be significant changes in the lighting conditions and the spatial arrangement of objects which are present in the scene. Therefore, we must ensure that these changes are eventually incorporated in the model.

Based on the above characteristics, we have concluded that the background model in our case should be adaptive. Similarly, a statistical model is more suited to our application, as it is more robust in the presence of noise and captures more information about how each pixel might be distributed.

3.2 Model Description

We model our background as a statistical model proposed by Horprasert et al. in [13] and used by Ahn et al. to extract foreground in [1]. It is based on a color model that separates the brightness component of a pixel from its chromaticity component.

The reason we chose this model is because it captures the property of human vision called *color constancy*, which says that humans tend to assign a constant color to an object even under changing illumination over time or space [14]. Therefore, a color model that separates the brightness from chromaticity component is closer to the model of human vision. Also, to be able to distinguish the foreground from the background pixels we need to differentiate the color of the pixel in the current image

from its expected color in the background. We do this by measuring the brightness and chromatisity distortion for all pixels.

According to the model, the i^{th} pixel in the scene image can be modeled as a 4-tuple $< E_i, s_i, a_i, b_i >$, where E_i is the expected RGB color value, s_i is the standard deviation of color value, a_i is the variation of the brightness distortion, and b_i is the variation of the chromaticity distortion of the i^{th} pixel. The exact mathematical formulation will be described later.

Assuming the brightness distortion and the chromaticity distortion to be normally distributed, we can estimate the likelihood of a pixel belonging to the background using the standard normal distribution.

3.2.1 Model Parameters

More formally, background model estimation consists of estimating the following parameters for each pixel.

- Let $E_i = [\mu_R(i), \mu_G(i), \mu_B(i)]$ be defined as the i^{th} pixel's expected RGB color value in the background scene image, where $\mu_R(i)$, $\mu_G(i)$ and $\mu_B(i) \in [0, 255]$.

- Let $\sigma_i = [\sigma_R(i), \sigma_G(i), \sigma_B(i)]$ be defined as the i^{th} pixel's standard deviation in the RGB color, where $\sigma_R(i)$, $\sigma_G(i)$ and $\sigma_B(i)$ are the standard deviation of the i^{th} pixel's red, green, blue values measured over a series of M observations.

- The brightness distortion (α) is a scalar value that brings the observed color, I_i, close to the expected chromaticity line, E_i. In other words, it is the α that minimizes ϕ where

$$\phi(\alpha_i) = (I_i - \alpha_i E_i)^2 \tag{3.1}$$

- Let a_i represent the variation of the i^{th} pixel's brightness distortion, α_i, as measured in equation 3.1.

- The chromaticity distortion is defined as the orthogonal distance between the observed color, I_i, and the expected chromaticity line, E_i. The chromaticity

distortion is given by

$$CD_i = \|I_i - \alpha_i E_i\| \tag{3.2}$$

- Let b_i represent the variation of the i^{th} pixel's chromaticity distortion, CD_i, as measured in equation 3.2.

3.2.2 Parameter Computation

There are some differences between how the model parameters are computed in our approach as compared to the approach proposed by Horprasert et al. [13].

The main difference is that they assume the training set to consist of images which are purely background images and have no foreground objects in them. Thus, they assign equal weight to all the sample frames when computing the expectation and standard deviation. On the other hand, we assume that each frame can potentially have foreground objects and thus assigned weights that are directly related to the likelihood of a given pixel being a part of the background. Since we do not have any information about the background to begin with, we use another technique to bootstrap the system and compute weights, which will be described in section 3.2.3. For now, let us assume that the weight of the i^{th} pixel in the j^{th} training set image is w_{ji}.

We perform all our computations in the RGB color space, as all our data is in this format. It is possible to devise equivalent models in other color spaces as well. Let $I_{ji} = [I_R(j,i), I_G(j,i), I_B(j,i)]$ denote the i^{th} pixel's RGB color in the j^{th} training image. We can calculate the expected RGB color of the i^{th} pixel, E_i, as follows:

$$E_i = [\mu_R(i), \mu_G(i), \mu_B(i)] = \frac{\sum_{j=1}^{M} w_{ji} I_{ji}}{\sum_{j=1}^{M} w_{ji}} \tag{3.3}$$

Having computed the expected color, we calculate the standard deviation in the RGB color of the i^{th} pixel as follows:

$$\sigma_i = [\sigma_R(i), \sigma_G(i), \sigma_B(i)] = \sqrt{\frac{\sum_{j=1}^{M} w_{ji}(I_{ji} - E_i)^2}{\sum_{j=1}^{M} w_{ji}}} \tag{3.4}$$

The brightness distortion of the i^{th} pixel in the j^{th} image, α_{ji}, is

$$\alpha_{ji} = \underset{\alpha}{\operatorname{argmin}} \left[\left(\frac{I_R(j,i) - \alpha\mu_R(i)}{\sigma_R(i)} \right)^2 + \left(\frac{I_G(j,i) - \alpha\mu_G(i)}{\sigma_G(i)} \right)^2 + \left(\frac{I_B(j,i) - \alpha\mu_B(i)}{\sigma_B(i)} \right)^2 \right]$$
$$= \frac{\left(\frac{I_R(j,i)\mu_R(i)}{\sigma_R^2(i)} + \frac{I_G(j,i)\mu_G(i)}{\sigma_G^2(i)} + \frac{I_B(j,i)\mu_B(i)}{\sigma_B^2(i)} \right)}{\left(\left[\frac{\mu_R(i)}{\sigma_R(i)} \right]^2 + \left[\frac{\mu_G(i)}{\sigma_G(i)} \right]^2 + \left[\frac{\mu_B(i)}{\sigma_B(i)} \right]^2 \right)} \tag{3.5}$$

and the chromaticity distortion of the i^{th} pixel in the j^{th} image, CD_{ji}, is

$$CD_{ji} = \sqrt{\left(\frac{I_R(j,i) - \alpha_{ji}\mu_R(i)}{\sigma_R(i)} \right)^2 \left(\frac{I_G(j,i) - \alpha_{ji}\mu_G(i)}{\sigma_G(i)} \right)^2 \left(\frac{I_B(j,i) - \alpha_{ji}\mu_B(i)}{\sigma_B(i)} \right)^2} \tag{3.6}$$

The variation of the brightness distortion of the i^{th} pixel, a_i is given by

$$a_i = \sqrt{\frac{\sum_{j=1}^{M} w_{ji}(\alpha_{ji} - 1)^2}{\sum_{j=1}^{M} w_{ji}}} \tag{3.7}$$

The variation of the chromaticity distortion of the i^{th} pixel, b_i is given by

$$b_i = \sqrt{\frac{\sum_{j=1}^{M} w_{ji}(CD_{ji})^2}{\sum_{j=1}^{M} w_{ji}}} \tag{3.8}$$

3.2.3 Bootstrapping the Model

To be able to estimate the parameters described in the previous section, we need weights for all pixels in all of the training images. As we mentioned earlier, the reason we need weights is to give more emphasis to pixels that belong to the background rather than the foreground. But the question that faces us is how can we decide which pixels belong to the background unless we have a valid background model. As is clear, this is a cyclic problem. To counter this problem, we need to bootstrap the algorithm somehow. We use optical flow and image continuity principles to get our first estimate of the background, and then improve our estimate iteratively.

Thus, the input for the bootstrapping algorithm consists of a set of training images, I_1 to I_M and the output is a set of weights for each pixel in each of the images, i.e.

w_{ji} for $1 \leq j \leq M$.

The procedure we adopt to compute those weights is to first estimate the background image using an energy minimizing algorithm and then determine the weights based on how different the pixels in the training image are from pixels in the background image and what is the optical flow at that location.

Background Image

Given a set of K images for recovering the background, we assume that every pixel in the resulting composite background image will come from one of these K images and thus, we can assume that every pixel in the composite image takes a label in the set $\{1, 2, ..., K\}$. In other words, if the i^{th} pixel takes a label j, then the composite background image will copy i^{th} pixel from the jth image in the training set. This composite background image is used only for bootstrapping the actual model. Hence, it suffices to have only one background image instead of a mixture of images. Our actual model described in Section 3.2.2 uses this composite background image to compute weights.

We formulate a Markov Random Field (MRF) model for this labeling problem. This is because under the setting that we have described, neighboring pixels should have a higher probability of having the same label. This constraint is captured naturally in a MRF formulation.

Our MRF optimizes the following objective function:

$$E^B(l) = \sum_{(i,j)\in\mathcal{N}} V_{i,j}^B(l_i, l_j) + \sum_{i\in\mathcal{P}} F_i^B(l_i) + \sum_{i\in\mathcal{P}} D_i^B(l_i) \tag{3.9}$$

where l is the field of source labels for the composite background image, $\mathcal{P}$ is the set of pixel sites, $\mathcal{N}$ is the 4-neighborhood graph, $V_{i,j}^B(l_i, l_j)$ encourages neighboring pixels to have the same label, $F_i^B(l_i)$ encourages the label for the i^{th} pixel to be sourced from an image which has small optical flow at that spatial location,and $D_i^B(l_i)$ encourages the pixel to be sourced from an image which has less contrast at that spatial location.

These terms are defined as:

$$V_{i,j}^{B}(l_i, l_j) = c_n \delta(l_i, l_j) \tag{3.10}$$

$$F_i^B(l_i) = \exp\left(\|\phi_i(l_i)\|\right)/c_f \tag{3.11}$$

$$D_i^B(l_i) = \sum_{j \in \mathcal{N}(i)} \exp\left(-\|I_{l_i}(i) - I_{l_i}(j)\|\right)/c_d \tag{3.12}$$

where $\delta(\cdot,\cdot)$ is the Kronecker delta function, $\phi_i(k)$ is the optical flow of the i^{th} pixel in the k^{th} image, and $\|I_k(i) - I_k(j)\|$ is the normed difference between the i^{th} and j^{th} pixels in the k^{th} image. c_n, c_f and c_d are some constants.

We use graph cuts to solve the labeling problem. If the resulting label image after energy minimization is denoted by L and the resulting composite background image is denoted by B, then

$$B(i) = I_{L_i}(i) \tag{3.13}$$

Weights

Given an estimate of the background image B, we compute the weights as a function of the optical flow and the normed difference of the training image from the background image:

$$w_{ji} = g(w_1, w_2) \tag{3.14}$$

where $w_1 = g_1(\|\phi_i(j)\|)$ and $w_2 = g_2(\|I_j(i) - B(i)\|)$. First, lets analyze w_1. We want w_1 to have the following properties:

- w_1 should attain its maxima at $\|\phi_i(j)\| = 0$ and should $\to 0$ as $\|\phi_i(j)\| \to \infty$.
- It should decrease at a rate that increases with the optical flow.

A simple function that satisfies both the criteria is the exponential decay function given by $w_1 = \exp\left(-\|\phi_i(j)\|/c_1\right)$, where $1/c_1$ is the decay constant. For every increase of c_1 in the optical flow, w_1 will further decrease by a factor of $1/e$. In practise, we take c_1 to be around 3 pixels.

Using exactly the same logic for w_2, we assume the following form for w_2: $w_2 = \exp(-\|I_j(i) - B(i)\|/c_2)$. We take c_2 to be around 15, which means that for every 15 units of difference between the training image and the background image intensity, w_2 will decrease by a factor of $1/e$.

We take w_{ji} to be the sum of w_1 and w_2 as follows:

$$w_{ji} = w_1 + w_2 = k_1 \exp(-\|\phi_i(j)\|/c_1) + k_2 \exp(-\|I_j(i) - B(i)\|/c_2) \tag{3.15}$$

where k_1 and k_2 are constants that determine the relative importance of w_1 and w_2. The reason that we don't compute w_{ji} as $(w_1 * w_2)$ is because multiplication is not robust to errors. In case any of the w's is wrongly computed as 0, the entire term would be wrongly computed as 0.

3.3 Foreground Extraction with Graph Cuts

Having estimated the background model parameters as described in section 3.2.2, we move on to foreground extraction based on those parameters. Given an input frame, we model the foreground extraction as a binary labeling problem that labels each pixel in the given frame as either foreground or background. We then use Markov Random Fields (MRF) to solve the binary labeling problem. This is because due to the continuous nature of real-world objects, neighboring pixels should have the same label except on object boundaries.

We formulate an energy minimization framework for the MRF which is implemented using graph cuts. Our data penalty and smoothness penalty terms are based on the brightness and chromaticity distortion distributions estimated earlier. The approach is similar to that used by [1] except that we do not use trimaps and our penalty functions are formulated differently.

The objective function to minimize is similar to that used in the GrabCut technique[25] and makes use of the color likelihood information. Our MRF formulation for fore-

ground extraction attempts to minimize the following function:

$$E^F(f) = \sum_{(i,j)\in\mathcal{N}} V_{i,j}^F(f_i, f_j) + \sum_{i\in\mathcal{P}} D_i^F(f_i) \tag{3.16}$$

where $\mathcal{P}$ is the set of pixel sites, $\mathcal{N}$ is the 4-neighborhood graph, $V_{i,j}^F(l_i, l_j)$ encourages neighboring pixels to have the same label, and $D_i^F(l_i)$ encourages the pixel to be labeled as a background if it has a high likelihood of being sourced from the background color model.

In the resulting label image, we encourage neighboring pixels to have the same label, but at the same time, we want the segmentation boundaries to align with the contours of high image contrast. Therefore, we define the smoothness penalty as:

$$V_{i,j}^F(f_i, f_j) = \exp\left(-\|I(i) - I(j)\|^2/\beta\right)\delta(f_i, f_j) \tag{3.17}$$

where $\delta(\cdot,\cdot)$ is the Kronecker delta function, $\|I(i) - I(j)\|$ is the normed difference between the i^{th} and j^{th} pixels in the given image, and the constant β is chosen (Boykov and Jolly 2001 [3]) by the expectation of $2\|I_p - I_q\|^2$ over all $\{p, q\} \in \mathcal{N}$.

The data term measures how well the pixel i in the given image fits the statistical distribution of the background model. Therefore, we define the data term as:

$$D_i^F(f_i) = -log\ \mathcal{P}(I(i)|f_i) \tag{3.18}$$

After we have segmented the image into foreground and background regions, we compute all the connected components in the foreground and remove components containing less than 100 pixels. This threshold was chosen with empirical observation and can be improved by scaling it with the square of the estimated distance to the camera, if the camera is calibrated. Also, we update the background model parameters using the segmentation results.

Some sample results for foreground extraction are presented in Figures 3-1, 3-2, 3-3 and 3-4. From the figures we can see that the extraction is fairly accurate in

Figure 3-1: Extracted foregrounds for dataset S1 from PETS-2006

Figure 3-2: Extracted foregrounds for dataset S2 from PETS-2006

Figure 3-3: Extracted foregrounds for dataset S5 from PETS-2006

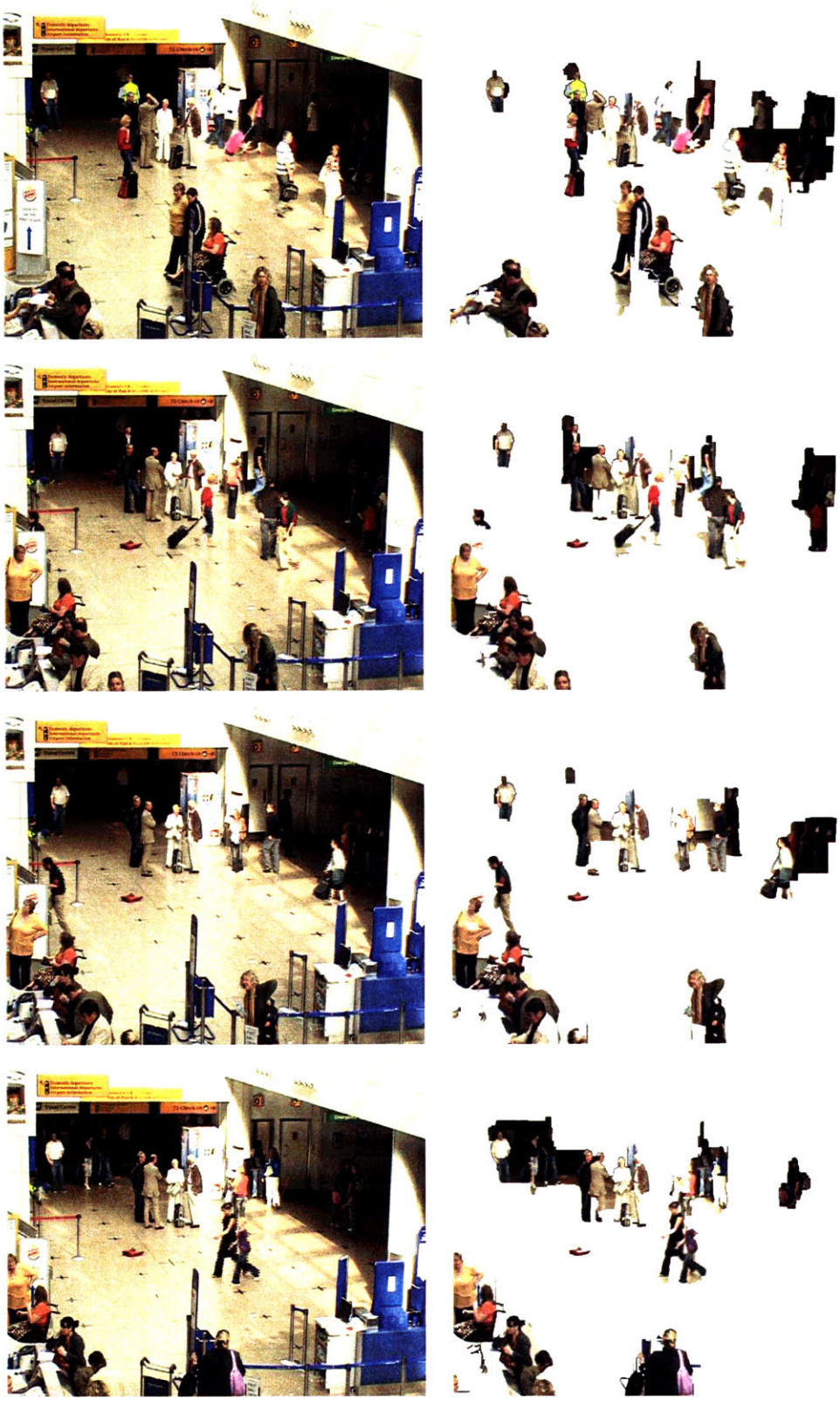

Figure 3-4: Extracted foregrounds for dataset S7 from PETS-2007

most cases. For example, in Figure 3-1, a man standing on a floor upstairs has also been segmented. There are a few cases when the segmentation is problematic. First, foreground extraction is problematic in regions where the background initialization was not proper due to presence of a foreground object for the entire duration. For example, in sequence S2 (Figure 3-2), the garbage bins were part of the background model, and as they moved away, the actual background was segmented as a foreground. Second, in areas of bright illumination, the shadows are captured to be a part of the foreground. Third, in case of an object having multiple states, some of the states are segmented as foreground. For example, in sequence S7 (Figure 3-4), for the door present on the upper right, the background model captured the door in an open state. As a result, whenever the door is open, it is segmented as foreground. Fourth, in regions where the camera view is far-field instead of mid-field, the algorithm does a more coarse job joining multiple people together as part of foreground, instead of a finer segmentation.

Chapter 4

Change Analysis and Abandoned Object Detection

In this chapter, we will describe the techniques we use to detect potentially abandoned objects in a crowded scene. Even though the techniques used may have several applications, we are only dealing with this one.

To be able to detect an object in a video clip and mark it as abandoned, we have to characterize formally as to what we mean by the term "abandoned". If we consider a video clip of a crowded place, like an airport or a train station, there would be several parts of the scene that are occupied by people or objects which are not in motion. But are they abandoned? This is a difficult question to answer and there is no objective answer to that question. For the purpose of our research, we define an abandoned object as a non-living entity that is not a part of the background model, has been present in the scene at the exact same location for quite some time, and the perceived owner of the object is currently not situated close to it. It is entirely possible that such an object is not an abandoned object, but even then, all objects meeting these criteria might be worthy of closer inspection.

For this characterization to be of any practical significance, we would have to make these criteria more concrete. So, we would define an object as "abandoned" if it has been static at a particular location for more that T seconds and the perceived owner is not present in the radius of r pixels. These values are inputs to the system

and can be easily modified as per requirement.

The problem of detecting potentially abandoned objects has been decomposed into two main steps. First, we determine those regions of the image which are consistently labeled as foreground for a period of time exceeding T seconds. Second, we classify the region as either a busy part of the scene, a static living entity or a static non-living entity.

There are several points we take into account with respect to the design of the algorithm. First, we have several redundancies in place to minimize the cost of errors. In a crowded setting like an airport or a train station, errors are very common and thus, the cost of errors should not be very high and error recovery should be easy. Second, we would want to err on the side of conservatism. We don't want any abandoned objects to be missed out and thus, we would prefer false positives over false negatives. Depending on the context, these design considerations could be different in other applications and will guide the choice of parameters.

4.1 Potential Static Blob Identification

We keep a moving average statistic for every pixel that keeps track of how often a pixel is being classified as a foreground. Assuming a label of 1 for foreground and 0 for background, we have:

$$MA_i[t] = \lambda Label_i[t] + (1 - \lambda) MA_i[t-1] \tag{4.1}$$

where $MA_i[t]$ is the moving average at time t at pixel location i, $Label_i[t]$ is the label assigned to the i^{th} pixel by the foreground extraction algorithm, and λ is a constant $\in [0, 1]$ that determines the importance of the current label with respect to the history. It is easy to see that $MA_i[t]$ will always lie within $[0, 1]$. Figure 4-1 shows the moving average snapshot for datasets at some instant of time.

After updating MA with the latest labeling, we check to see if there is some region in the image that has high values of MA which are above a certain threshold. But

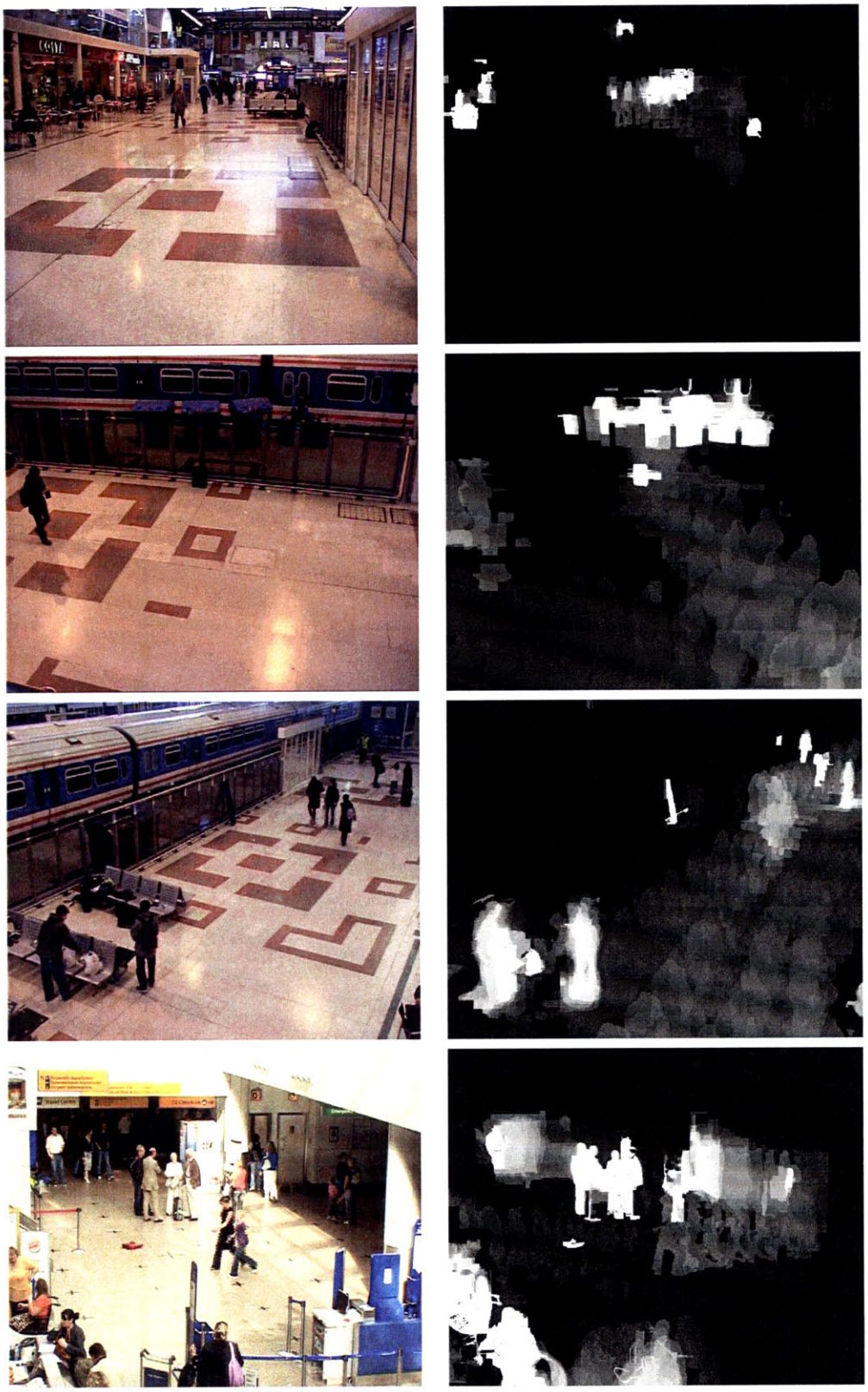

Figure 4-1: Moving average image, MA, in sample datasets

before that, we have to answer the question of how high this threshold should be? To answer this question, consider a pixel that had a MA value of 0 to begin with. After being labeled as foreground continuously for K image frames, its MA value will change to $\lambda \sum_{k=1}^{K}(1-\lambda)^{(k-1)}$. If we want to detect an object that has been lying static in the scene for more that T seconds, and our rate of sampling frames is S frames per second, then we should choose a threshold such that

$$\begin{aligned} threshold &\simeq \lambda \sum_{k=1}^{ST}(1-\lambda)^{(k-1)} \\ &\simeq 1-(1-\lambda)^{ST} \end{aligned} \tag{4.2}$$

We choose λ as 0.05, T as 15 seconds, and calculate the threshold based on the value of S accordingly. We mark all the pixels in the image which have a MA value greater than the threshold. Having marked all the pixels with this property, we use a connected components approach to divide them into distinct sets and discard the sets which have less than 100 pixels. This threshold was chosen with empirical observation and can be improved by scaling it with the square of the estimated distance to the camera, if the camera is calibrated.

At this stage, we are faced with the question of whether we want to segment/merge the blobs to identify distinct objects. Based on our analysis, we have realized that object segmentation is very unreliable in such crowded scenes and thus, it is better to deal at the pixel level than at the object level.

Therefore, having obtained sets of pixels that could belong to abandoned objects, we analyze each of these pixel sets in greater detail.

4.2 Region Classification for Abandoned Object Detection

We classify each pixel set into one of three classes:

- Class I: The first class consists of those regions in the image where the pixels

are consistently being classified as foreground, but the foreground object to which they belong is not the same across time. This is especially common in those areas of the scene which see a lot of traffic and a lot of people move through them, like an escalator or a ticket counter. So, even though the pixels have their MA value above the threshold, they are not originating from an abandoned object, but rather a lot of objects moving continuously through the scene.

- Class II: The second class consists of individuals or groups of individuals (living entities) standing still at a particular location in the scene. This typically happens when people are waiting for someone or are chatting in a group.

- Class III: The third class consists of objects (non-living entities) which have been kept, dropped or abandoned at some location in the scene. This typically happens when people keep their belongings on the floor and stand beside them, or if an object belonging to the background is moved through the scene (e.g. a trolley), or if some piece of luggage has been dropped/abandoned.

We are mainly interested in the objects belonging to class III.

4.2.1 Deciding Whether the Entity is Moving or Static

Given a set of pixel locations that have consistently been marked as foreground, we have to decide whether the pixel set belongs to class I or not. Roughly speaking, class I consists of those regions of the scene that are observing a lot of traffic. Thus, if we maintain an average of the input frames over time, we should see very little similarity between the average and the current input frame over that region.

Therefore, to distinguish the moving entities from the static ones, we maintain a exponentially weighted moving average,MF, over the input frames:

$$MF_i[t] = \lambda I_i[t] + (1 - \lambda) MF_i[t-1] \tag{4.3}$$

where $MF[t]$ is the moving average measure at time t and λ is a constant. Figure 4-2

Figure 4-2: Moving average image, MF, in sample datasets

shows the MF snapshot for datasets at some instant of time.

Now, our claim is that the similarity between $MF[t]$ and $I[t]$ over the given pixel set (say $\mathcal{S}$) will be low if the set belongs of class I, and high otherwise. This is easy to verify from the images in Figure 4-2. Foreground objects and people that are static in the scene can be seen clearly in MF, otherwise we can only see a ghost for people that are moving through the seen.

We use normalized cross-correlation to measure the degree of similarity. So, if the normalized cross-correlation, ρ, of $MF[t]$ and $I[t]$ over the set $\mathcal{S}$ is less than a certain threshold, we classify the set as belonging to class I:

$$\rho = \sum_{i \in \mathcal{S}} \frac{(MF_i - \overline{MF_{\mathcal{S}}})(I_i - \overline{I_{\mathcal{S}}})}{(\sigma_{MF_{\mathcal{S}}})(\sigma_{I_{\mathcal{S}}})} \tag{4.4}$$

where $\overline{MF_{\mathcal{S}}}$ and $\overline{I_{\mathcal{S}}}$ are the means and $\sigma_{MF_{\mathcal{S}}}$ and $\sigma_{I_{\mathcal{S}}}$ are the respective standard deviations of MF and I over the set $\mathcal{S}$.

4.2.2 Deciding Whether the Entity is Living or Non-Living

Given a pixel set that is consistently being marked as foreground and does not belong to class I, we have to decide whether the set belongs to class II or III. Roughly speaking, class II consists of living entities like humans or pets and class III consists of non-living entities like luggage.

If we observe a living being over the course of time, no matter how stationary it might be, there will still be some amount of motion or jitter on the fringes of the silhouette. Therefore, we make the following assumptions to take this decision: if we take a difference over the pixel set between two successive image frames, we can distinguish between static objects and humans based on whether the pixel set exhibits some motion or jitter on the fringes. We classify the entity as belonging to class II if there is some jitter on the silhouette boundary, and class III otherwise.

To detect this jitter, we compute the normed difference, $Diff$, over the pixel set

between the two image frames, say $I[t]$ and $I[t-1]$.

$$Diff_i = \|I_i[t] - I_i[t-1]\| \forall i \in \mathcal{S} \tag{4.5}$$

We can easily verify this observation in Figure 4-3 and Figure 4-4. Figure 4-3 consists of image snippets drawn from the normed difference over regions occupied by humans while Figure 4-4 consists of image snippets drawn from those occupied by objects. The motion on the fringes is clearly visible in case of humans.

Figure 4-3: Examples showing observable jitter in living entities

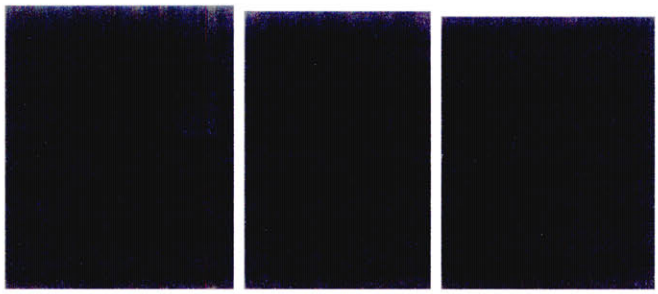

Figure 4-4: Examples showing no observable jitter in non-living entities

We then compute the Discrete Cosine Transform (DCT) of the normed differences, $Diff$. After we have computed the DCT of the pixel set, we take the first 5×5 coefficients of the DCT Image and use a Support Vector Machine (SVM) to classify the set. Normed difference for sets belonging to class II have higher intensities on the fringes and thus their DCT coefficients are larger than those belonging to class III. We use a set of 50 trained examples to train our SVM. As an example, the first

image in Figure 4-3 has a DCT matrix:

$$\begin{pmatrix} 5.60 & 1.28 & -1.32 & -0.00 & -0.96 \\ 1.18 & 0.61 & -0.15 & -0.40 & -0.89 \\ -2.81 & -0.36 & 0.61 & -1.23 & 0.36 \\ -1.30 & 0.36 & 0.90 & -0.88 & 0.42 \\ 0.58 & -0.53 & 0.25 & 1.32 & -0.50 \end{pmatrix}$$

while the first image in Figure 4-4 has a DCT matrix:

$$\begin{pmatrix} 0.83 & -0.07 & -0.14 & 0.02 & 0.01 \\ -0.17 & 0.10 & 0.04 & -0.14 & 0.11 \\ 0.04 & -0.11 & 0.00 & 0.06 & -0.01 \\ 0.04 & 0.04 & 0.02 & 0.01 & -0.01 \\ -0.02 & 0.03 & -0.01 & -0.06 & 0.03 \end{pmatrix}$$

It is easy to see a marked difference in the magnitude of coefficients for both the matrices. Our SVM has 8 support vectors. As an example, one of the support vectors has the following matrix:

$$\begin{pmatrix} 1.01 & -0.16 & 0.04 & -0.01 & -0.04 \\ -0.21 & -0.02 & 0.13 & 7.23e-03 & 4.20e-04 \\ -0.09 & -0.05 & 0.02 & 0.02 & 0.07 \\ 0.065 & 0.01 & -0.02 & -0.08 & -6.77e-03 \\ 0.01 & -0.06 & -0.03 & 7.48e-03 & -0.03 \end{pmatrix}$$

The important thing to note here is that the classification technique has been devised keeping in mind the design constraint that false positives are better than false negatives. So, a person standing absolutely still and getting classified as an object is acceptable, but an object should not be classified as a human. Since there are several cases in which human motion is not perceptible by the camera, this processing stage adds the maximum number of false positives to the results.

4.2.3 Deciding Whether the Object could be Abandoned

Having detected a nonliving object which has been static in the scene, the next step is to determine if the object has been abandoned by its owner. To make this determination, we check to see if the perceived owner of the object is present in the immediate vicinity or not.

To determine the owner of the object, we look for the first frame where the object appears in its current location. We do this by searching for the first input frame whose normalized cross-correlation with the current input frame over the pixel set $\mathcal{S}$ is greater than a certain threshold.

Figure 4-5: Earliest frame where the object appears compared with the current frame

After locating the frame, we look for foreground blobs in the immediate vicinity of the dropped object. The blob which is connected to the object in the foreground image is assumed to be its owner. If there is no such blob or there are multiple such blobs, we mark the object as abandoned to be on the safer side. But if there is a unique such blob, to decide whether the object has been abandoned, we look for the perceived owner within a radius of r pixels in the current frame. We do this by taking the normalized cross-correlation of the pixel set containing the perceived owner with the current input frame, over a window of dimension $2r \times 2r$ centered at the object. If we cannot find the pixel set within this window, we flag the object as potentially abandoned and respond in an appropriate way.

Figure 4-6 shows an example where we use this technique. The image on the left shows the frame where the abandoned object appears in its current location for the first time and the image on the right shows the current frame.

This method of looking for the true owner does not work well in extremely crowded scenes due to a couple of reasons. First, in extremely crowded scenes, the object blob is usually connected to more than one foreground blob; hence, it becomes difficult to decide with any accuracy as to which one of them is the true owner. Second, in case where the object blob is only connected to a unique blob, that blob typically consists of more than one person and under such cases, cross-correlation usually fails.

4.2.4 Examples

Figure 4-6 shows several examples of abandoned object detection in sample datasets. Green boxes indicate entities belonging to class I, blue boxes to class II, and red boxes to class III. For our purpose, we are only interested in red boxes, which correspond to objects which have been identified to be potentially abandoned.

From the results, we can see that our system has accurately identified the abandoned object in all four cases. It had no false negatives, although it had some false positives. In the first sequence, a person sitting on a chair in the left, a person standing in front of the coffee shop and a person standing on the floor upstairs were detected as abandoned objects due to their imperceptible motion. In the second sequence, the region that was occupied by the garbage bins was classified as an abandoned object. This was because the garbage bins were initially stationary and were incorporated as part of the background model. In the third and fourth sequence, there were no false positives and the system correctly identified the abandoned object.

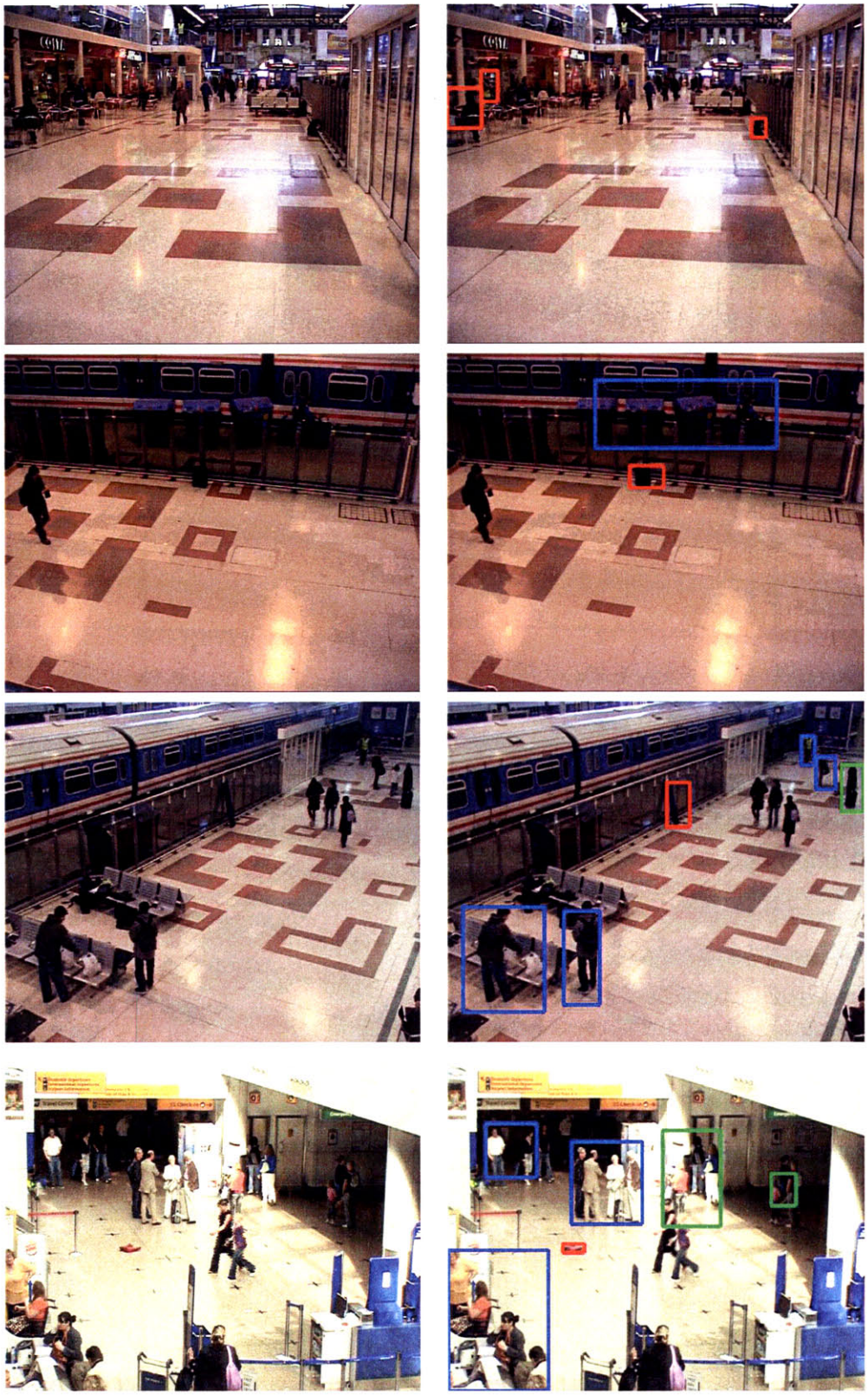

Figure 4-6: Abandoned object detection in sample datasets

Chapter 5

Assessment and Evaluation

In this chapter, we describe the experiments we performed to test our abandoned object detection algorithm under different surveillance settings. An analysis of the results is provided along with the results.

5.1 Datasets

To evaluate our system, we used public datasets from PETS 2006 and PETS 2007. We will present our results on 4 of these sequences, all of which contain an instance of abandoned luggage.

The PETS-2006 dataset contains sequences which have been taken from a moderately busy railway station. In consists of people walking individually as well as in larger groups. The videos have a resolution of 720 $\times$ 576. The PETS-2007 dataset contains sequences from an airport that is densely crowded and has a variety of activities going on.

As mentioned in Section 1.2.3, our algorithm is designed for mid-field domain. It works best when objects are clearly visible. Some of the samples in the PETS dataset consist of regions where the objects are not clearly visible. But, we do not suppress any part of the scene.

In all of the sample datasets, the item which is being abandoned is different in shape and size.

- Sequence S1 from PETS-2006: The dataset S1 consists of an "abandoned object activity" where a person drops off a backpack and walks away. Challenges include a man sitting on a chair, a man standing still on the left and a person standing on the floor upstairs.

- Sequence S2 from PETS-2006: The dataset S2 features an "abandoned object event" where two men meet in front of a fence. They exit from the scene, leaving the luggage behind. In the meantime, a worker directly behind the men, on the other side of the tinted wall, moves three garbage bins away with a tractor. Challenge includes handling the motion of the garbage bins.

- Sequence S5 from PETS-2006: The dataset S5 features an "abandoned object event" where a person drops off a large piece of luggage in front of a tinted wall. The shape of the luggage is elliptical, like a human being. Challenge includes differentiating the luggage from human beings.

- Sequence S7 from PETS-2007: The dataset S7 features an "abandoned object event" when a lady drops off a red bag in the center of the scene. The scene is very crowded and sees a lot of foot traffic. Challenge includes handling of occlusion and dense crowds.

The results that we obtained with the sample datasets were in line with our goals and design constraints. We designed our system to prefer false positive over false negative. There were no false negatives in the system and we were able to detect all instances of abandoned objects. But, there were some instances of false positives.

We will now go over all the subparts of the system and discuss the results individually for each of them.

5.2 Background Model

We will first present the background images for all four datasets. To initialize our background model, we used about 200 uniformly sampled images from the video

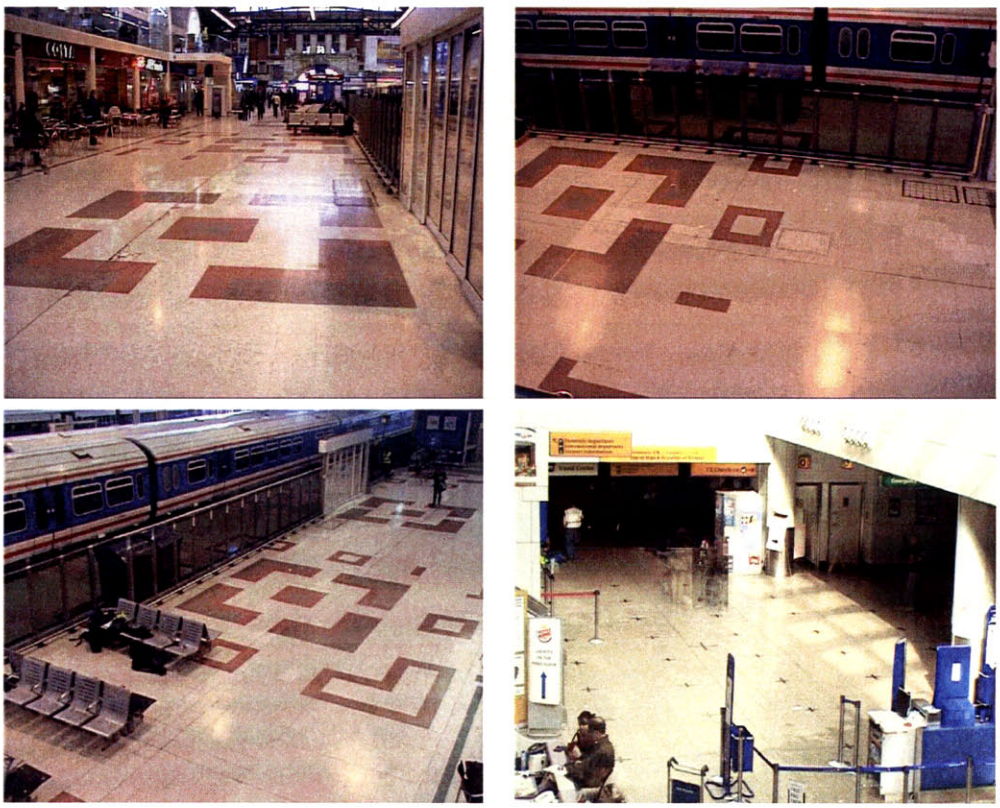

Figure 5-1: Background Image for all four datasets

clip. Fig. 5-1 shows the background images (E) obtained using the weighted average method in Section 3.2.2.

In all four datasets, we can observe that some individuals have also been included as a part of the background model. This is mainly because they were present at that location for the entire duration of background model initialization and hence, they were inferred to be a part of the background.

Another problem with the background model that can be observed in sequence S7 is that for image regions containing objects that can be present in more than one state (like a door), the background model can only capture one state. Sequence S7 contains a door on the upper right that is open in some of the training images and closed in others. The background model captures it in an open state.

5.3 Foreground Segmentation

Figures 5-2 to 5-17, show some sample results obtained using our foreground segmentation algorithm presented in Section 3.3. In our experiments, we were able to segment the foreground quite accurately most of the time except for a few notable exceptions.

First, foreground extraction is problematic in regions where the background initialization was not proper due to presence of a foreground object for the entire duration. For example, in sequence S2 (Figure 5-6 to 5-9), the garbage bins were part of the background model, and as they moved away, the actual background was segmented as a foreground. Second, in areas of bright illumination, the shadows are captured to be a part of the foreground. Third, in case of an object having multiple states, some of the states are segmented as foreground. For example, in sequence S7 (Figure 5-14 to 5-17), for the door present on the upper right, the background model captured the door in an open state. As a result, whenever the door is open, it is segmented as foreground. Fourth, in regions where the camera view is far-field instead of mid-field, the algorithm does a more coarse job joining multiple people together as part of foreground, instead of a finer segmentation.

Figure 5-2: Foreground results for dataset S1 from PETS-2006: Example 1

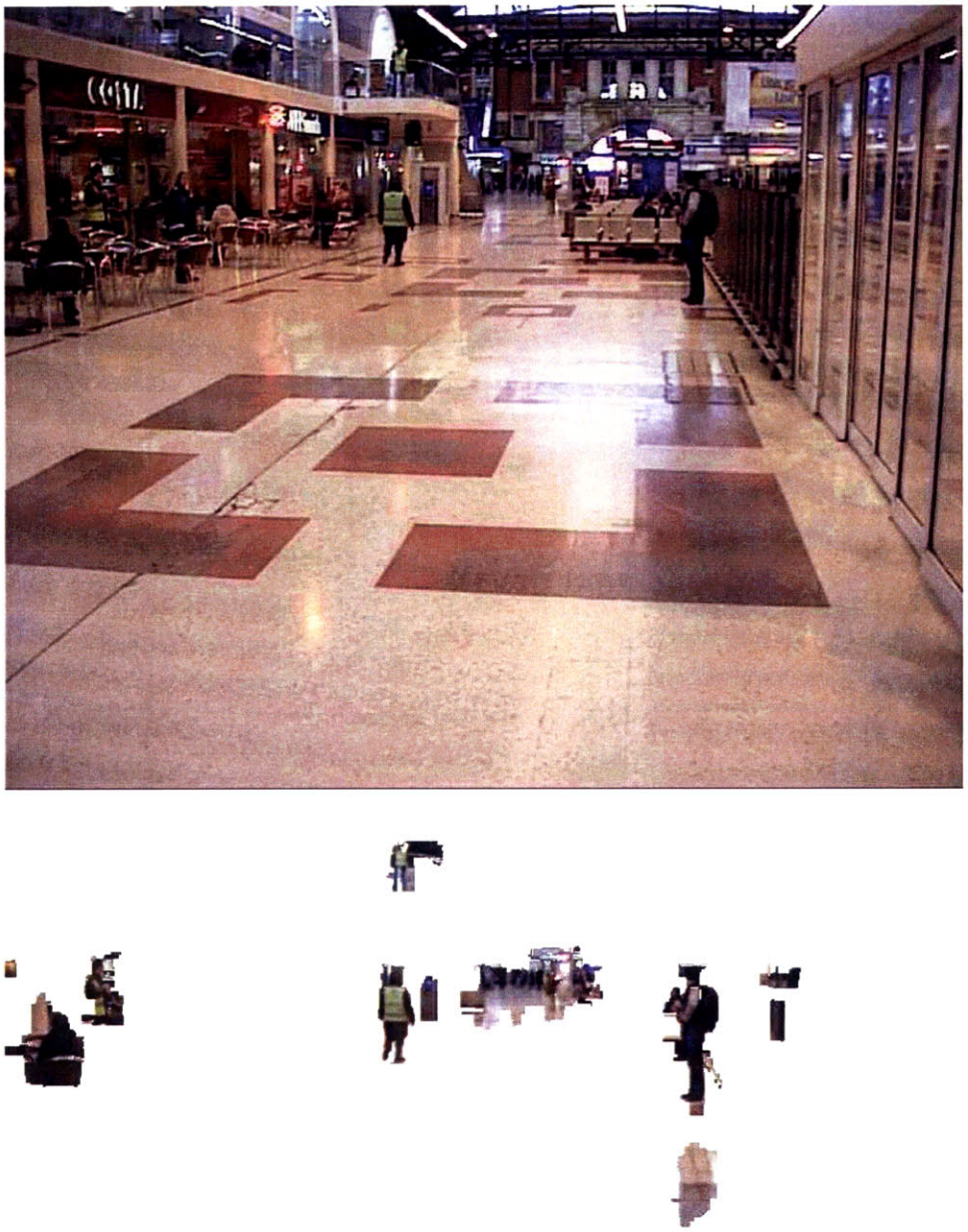

Figure 5-3: Foreground results for dataset S1 from PETS-2006: Example 2

Figure 5-4: Foreground results for dataset S1 from PETS-2006: Example 3

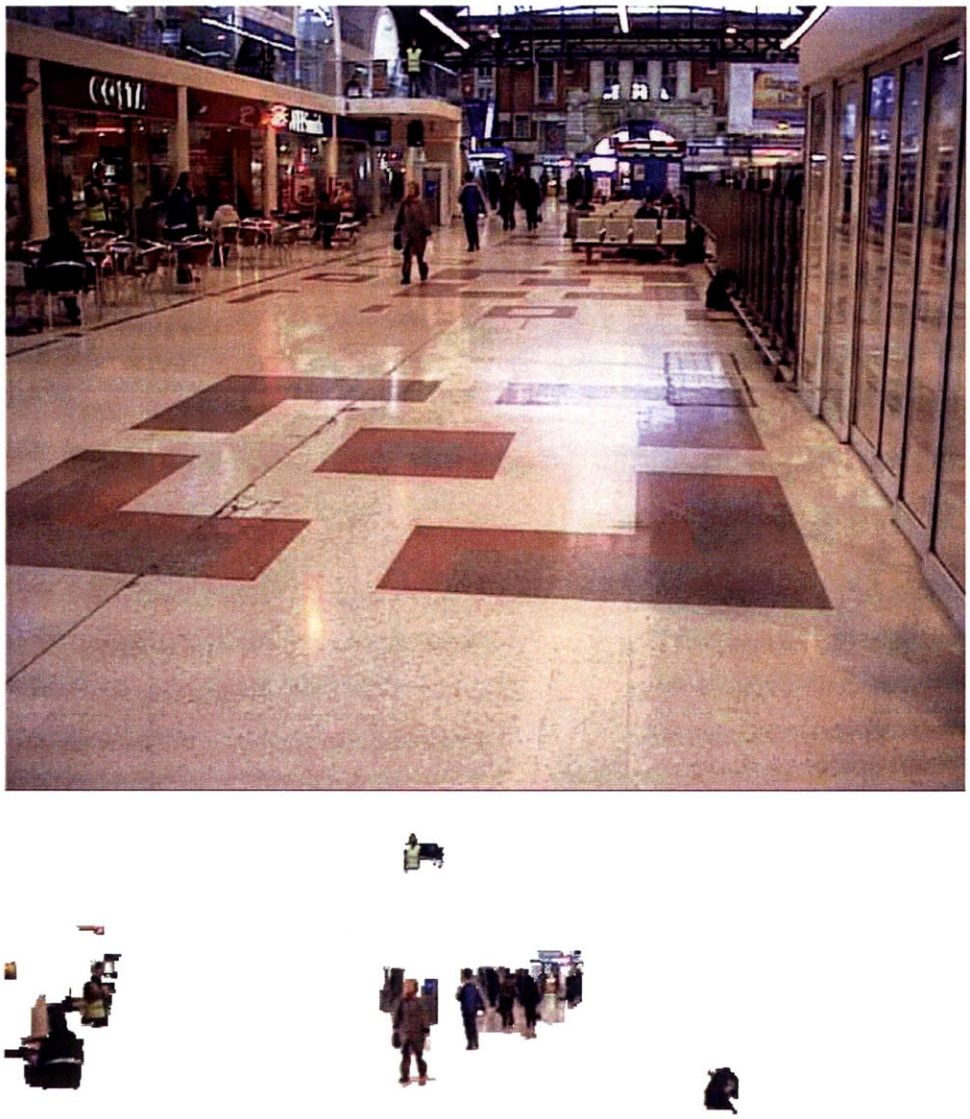

Figure 5-5: Foreground results for dataset S1 from PETS-2006: Example 4

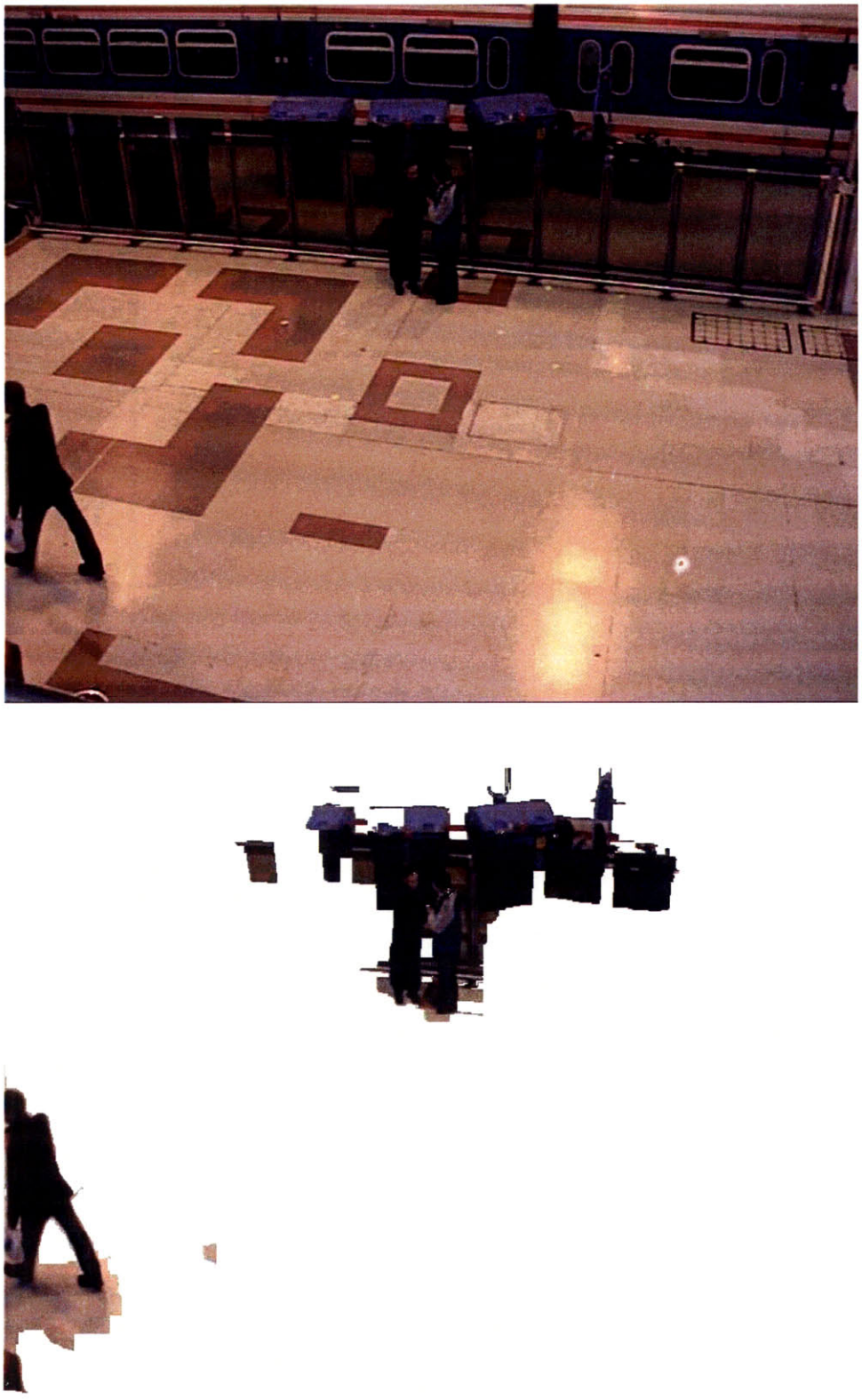

Figure 5-6: Foreground results for dataset S2 from PETS-2006: Example 1

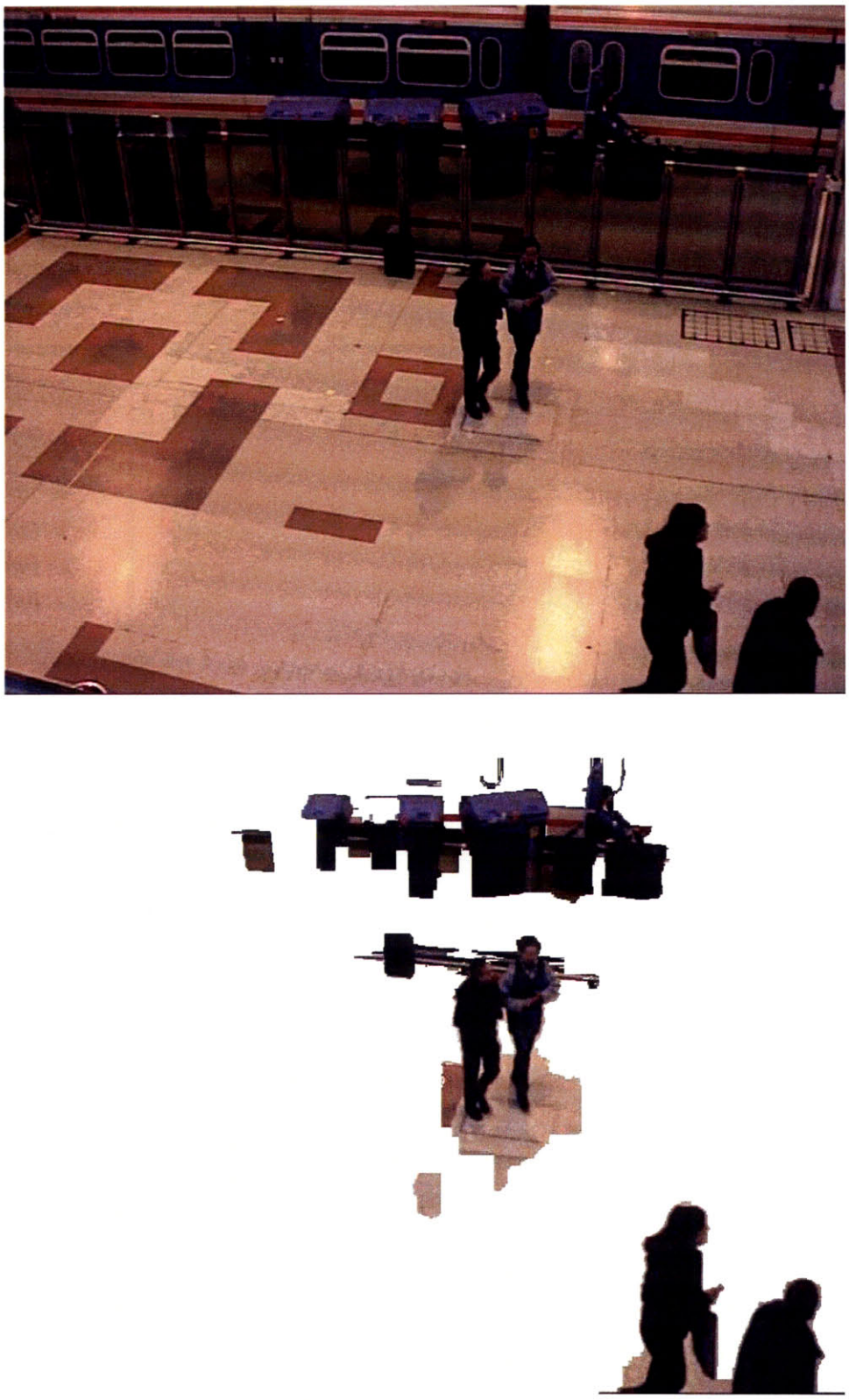

Figure 5-7: Foreground results for dataset S2 from PETS-2006: Example 2

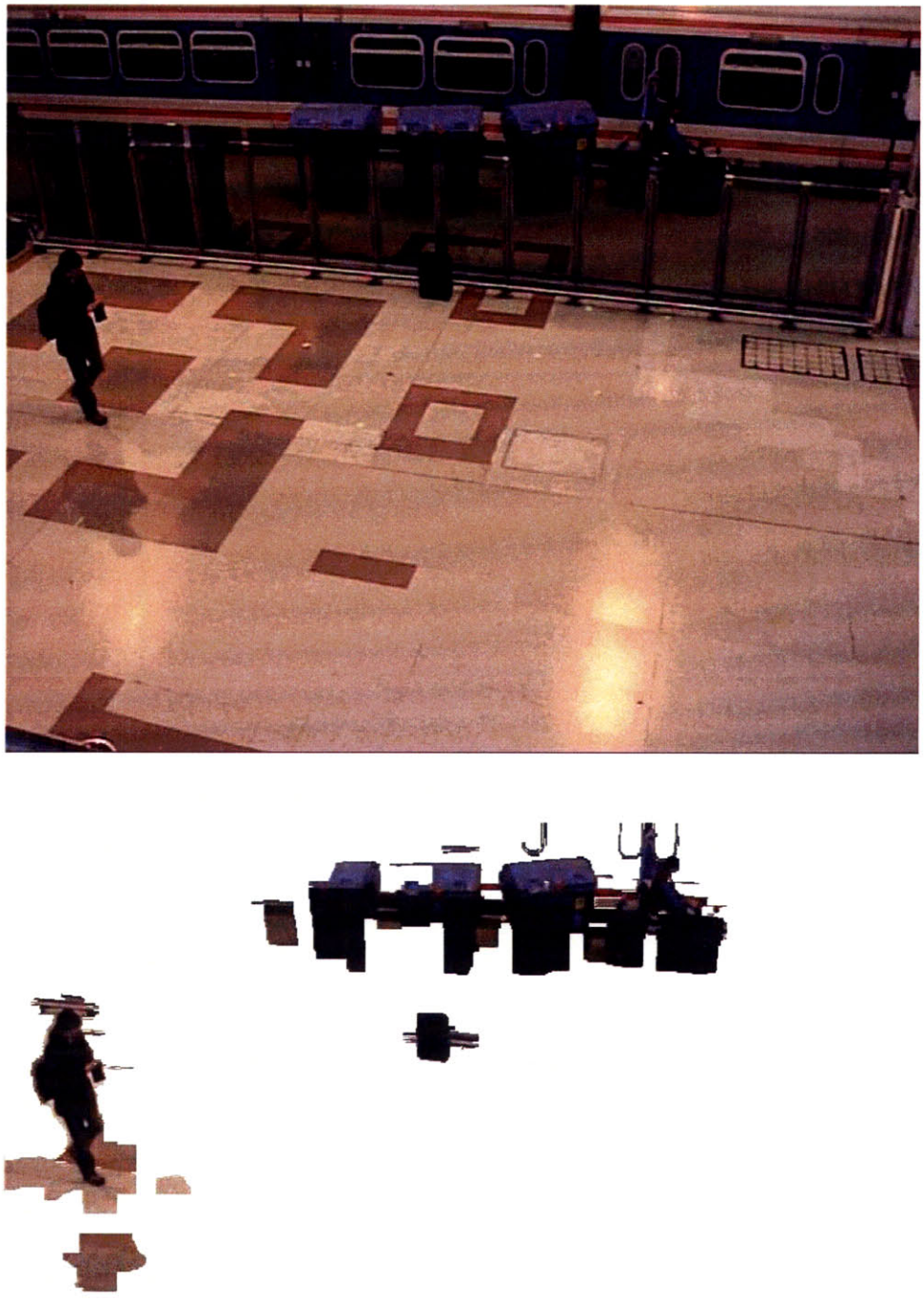

Figure 5-8: Foreground results for dataset S2 from PETS-2006: Example 3

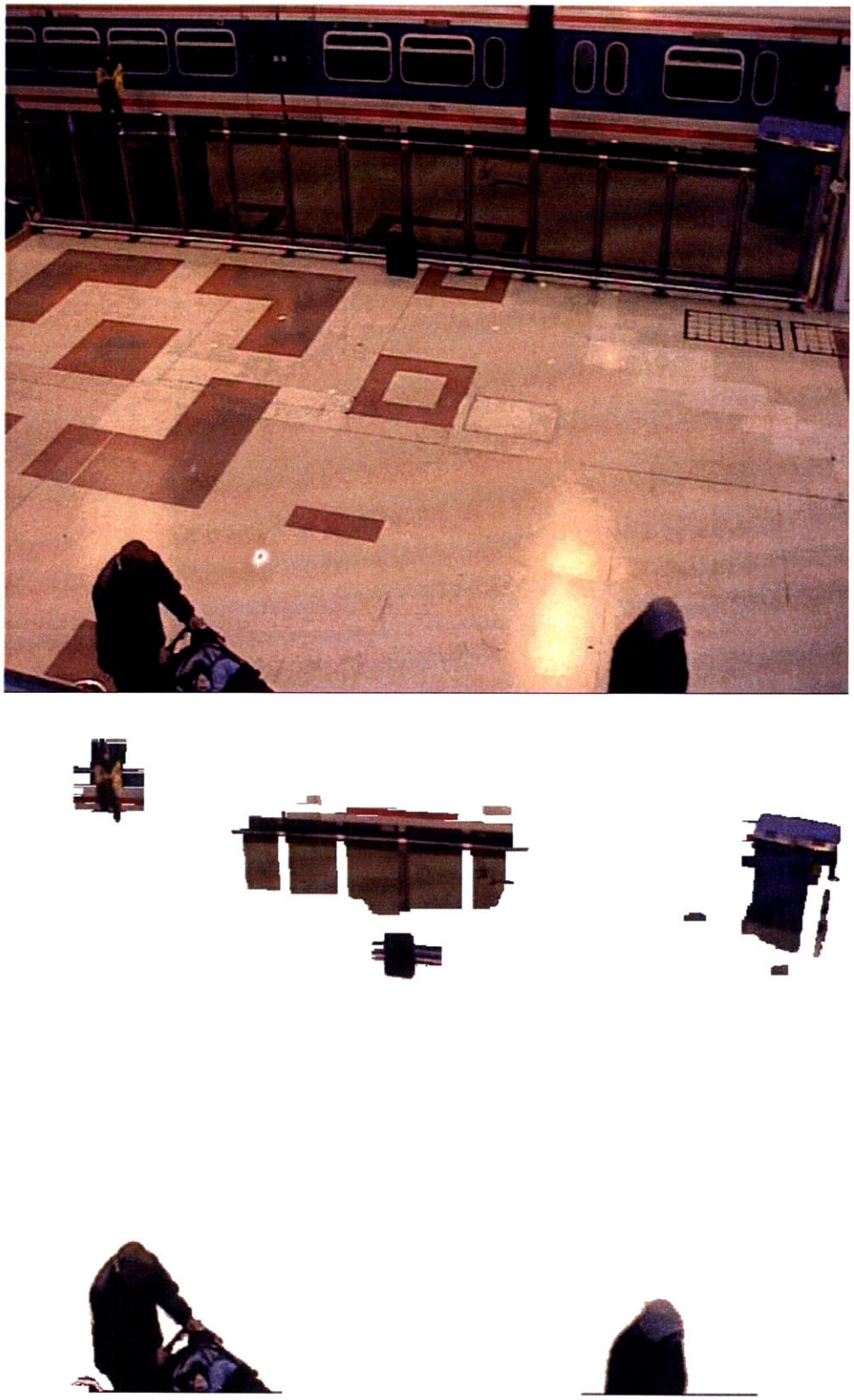

Figure 5-9: Foreground results for dataset S2 from PETS-2006: Example 4

Figure 5-10: Foreground results for dataset S5 from PETS-2006: Example 1

Figure 5-11: Foreground results for dataset S5 from PETS-2006: Example 2

Figure 5-12: Foreground results for dataset S5 from PETS-2006: Example 3

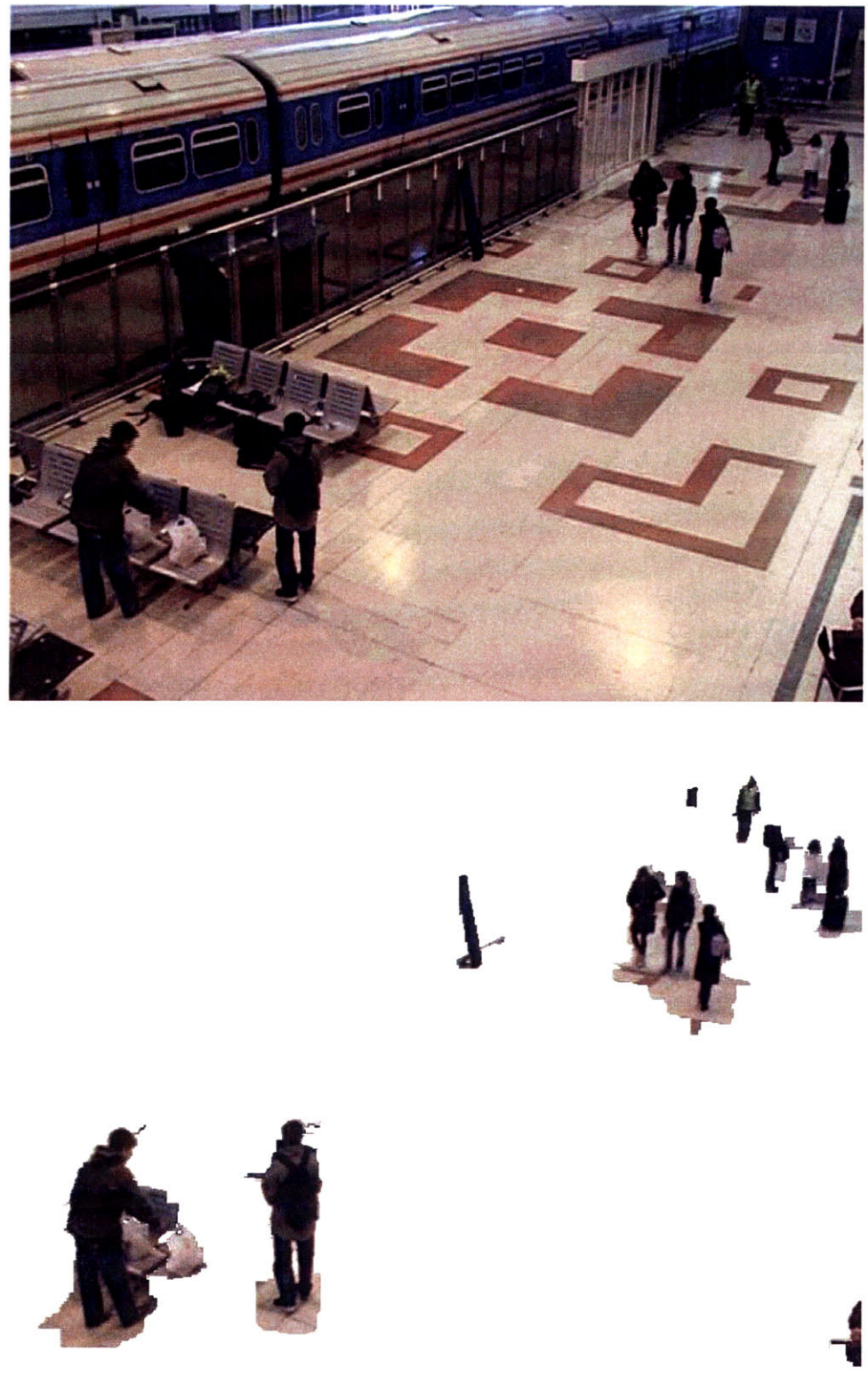

Figure 5-13: Foreground results for dataset S5 from PETS-2006: Example 4

Figure 5-14: Foreground results for dataset S7 from PETS-2007: Example 1

Figure 5-15: Foreground results for dataset S7 from PETS-2007: Example 2

Figure 5-16: Foreground results for dataset S7 from PETS-2007: Example 3

Figure 5-17: Foreground results for dataset S7 from PETS-2007: Example 4

5.4 Abandoned Object Detection

Our system accurately detected the abandoned object in all four cases. It had no false negatives, although it had some false positives.

In sequence S1 (Figure 5-18), a person sitting on a chair in the left, a person standing in front of the coffee shop and a person standing on the floor upstairs were detected as abandoned objects due to their imperceptible motion. The reason that the system was unable to perceive any motion in their case is that all of them fall under the far-field domain. Our system is designed for mid-field domain and our algorithm to distinguish between people and objects assumes that we will see motion on the fringe in case of people. If some region falls under the far-field domain, the motion on the fringe is imperceptible for the camera and hence, blobs in that region get classified as objects.

In sequence S2 (Figure 5-19), the region that was occupied by the garbage bins was classified as an abandoned object. This was because the garbage bins were initially stationary and were incorporated as part of the background model. When they moved from their original location, the system classified that region as foreground. Since there was no motion on the fringe in that region, the system identified it to be an abandoned object.

In sequence S5 and S7 (Figures 5-21 and ??), there were no false positives and the system correctly identified the abandoned object.

In summary, our system works very well even in densely crowded scenes. It is able to detect abandoned objects with high recall, and good precision. It does not make any assumptions about the abandoned objects and detects them even in cases where they bear a striking similarity to human beings.

Our system has the following advantages over traditional systems:

- We do not filter the objects based on shape or size. Therefore, we can identify abandoned objects of any shape or size.

- Computational load of the proposed system is very low. Since we perform binary labeling for foreground extraction and only pixel level operations for

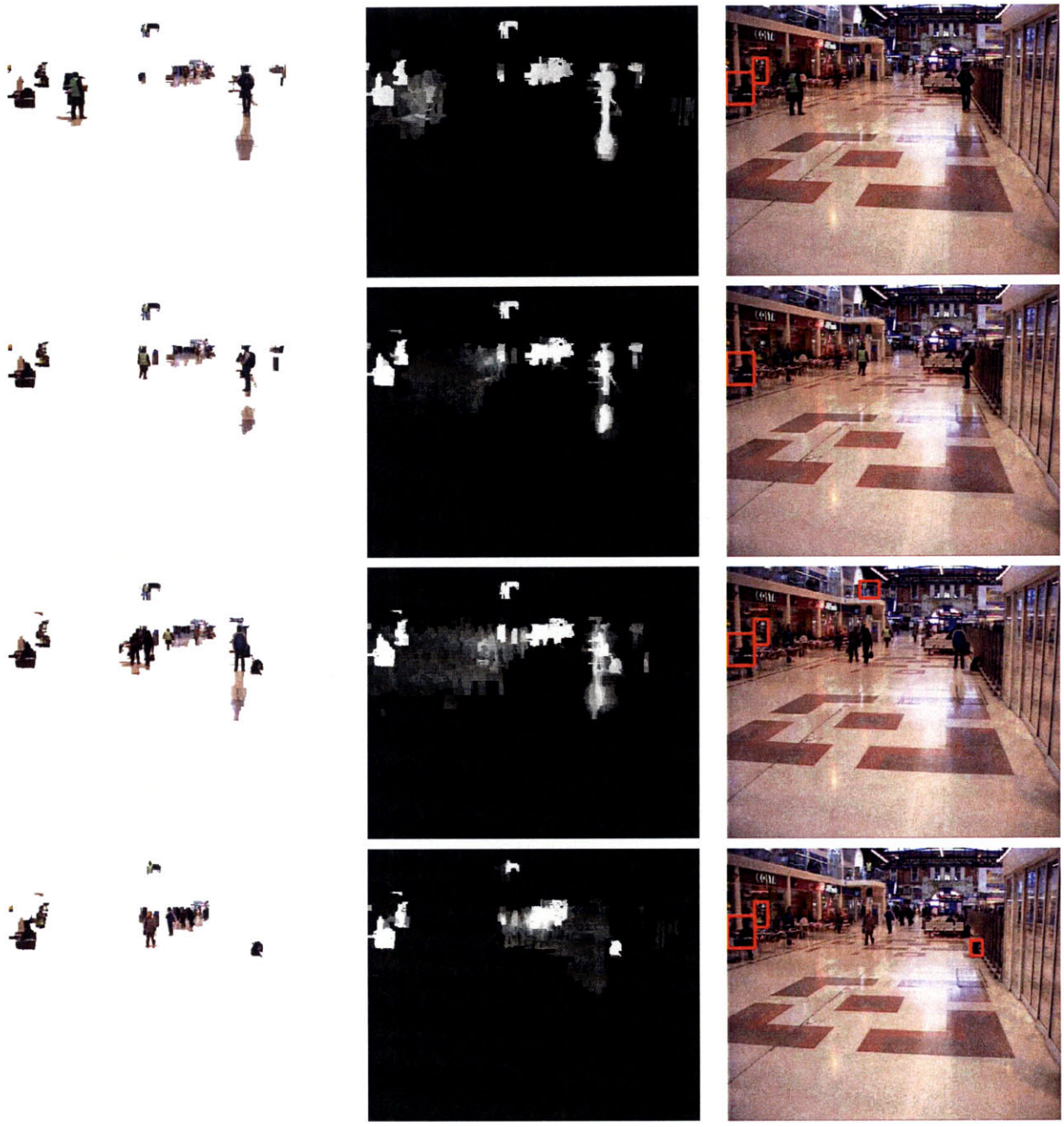

Figure 5-18: Sample results for dataset S1 from PETS-2006

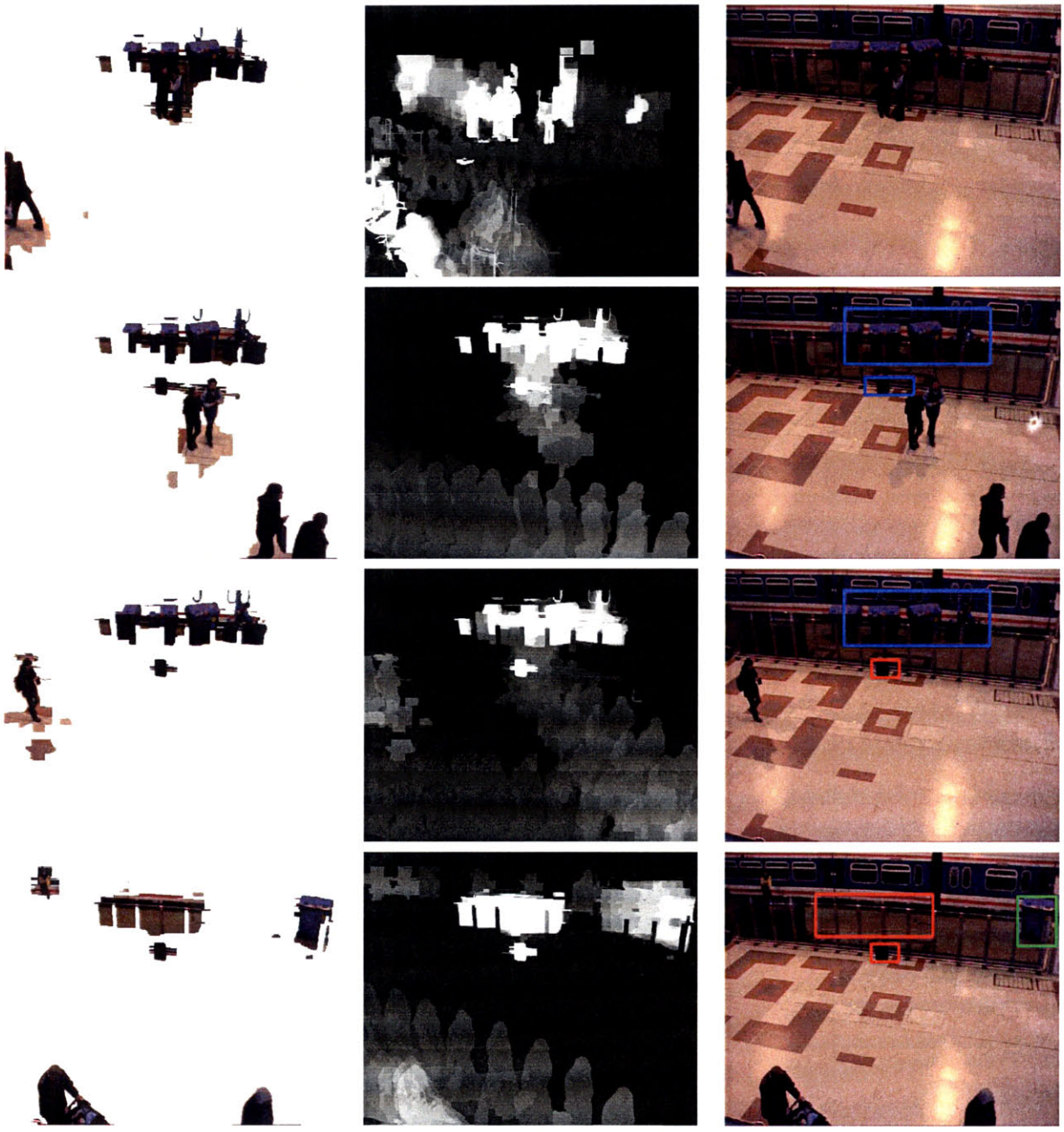

Figure 5-19: Sample results for dataset S2 from PETS-2006

Figure 5-20: Sample results for dataset S5 from PETS-2006

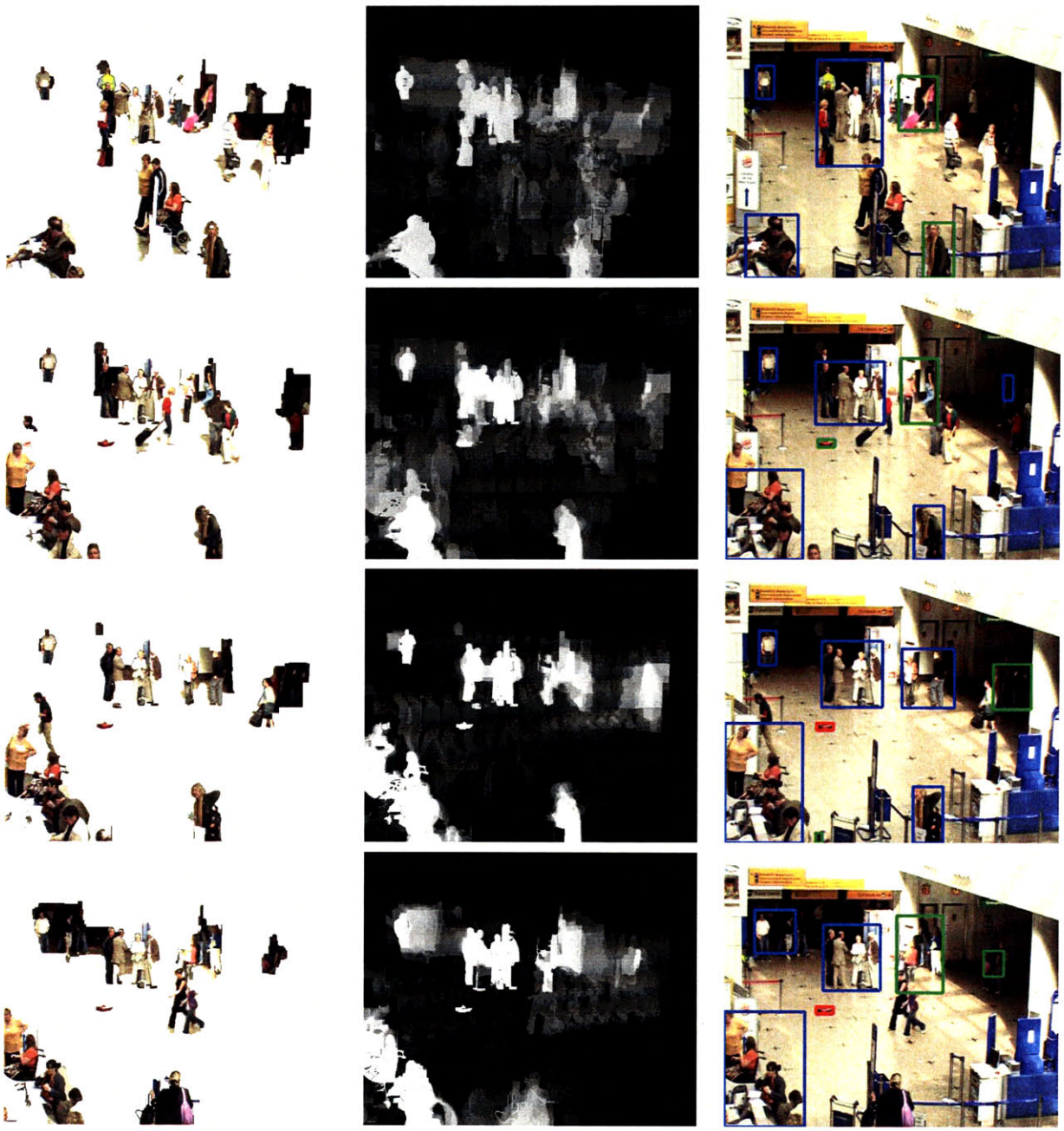

Figure 5-21: Sample results for dataset S7 from PETS-2007

change analysis, the system could be used real-time with some optimizations.

- The storage requirements of the system are very low as we do not have to store huge amounts of tracking data.

Our system also has the following shortcomings:

- In case of occlusion of the abandoned object by a moving entity, the system is not able to identify the abandoned object.

- In case of imperceptible motion by an individual standing at a location (particularly far-field), the system usually classifies him/her as an object, thus increasing the number of false positives.

Chapter 6

Conclusion

We have proposed a system for abandoned object detection in crowded scenes that addresses some of the major challenges in the field. Use of a Markov Random Field based foreground segmentation algorithm, along with a pixel level approach for detecting static blobs in the scene, allow us to identify potentially abandoned objects effectively.

We have a background model that can be initialized in the presence of foreground objects. It computes brightness and chromaticity distortion information and models foreground segmentation as an energy minimization problem, imposing smoothness constraints. We use a pixel level approach to determine blobs that have been static in the scene for a certain amount of time and employ a Support Vector Machine based classifier to determine whether the blob consists of an object or person. Our approach is especially effective for mid-field views.

Our system is very time efficient as it performs only pixel level operations. It is also space efficient as it does not store object trajectories from the past. Moreover, we do not have to deal with hard problems like object tracking and segmentation in our system, which are prone to errors in crowded scenes. It does have some false positives, but hardly any false negatives and thus, can be very effective in guiding security personnel about any potentially abandoned objects in the scene.

Bibliography

[1] Jung H. Ahn and Hyeran Byun. Accurate foreground extraction using graph cut with trimap estimation. In *PSIVT*, pages 1185–1194, 2006.

[2] D. Biliotti, G. Antonini, and J.P. Thiran. Multi-layer hierarchical clustering of pedestrian trajectories for automatic counting of people in video sequences. In *Motion*, pages II: 50–57, 2005.

[3] Y.Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary region segmentation of objects in n-d images. volume 1, pages 105–112 vol.1, 2001.

[4] Bjorn Schuller Dejan Arsic, Martin Hofmann and Gerhard Rigoll. Multi-camera person tracking and left luggage detection applying homographic transformation. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2007.

[5] C. Rougier M. Dahmane E. Auvinet, E. Grossmann and J. Meunier. Left-luggage detection using homographies and simple heuristics. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2006.

[6] Bo Wu Vivek Kumar Singh Fengjun Lv, Xuefeng Song and Ramakant Nevatia. Left-luggage detection using bayesian inference. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2006.

[7] E.B. Fox, D.S. Choi, and A.S. Willsky. Nonparametric bayesian methods for large scale multi-target tracking. pages 2009–2013, 29 2006-Nov. 1 2006.

[8] Zhouyu Fu, Weiming Hu, and Tieniu Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *ICIP (2)*, pages 602–605, 2005.

[9] Xiaogang Wang Gerald Dalley and W.E.L. Grimson. Event detection using an attention based tracker. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2007.

[10] Leo A. Goodman and Harry Markowitz. Social welfare functions based on individual rankings. *The American Journal of Sociology*, 58(3):257–262, November 1952.

[11] Sadiye Guler and Matthew Farrow. Abandoned object detection in crowded places. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2006.

[12] I. Haritaoglu, D. Harwood, and L.S. Davis. W4: Real-time surveillance of people and their activities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):809–830, August 2000.

[13] T. Horprasert, D. Harwood, and L. S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proc. IEEE ICCV*, volume 99, pages 1–19.

[14] Anya C. Hurlbert. The computation of color. Technical report, Cambridge, MA, USA, 1989.

[15] Jorge Ral Gmez Jess Martnez-del Rincn, J. Elas Herrero-Jaraba and Carlos Orrite-Uruuela. Automatic left luggage detection and tracking using multi-camera ukf. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2006.

[16] I.N. Junejo, O. Javed, and M. Shah. Multi feature path modeling for video surveillance. In *ICPR*, pages II: 716–719, 2004.

[17] Pedro Quelhas Kevin Smith and Daniel Gatica-Perez. Detecting abandoned luggage items in a public space. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2006.

[18] Xi Li, Weiming Hu, and Wei Hu. A coarse-to-fine strategy for vehicle motion trajectory clustering. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 591–594, Washington, DC, USA, 2006. IEEE Computer Society.

[19] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press, 2001.

[20] Peter Nillius, Josephine Sullivan, and Stefan Carlsson. Multi-target tracking - linking identities using bayesian network inference. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2187–2194, Washington, DC, USA, 2006. IEEE Computer Society.

[21] Nuria M. Oliver, Barbara Rosario, and Alex P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[22] A.E.C. Pece. From cluster tracking to people counting. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 9–17, 2002.

[23] F.M. Porikli and T. Haga. Event detection by eigenvector decomposition using object and frame features. In *Detection and Recognition of Events in Video*, page 114, 2004.

[24] F. Porikly and Z. Yin. Temporally static region detection in multi-camera systems. In *Proc. IEEE Workshop on Performance Evaluation in Surveillance and Tracking*, 2007.

[25] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[26] A. Senior. Tracking people with probabilistic appearance models. In *International Workshop on Performance Evaluation of Tracking and Surveillance*, pages 48–55, 2002.

[27] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[28] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, pages II: 246–252, 1999.

[29] C. Stauffer and W.E.L. Grimson. Learning patterns of activity using real-time tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):747–757, August 2000.

[30] Josephine Sullivan and Stefan Carlsson. Tracking and labelling of interacting multiple targets. In *ECCV*, pages 619–632, 2006.

[31] X.G. Wang, K. Tieu, and W.E.L. Grimson. Learning semantic scene models by trajectory analysis. In *ECCV*, pages III: 110–123, 2006.

[32] T. Xiang and S.G. Gong. Video behaviour profiling and abnormality detection without manual labelling. In *ICCV*, pages II: 1238–1245, 2005.

[33] Lihi Zelnik-Manor and Michal Irani. Event-based analysis of video. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:123, 2001.

[34] H. Zhong, J.B. Shi, and M. Visontai. Detecting unusual activity in video. In *CVPR*, pages II: 819–826, 2004.