Binary Matrix for Pedestrian Tracking in Infrared Images

A thesis submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy

By

Keshava Grama
School of Computing
Edinburgh Napier University, UK

June 25, 2013

Abstract

The primary goal of this thesis is to present a robust low compute cost pedestrian tracking system for use with thermal infra-red images. Pedestrian tracking employs two distinct image analysis tasks, pedestrian detection and path tracking. This thesis will focus on benchmarking existing pedestrian tracking systems and using this to evaluate the proposed pedestrian detection and path tracking algorithm.

The first part of the thesis describes the imaging system and the image dataset collected for evaluating pedestrian detection and tracking algorithms. The texture content of the images from the imaging system are evaluated using fourier maps following this the locations at which the dataset was collected are described.

The second part of the thesis focuses on the detection and tracking system. To evaluate the performance of the tracking system, a time per target metric is described and is shown to work with existing tracking systems. A new pedestrian aspect ratio based pedestrian detection algorithm is proposed based on a binary matrix dynamically constrained using potential target edges. Results show that the proposed algorithm is effective at detecting pedestrians in infrared images while being less resource intensive as existing algorithms.

The tracking system proposed uses deformable, dynamically updated codebook templates to track pedestrians in an infrared image sequence. Results show that this tracker performs as well as existing tracking systems in terms of accuracy, but requires fewer resources.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Pedestrian detection and tracking in image sequences has a wide range of practical applications (this thesis will focus on pedestrian tracking for surveillance, other application areas are in appendix C) and is an active research area. Diverse solutions have been identified as being able to solve challenges in this field such as image segmentation, target occlusion etc. This thesis is focused on a subset of image sequences, those from a thermal infrared camera recording the spectral intensity at 8 - 15μm.

The camera used for recording the dataset uses a micro-bolometer array as a sensor, technical specifications for which is available on the FLIR website [1]. A basic introduction to how a bolometer works when recording infrared radiation is in Appendix B and the book *Uncooled Infrared Imaging Arrays and Systems* by Kruse et al. [2] is a good reference.

After the spectral intensity has been recorded by the array, it is rescaled and interpolated to generate a 560x720 pixel image. Image sequences recorded using this camera form the dataset that has been used to evaluate the algorithms that are described in this document. A sample image recorded using this camera is shown in figure 1.1(a).

In infrared images, pedestrians are easily distinguished from the background because,

- Pedestrian temperature range is usually distinct from the background.
- Less texture information in infrared images

The above statements make pedestrian tracking in infrared images seem trivial. This document will highlight some of the challenges in this domain and also identify some solutions for the same.

The following sections will introduce in the following order; infrared images, image segmentation and object tracking. These subjects are presented before discussing the research context, problem area, the research questions and the contributions to the body of knowledge as they will help understand the motivations behind the research presented in this document.

(a) Sample Infrared Image.

(b) Segmented Infrared Image.

Figure 1.1: Sample Infrared image

1.1 Infrared Image and Texture

A scene in an image will consist of a set of objects that were in the field of view (FOV) of the camera when the image was recorded. Textural cues are used to discriminate between the various objects in a scene and any environmental changes that might be affecting the image recorded. Texture in the context of an infrared image, refers to the intrinsic properties of an object (its temperature, surface texture, emissivity and reflectivity in the IR domain) that affect the *recorded* absolute temperature of the object (for more information on why the temperature recorded by the camera sometimes in different from the absolute temperature of the object see book by Kruse et al. [3]).

The images are formed by recording the magnitude of incident radiation (Appendix B), the recorded value has all the above information. Research has shown that object texture information other than the temperature, while reduced, is not lost during the digitisation and recording process [4, 5]. By identifying the textural cues in a scene, it should be possible to discriminate between objects that are at similar or at the same temperature.

1.2 Image Segmentation and Tracking

In the context of pedestrian detection and tracking, the objects in a scene can classified into two groups, foreground and background. For example, in an image from a sequence observing a street, the background objects are usually static (parked vehicles, buildings, bins, trees etc., figure 1.1(a), C); foreground objects will be in motion (moving vehicles, pedestrians and cyclists; figure 1.1(a), A and B). This does not mean that all background objects are static, background objects sometimes move under the influence of environmental factors (wind, rain, snow etc.), human action (bins being moved, trees and plants being moved) etc.

1.2.1 Background Segmentation

The first step in tracking moving objects, is to segment the pixels in an image into two sets. One consisting of all the background pixels and the second consisting of all the foreground pixels. The algorithms used for achieving this are known as background segmentation algorithms. There are object tracking algorithms that do not utilise a distinct background segmentation mechanism.

The terms *'objects'* and *'pedestrians'* are used interchangeably in this chapter, because when referring in a general to a tracking/classification algorithm, a set of tracked/evaluated pixels is referred to as an object. In the context of a pedestrian tracking/detection algorithm, that object is assumed to represent a pedestrian. Figure 1.1(b) is the segmented equivalent of figure 1.1(a), the background is filled in blue and the foreground objects are filled in green (the image was segmented by hand).

1.2.2 Pedestrian Detection

Foreground regions when identified in an image might be composed of individual objects, amalgamated objects (occluded objects or objects in close proximity) or mis-classified background regions. These regions are then evaluated by an object classification algorithm, to identify and classify the various objects in the foreground into *'objects of interest'* (in this case, pedestrians) or other foreground objects (vehicles, cyclists, misclassified background etc.)

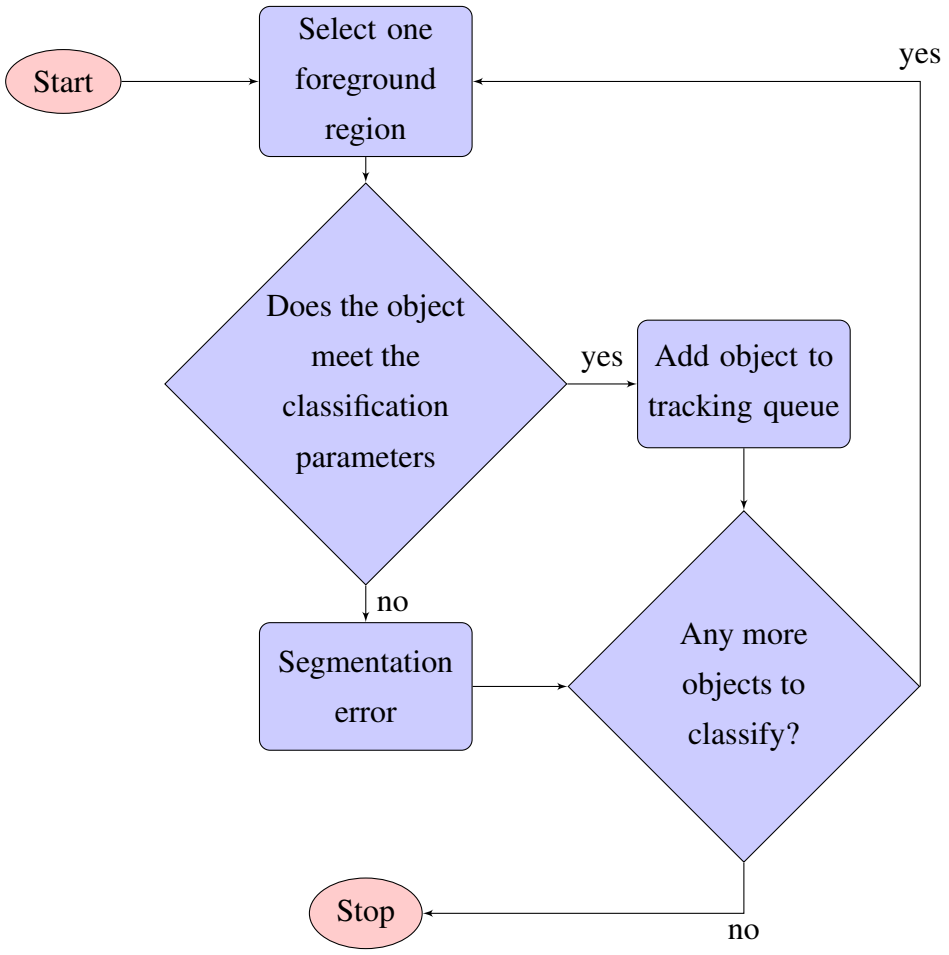
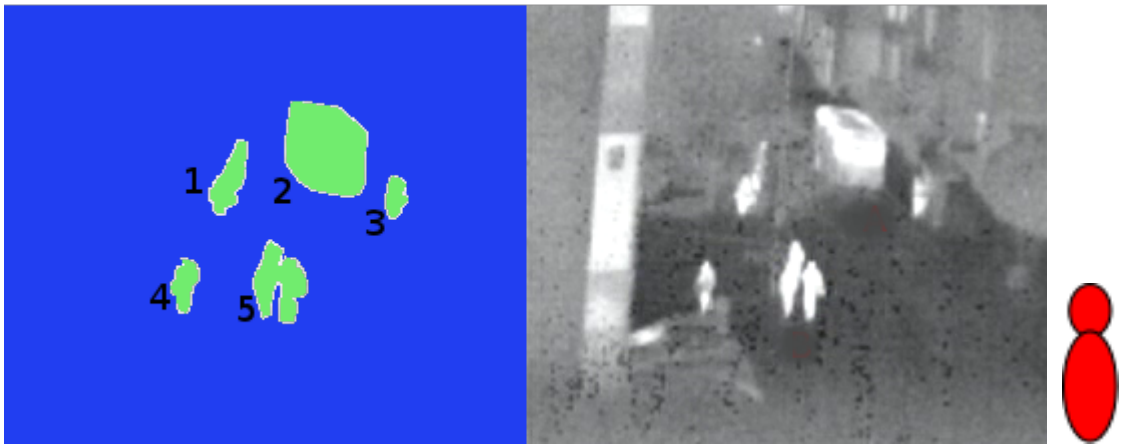


Figure 1.2: Generic Object Classifier

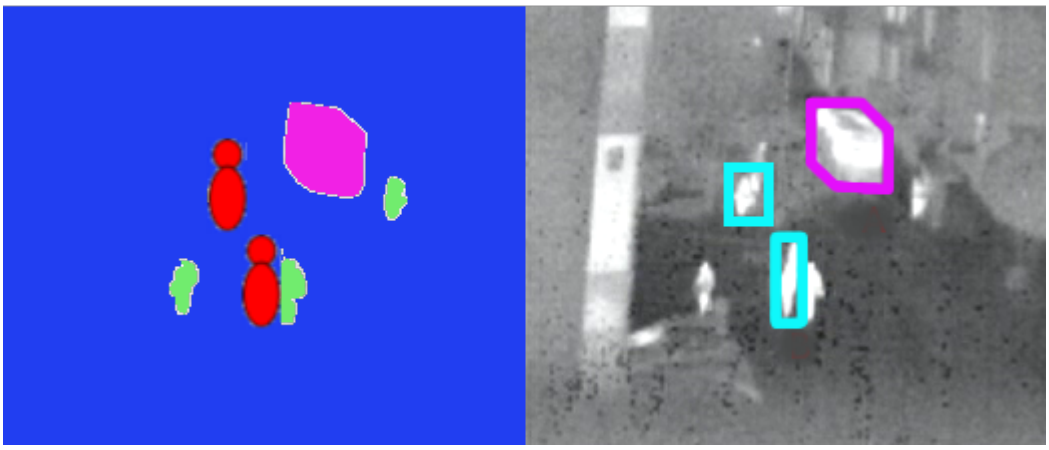
A generic object classification algorithm is shown in figure 1.2, it loops through the foreground regions from the segmented image to identify objects of interest. To illustrate object classification, a generic template (figure 1.3(b)) will be used with the foreground regions from figure 1.1(a). Figure 1.3(a) is a close-up of the foreground regions, the foreground objects are evaluated by the classifier as numbered.

In figure 1.3(a), objects 1 and 5 are composed of more than 1 pedestrian target; object 2 is a vehicle; objects 3 and 4 are single pedestrians. Using these objects, we can look at the terms associated with the different types of pedestrian objects that might be in the foreground.

Object 1 consists of at-least three pedestrians in close proximity. However, as there is insufficient background or temperature difference between them, when the image was



(a) Foreground Regions from Figure 1.1a. (b)



(c) Foreground regions after classification

Figure 1.3: Segmentation and Classification

segmented, the pedestrians are amalgamated into one. Object's 2 and 4, have distinct edges and are easily segmented. The lower half of object 3 has a distinct edge and is easily segmented, but, the upper half is not distinct and has been *'eroded'*. Object 5, is composed of three pedestrians, two of which are easily identified and a third pedestrian which is *'occluded'*. Object 5 as a whole has distinct edges, and is easily segmented.

After the *'objects of interest'* have been identified and labelled, the objects identified as valid are then processed by the tracker. Figure 1.3(c) simulates this by fitting the two of the foreground objects with bounding boxes (light blue boxes), also a foreground object that is not of interest to the tracker (pink box) and misclassified background pixels (green). The performance of a classifier is expressed in terms of its True Positive TP_c, True Negative TN_c, False Positive FP_c and False Negative FN_c rates, more on this in the literature review.

1.2.3 Pedestrian Tracking

Once the objects in the foreground have been classified, the *'objects of interest'* (pedestrians) are tracked by the tracking algorithm to produce the pedestrian track data. Figure 1.4, illustrates a generic tracking algorithm. As with the object classification, there are various ways of implementing an object tracking system.

There are generally two main types of trackers, the first type of tracker identifies and associates (match) the objects in the current image to objects in the previous images of the sequence. The second type of tracker, predicts object locations for the current image and verifies the presence of the objects.

The image sequence in 1.5(a) to 1.5(c), has four objects. The tracks for the objects in figure 1.5(g) and the initial locations are in figure 1.5(d). Figure 1.5(e) shows the results of a tracker using the identify and associate mechanism and figure 1.5(f) shows the results of a tracker using the predict and verify mechanism. As object 4 at the end of the sequence has a sharp change in direction the predicted location and the current location do not match (figure 1.5(f); dark green circle: predicted location, light green circle: ground truth location), a more detailed review of trackers is in the literature review (chapter 2).

The object track data produced by the tracker is evaluated against the *'ground truth'*. The *'ground truth'* is a set of events of interest in the image sequence, that the tracker is required to identify. The ground truth GT for an image sequence consists of the following:

GT_i**:** Total number of pedestrian objects.

$GT_{p,i}$**:** Number of points where object i must be identified.

$GT_{t,i}$**:** The track for object i

GT_o**:** Occlusion events

An occlusion event occurs when an object being tracked, while still physically present in the area under observation, is not part of the foreground. This is usually due to some other object, background or foreground, being in the line of sight between the occluded object and the camera.

As the occluded object is no longer present in the foreground, there is nothing in the current image to track. When the occlusion event ends, the tracker is now presented with what appears to be a new target, this creates a false target and decreases the overall accuracy of the tracking algorithm.

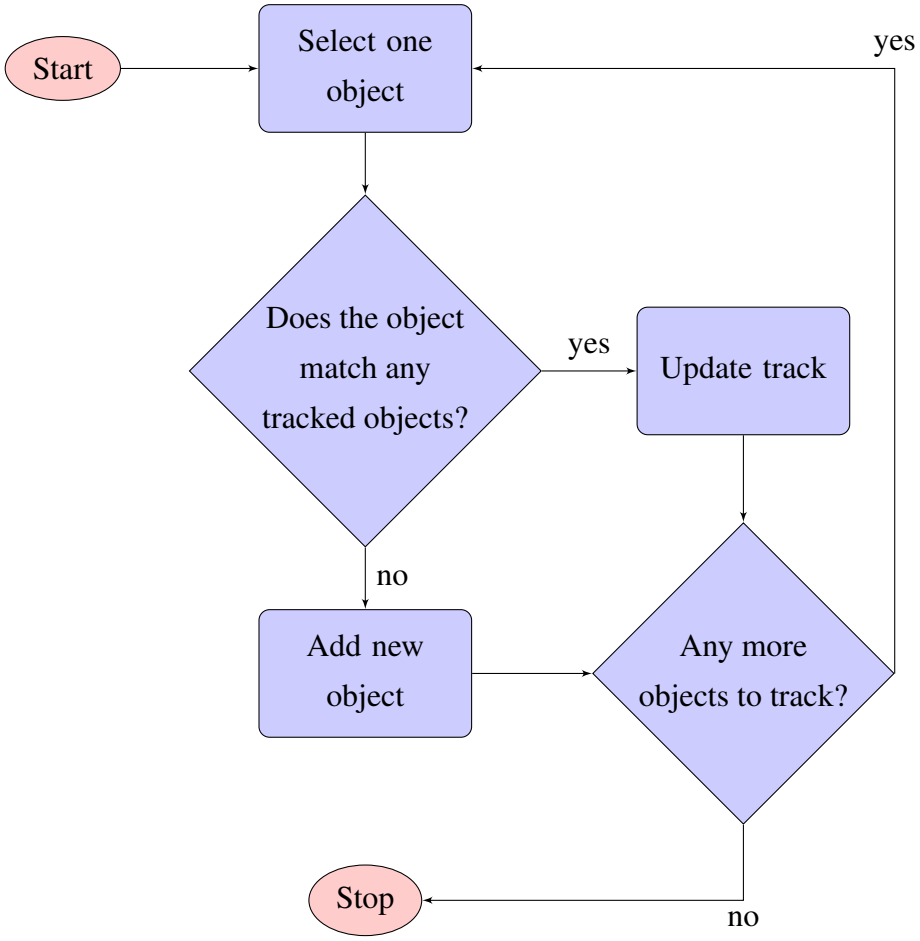
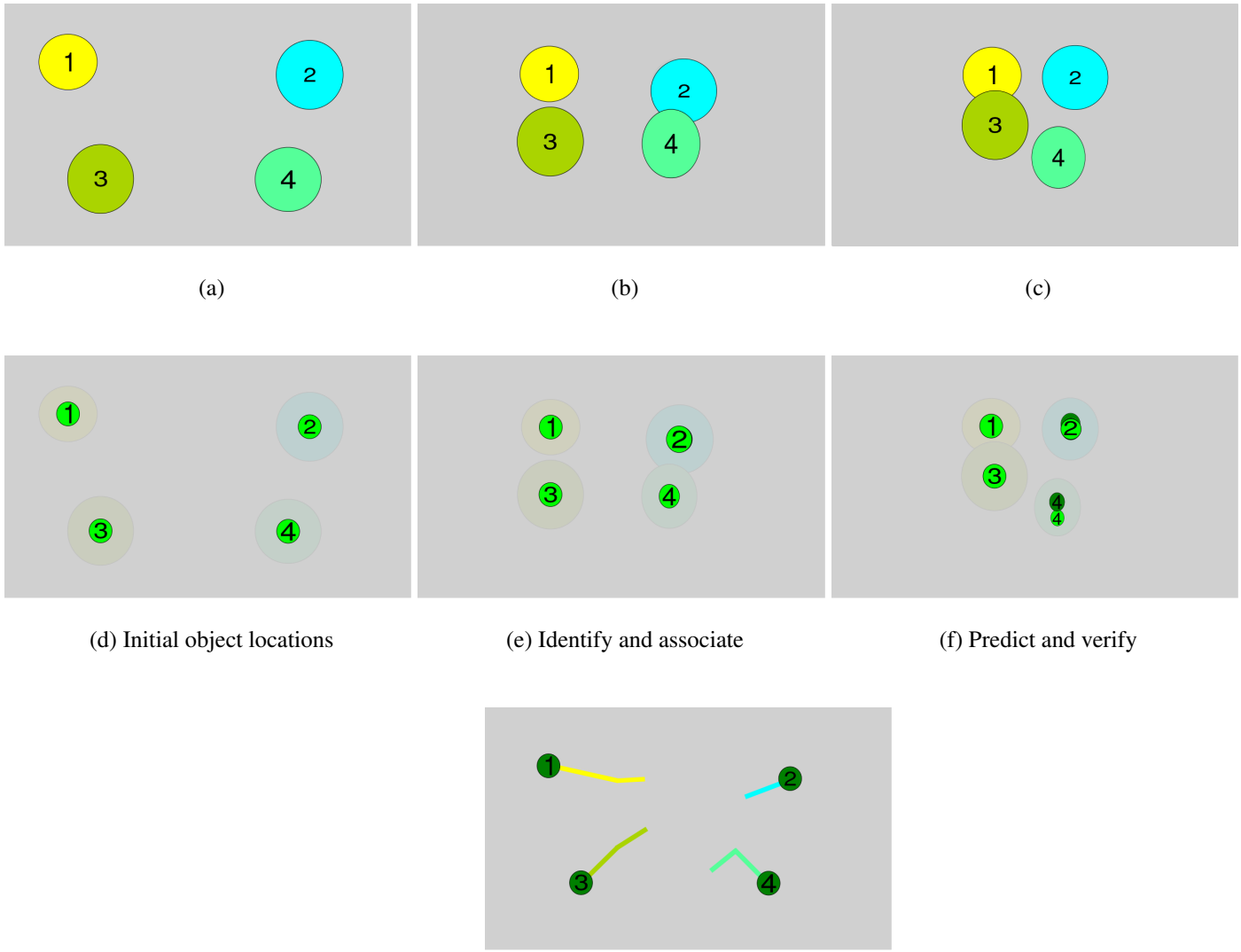


Figure 1.4: Generic Object Tracker



(a) (b) (c)

(d) Initial object locations (e) Identify and associate (f) Predict and verify

(g) Object Tracks

Figure 1.5: Image sequence to illustrate object tracking and ground truth.

1.3 Research Context

Current research into pedestrian detection/tracking as stated previously, is a very active area. A large number of pedestrian detection and tracking algorithms have been proposed, evaluated and found to the effective at both accurately detecting and/or tracking pedestrians with both infrared and visual image sequences. The use of these algorithms with infrared images sequences is a recent development, after the year 2000 when thermal infrared cameras became more accessible (cheaper).

This reduction in the cost of infrared cameras has also increased the research into the information recorded in an infrared image. Results from these studies has shown that texture information in the form of emissivity, surface texture along with the temperature of the object are present in an infrared image.

Some initial research into pedestrian tracking in infrared image sequences used techniques previously demonstrated as being effective with visual image sequences. A large body of work with pedestrian detection in infrared images has been done to improve pedestrian safety this has led to the development of pedestrian classifiers that are effective while utilising the textural properties of pedestrians in infrared images. This presents an opportunity to develop a static pedestrian tracking system utilising infrared image sequences.

1.4 Problem Areas

The problem areas identified and addressed in this document are in the following domains:

Pedestrian detection and tracking metrics: As stated above, a large number of pedestrian detection and tracking systems have been proposed and found to be effective at identifying and tracking pedestrians in both infrared and visual image sequences. Existing metrics for accuracy are thorough and effective. Metrics for the speed of the detection/tracking systems on the other hand tend to be ad-hoc, usually, the frame rate on *'a desktop class system'*. This measure however, is unreliable and will not necessarily scale with changes in the number of targets or in the underlying hardware used.

Pedestrian detection and tracking algorithms: Most pedestrian detection and tracking systems used with infrared image sequences, tend to assume that the pedestrian target will be brighter than the background (i.e. the background is at a lower temperature). While this is usually true, cluttered urban environments tend to have heat sources which are brighter or at the same temperatures as pedestrians (cars, vents etc.) Intensity based systems may not be as effective as indicated in literature with image sequences recorded in these environments.

The research questions outlined below are formulated to address some of the issues identified above.

1.5 Research Questions and Contribution to Knowledge

In the previous section we identified three subject domains related to pedestrian detection and tracking with infrared videos where there are some deficiencies in the existing body of knowledge. The following research questions will address some of the issues identified and hence contribute to the body of knowledge about pedestrian tracking in infrared image sequences. The research questions are:

1.5.1 Research Questions

The research questions that will be investigated in this thesis are as follows:

1. Is using the time taken to process a single target an effective metric to predict the performance of a tracking system for a specified target (pedestrian) density?

2. Is a simplified matrix pedestrian aspect ratio classifier effective and accurate at classifying pedestrians in infrared image sequences?

3. How effective is a codebook tracker at tracking pedestrian targets in infrared image sequences?

1.5.2 Novelty and Contributions

The motivation behind the research presented in this document is to develop a low compute cost algorithm for pedestrian detection and tracking. The novel aspect-ratio based pedestrian detection method is shown to be robust and effective at identifying pedestrians in infrared image sequences. Building on this detection mechanism, a novel deformable codebook based tracker is outlined and tested with an infrared image sequence dataset. The combination of the novel detection mechanism and tracker is then demonstrated as being as effective as state of the art, while requiring fewer compute resources.

Speed is used as a surrogate for the compute resources required by a given pedestrian classifier/ tracker combination when processing an image sequence from the dataset. This is validated by comparing the performance of the new algorithm with three different pedestrian classifiers and an object tracker previously described in literature. The performance data is generated using a dataset consisting of 45 sequences with different pedestrian densities.

1.6 Summary and Document Outline

This chapter has served as an introduction to infrared images, object classification and tracking and associated metrics. This basic introduction served as background to help understand the research context and the research questions proposed above. However, to make an informed decision on the novelty of the proposed pedestrian classifier and tracker a better understanding of existing literature is required.

The next chapter presents a detailed literature review to help with this. After the literature review, the following chapters present, in order; the imaging system and dataset used to evaluate the performance of the pedestrian classifier and tracker, metrics of existing pedestrian tracking systems, the novel pedestrian classifier and tracker and the system metrics of the same. Finally, a roadmap for further work based on the research in this document is presented; this also serves as a summary for the document.

Chapter 2

Literature Review

The motivation for research presented in this document, as stated previously, was the development of a robust low-compute cost pedestrian tracking system for use with thermal infrared image sequences. Figure 2.1 is a taxonomy for pedestrian tracking systems derived from one used by Wang, Hu and Tan in their 2003 [6] survey of developments in human motion analysis.

Reviewing the entire body of knowledge that applies to pedestrian tracking systems, is impractical, so the detailed literature review is constrained to exemplar algorithms from the same genus as the the proposed algorithms. These areas are highlighted in red in figure 2.1. The genus in highlighted in green consists of algorithms that use alternative approaches to the ones used by the proposed algorithms.

This chapter is organised as follows; first, the use of infrared image sequences in pedestrian tracking is reviewed, this is followed by a review of background models by means of two pixel background models. This is followed by a review pedestrian classifiers, one template based approach and one an appearance based approach. After this object trackers are reviewed, using three exemplars. Finally the metrics used to evaluate the performance of pedestrian tracking systems are reviewed and their relation to the accuracy and precision.

2.1 Use of Infrared Images in Pedestrian Tracking

The research context (section 1.3) stated that infrared cameras became more accessible after the year 2000, anecdotal evidence for this can be found in a google scholar search for the term “infrared pedestrian tracking”. When the results are grouped by publication year; upto the year 2000 there are 1,060 articles, 1,710 between 2001 and 2005, 3,780 between 2006 and 2010 and 1,270 articles after 2010.

A comparison study between the use of infrared and visible images for pedestrian classification published by Fang et al [7] in 2003 identified the following properties of pedestrians in infrared images:

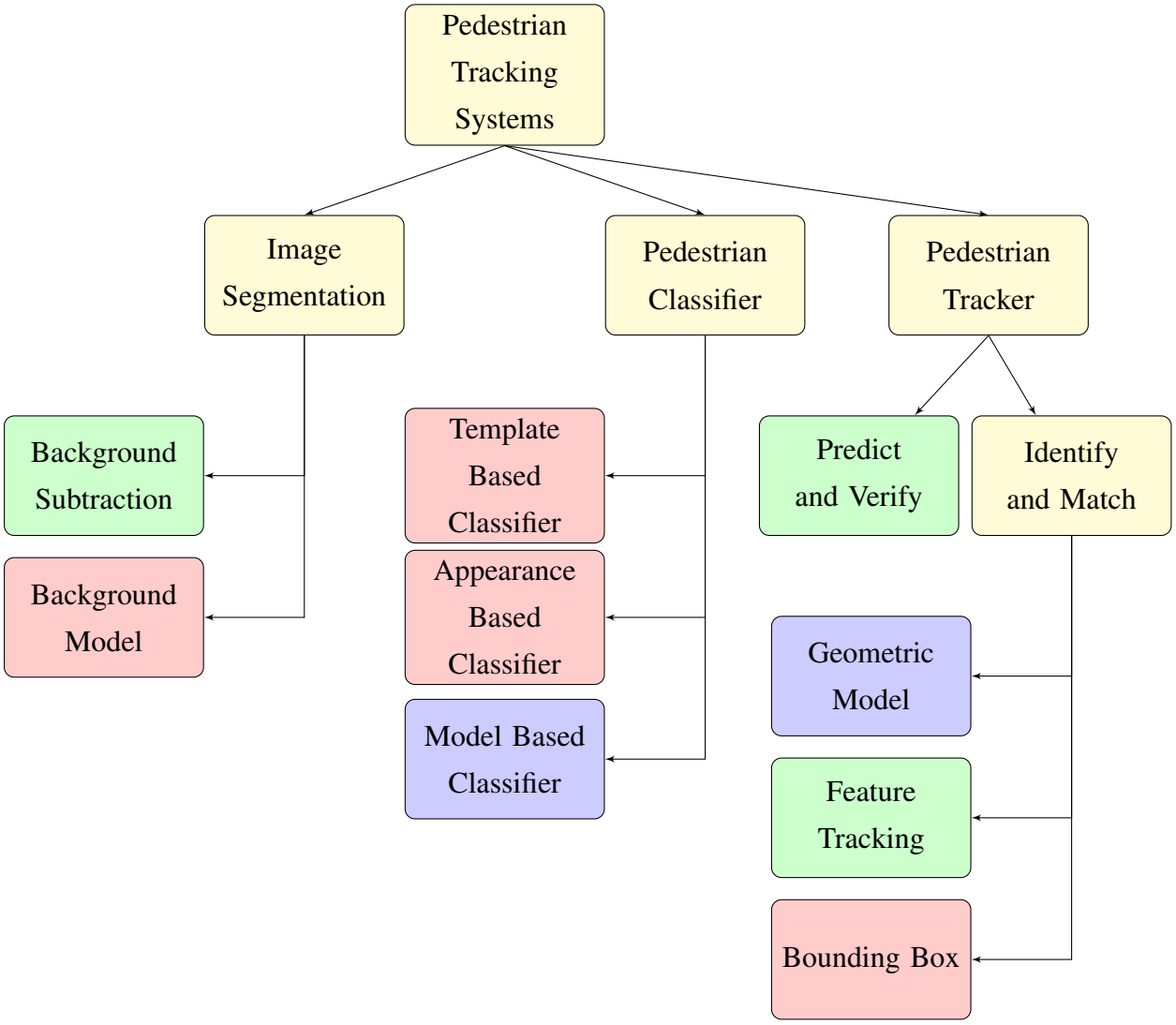


Figure 2.1: Taxonomy for Pedestrian Tracking Systems

- Pedestrians are usually warmer than the background objects, which makes them brighter than the background [8, 9, 10, 11].
- Human body temperatures are usually in a narrow range, therefore the brightness of pedestrians in different images should be similar despite different clothing [7].
- Again due to the uniform human body temperature, pedestrian targets will have uniform intensity [7].
- Pixel intensities in images are usually clustered into narrow bands, which makes identifying potential regions of interest in images simpler [7].

Pedestrian classifiers and trackers using visual images often utilise surface texture to discriminate between foreground regions, this has led to the development of multiple pixel colour models [12]. The goal of most of the visual colour models is to separate the *chroma* (colour) information for a pixel from its intensity. Isolating the pixel colour from its intensity mitigates the problems caused by illumination changes both global and local (clouding, shadows etc.).

Comparative studies of texture content of infrared and visible images, have demonstrated the presence of surface texture in infrared images [4]. This same surface texture content is insufficient to permit the use of techniques that use pedestrian texture and appearance such as 2D/3D geometric pedestrian models for pedestrian classification and tracking in infrared images [7]. Further evidence for this was published by Broggi et al. in 2004 [10], who as part of a different group, had previously investigated the use of 3D templates for pedestrian validation after using a pedestrian symmetry classifier [13].

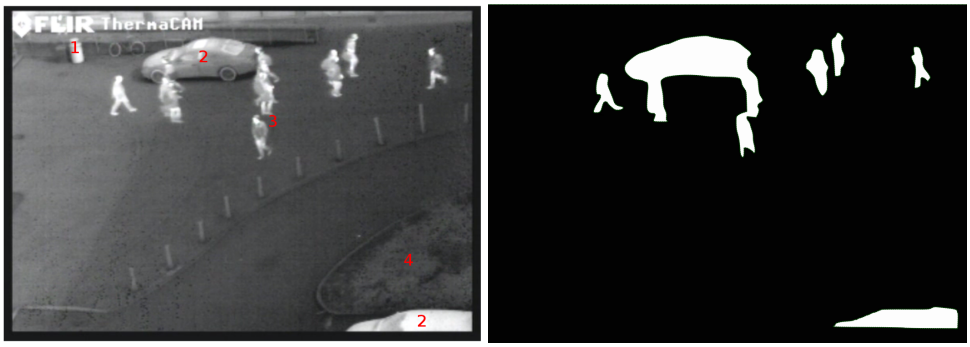
The above reasons are used to exclude visible geometric model based classifiers and trackers from this review, highlighted in blue in figure 2.1. With this background the next section will review background models and the reasons for their use.

2.2 Background Models and Image Segmentation

Image segmentation is the process by which extraneous background pixel information is removed from the image sequence to reduce the search space for the pedestrian classifier or tracker. Kim et al. [14] use the term *'salient motion'* to describe the information that is not removed.

This is in reference to the fact that for pedestrian tracking applications, not only does the background model need to be able to discriminate between foreground and background objects. It also needs to be able to discriminate between movement by objects in the foreground and movement by background objects due to environmental factors. Additionally it also must be capable of identifying apparent movement caused by shadows and changes in illumination etc.

In figure 2.2(a), we see a typical infrared image from a sequence used for pedestrian detection and tracking. This image shows a range of objects, both from the background and the foreground; Static, warm background object *(1)*, Non-pedestrian foreground targets *(2)*, Pedestrian foreground targets *(3)* and Background objects with a limited range of motion *(4)*. Figure 2.2(b) is the segmented image equivalent of figure 2.2(a).



(a) Infrared Image (b) Infrared image after segmentation

Figure 2.2: Sample Infrared Image

Most background subtraction algorithms used with surveillance cameras employ some form temporal differencing (Figure 2.3) [15] to segment the background from the image [16]. The simplest form is when a static background image is subtracted from the current image in the sequence.

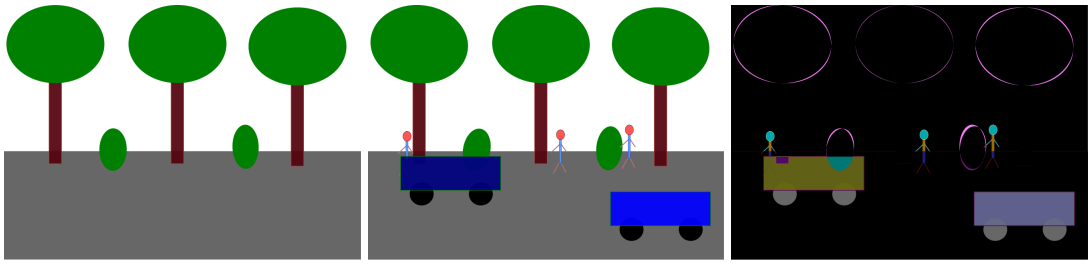
This is illustrated in figure 2.3 where the foreground regions (figure 2.3(c)) in the current image $X_{x,y}$ (figure 2.3(b)) are identified by subtracting the background image $B_{x,y}$ (2.3(a)):

$$I_{x,y} = |X_{x,y} - B_{x,y}| \tag{2.1}$$

Initial background subtraction algorithms relied on manual initialisation to identify the background images which would then be used as a background model. However, these were found to be unreliable as the background being modelled was dynamic with changes in illumination, shadows [17], distortion [12], background object motion/relocation [14, 18] etc.

2.2.1 Adaptive models and Update Mechanisms

Adaptive background models [19, 14, 20, 12, 21, 22, 23], were developed to overcome the problems associated with static background models. The background model is generated from either a static initial image without any foreground objects [19, 24]; or, from a set of training images, some of which may contain some foreground objects [14, 22, 25, 26, 21, 27, 28, 29]. After the initial background model is generated, these models are then updated periodically. The update can be global, regional or per-pixel [21] or in some cases the algorithm may use all three [20].



(a) Background image

(b) Foreground image after time t

(c) Difference between image at t and background image

Figure 2.3: Illustration of Temporal Differencing

2.2.1.1 Global Background Update

These algorithms update the entire background model periodically [30, 31, 32, 20]. One example of this, by Oliver, Rosario and Pentland [30] models the background as an eigenvector matrix of the training images. To classify the pixels in the current image, the image is projected onto the eigenspace which is a good model for the static pixels and not for the pixels that may have changed, the difference between the eigenspace image and the current image will represent foreground regions. In summary;

Training:

if the training set consists of n images with p pixels

the average image μ_n is then subtracted from the training set n to produce a new set of images n'.

the covariance matrix for n' is then computed, and the best M vectors are stored in an eigenvector matrix Φ_{Mn} of size $M \times p$.

Classification:

Every new image I is projected onto the eigenspace: $I' = \Phi_{Mn}(I - \mu_n)$

I' is then used to identify the static points as $I'' = \Phi_{Mn}^T I' + \mu_n$

as I'' will not contain any small moving objects, foreground objects are to be found when $|I - I''| > T$

Empirically derived values and updates:

For the above algorithm, the number of training image n, the number of eigenvector to be used M and the threshold T need to be empirically derived and set.

There is no explicitly defined background update mechanism.

2.2.1.2 Regional Background Update

The background model in these algorithms is assumed to be made of connected pixel blocks of fixed arbitrary sizes and are classified as belonging to the foreground or background [33, 34, 32, 20]. These algorithms are based on the assumption that any changes that affect one pixel in an image will also affect the neighbouring pixels.

The update mechanism used by the Wallflower background model by Toyama et al [20] a good example of region level background update mechanism. In their implementation, to avoid erosion of homogeneously coloured objects, neighbouring pixels of new foreground pixels were evaluated to check if they belong to a foreground object. The individual foreground pixels were identified by a Wiener filter which using the most recent colour and intensity values of the pixel.

The regional update is identified by computing a normalised histogram for any region found to contain a foreground pixel. The region is defined by the new foreground pixel and its four neighbouring pixels. If the difference between the normalised histogram and the background is above a set threshold then the entire region was classed as foreground, else, the region is suppressed into the background. They found that while using a regional update mechanism to identify contiguous regions reduced target erosion, the problems introduced in the form of erroneous foreground suppression outweighed the benefits of implementing the same.

2.2.1.3 Single Pixel Background Update

The background model is assumed to be a collection of independent pixels, every pixel is independently classified. The neighbouring pixels do not affect how a pixel is classified [14, 22, 25, 26, 35]. These techniques have been found to most responsive and accurate when classifying pixels hence there is a large body of work that has gone into benchmarking the performance of these algorithms, in response to global illumination changes [19, 26, 36, 37], shadows [14, 22, 32] background object motion [14]. Due to these reasons, the tracking system implemented using the proposed algorithms use a single pixel background model. Hence, two examples will be described and evaluated in detail.

2.2.2 Pixel Level Image Segmentation

As identified previously, single pixel level background modelling and subtraction algorithms have been found to be very effective when classifying foreground and background regions. Hence, there is a large body of work in relation to this. As it would be impractical to examine all the algorithms that have been found to be effective, this literature review will examine select algorithms that operate in the Red Green Blue (RGB) colourspace.

Initial single pixel algorithms used predictive filters [20, 27] to segment the images, while these predictive models were effective indoors and in other controlled environ-

ments. They were ineffective in situations where the environmental changes were more unpredictable. Statistical distributions [19, 27, 22, 26] on the other hand were found to be effective when modelling the changes in pixel colour and intensity values, hence require a more in depth examination.

2.2.2.1 Mixture of Gaussians (MOG) Background Model

This was one of the first single pixel distribution background models to be implemented and tested. The initial implementations modelled the pixel as a single gaussian distribution and was used with grey-scale images from visual cameras [19]. To improve the accuracy and help the classifier effectively compensate for illumination changes and shadows, the pixel model was modified to use multiple gaussian distributions [27].

MOG background modelling algorithms use the property of gaussian distributions where 95% of the distribution values lie within 3 standard deviations of the mean. For visual images, where each pixel has three (Red Green Blue) or four (RGB + Intensity) colour channels. Pixel classification occurs as follows:

- Every pixel is modelled as a set of gaussian distributions, initially as one distribution.
- For all pixel values within threshold T, standard deviations the pixel remains are part of the background and the probability distribution modelling the pixel updated accordingly.
- When a new pixel value, at time t is outside the T standard deviations. The pixel is classed as part of the foreground and a new distribution is recorded. Any subsequent matches to this distribution are classed as background matches.
- Periodically, the number of gaussian distributions for a pixel are compacted by removing the ones not used for a set period of time.
- The accuracy of the classifier is constrained by the following; a learning rate β and the proportion of the data that should be accounted for by the background B.

The implementation as described in literature [27]:

The colour of a pixel in the RGB colour space is represented by a vector,

$$P_C = (P_R, P_G, P_B) \text{ and at frame } f, \text{ the } i^{th} \text{ pixel is recorded as,} \tag{2.2}$$

$$X_{i,f} = (R_{i,f}, G_{i,f}, B_{i,f}) \tag{2.3}$$

$$\text{If a pixel is modelled as } k \text{ gaussians, where } k : 3 \leq k \leq 5 \tag{2.4}$$

the probability of the i^{th} pixel being part of any existing gaussian is modelled by:

$$Pr(X_{i,f}|X_{i,1}, X_{i,2}, \ldots, X_{i,f-1}) = \sum_{j=1}^{k} \omega_{i,f-1,k} * \eta(X_{i,f}, \mu_{i,f-1,f}, O_{i,f-1,k}) \text{ where,} \tag{2.5}$$

ω, is the weight for the j^{th} distribution

η, is a probablity density function

μ, is the mean for the j^{th} distribution and

O, the co-variance matrix for the j^{th} distribution

The information of interest here isn't whether the expected value $P(X_{i,f})$ is the same as the recorded value $X_{i,f}$, instead its how effective the existing distributions are at modelling the observed state of the pixel.

This is done by identifying the distribution that *'matches'* the current value for the pixel. In this case for the current value $X_{i,f}$ to match one of the existing k distributions its value must be within a set threshold T. Empirically, T has been found to be effective when it is set at 2.5 standard deviations from the mean for distribution k. Any pixel in the current frame f not matching any of the k distributions for i from frame $f-1$ is considered as a new/unique value and be classified as part of the foreground.

The weight in Equation 2.5 ω is a relative measure of the proportion of the observed pixels $X_{i,1}, X_{i,2}, \ldots, X_{i,f}$ that are part of distribution j. these weights are adjusted after classifying the current pixel for use with the next recorded pixel. This is done as follows:

$$\omega_{i,f,k} = (1-\beta)\omega_{i,f-1,k} + \beta M_{i,f,k} \tag{2.6}$$

Where, β is the learning rate that controls how quickly the model adapts to changes in the background. For any pixels that do not match existing distributions, the distribution with the lowest weight for that pixel is replaced with a new distribution with the mean $X_{i,f}$, a high variance σ with a low weight ω.

Any pixels matching existing distributions, the mean and variance for that pixel are updated as follows:

$$\mu_f = (1-p)\mu_{f-1} + pX_f \tag{2.7}$$

$$\sigma_f = (1-p)\sigma^2_{f-1} + p(X_f - \mu_f)^B.(X_f - \mu_f) \tag{2.8}$$

Where, $p = \beta\eta(X_f|\mu_k, \sigma_k)$ and

B is the proportion of the distribution that should be part of the background.

With this information we can now define the probability distribution function η as

$$\eta(X_{i,f}, \mu_{i,f-1,k}, O_{i,f-1,k}) = \frac{1}{2\pi^{\frac{n}{2}}|O|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_f-\mu_f)^B O^{-1}(X_f-\mu_f)} \tag{2.9}$$

After the model has been trained, the pixel distributions are ordered according to the weight variance ratio ω/σ ratio per pixel.

This ordering of the distributions ensures that the one that is most likely to match the expected pixel value $X_{i,f}$ will be tested first. As the pixel values change, the variance of

the distribution that most closely models this will decrease increasing the chances that it will persist and not be replaced (Equation 2.6).

From equations 2.5, 2.6, 2.7 and 2.8 we can see that this model has several static user defined parameters that are critical to how the model responds to changes in the environment. Detailed studies on how these parameters can be optimised for different environments [38] have been published. This background model has been found to be effective and about 80% accurate [14] when identifying foreground regions in colour video.

The drawbacks inherent to this background model apply to the static parameters that control the rate at which the model adapts (the proportion of background B and learning rate β). While the rest of the model adapts to changes in the environment, these parameters do not adapt, causing the background model to fail when the learning rate is too low for the changes in the scene.

However one weakness inherent in the model can be seen in Equation 2.6, where if a pixel X_i has been static and not part of the foreground for a long time then variance μ becomes very low and the pixel classification (2.5) becomes unstable.

One method for mitigating this is by introducing a new static parameter σ_{min} as the absolute variance. This means that equation 2.8 is modified as:

$$\sigma_n = \begin{cases} (1-p)\sigma^2_{f-1} + p(X_n - \mu_n)^B.(X_n - \mu_n) & \text{if, } \sigma_n \geq \sigma_{min} \\ \sigma_{min}, \text{ otherwise.} \end{cases} \tag{2.10}$$

To summarise, modelling the background as a mixture of gaussians is effective in segmenting the background from the foreground. While the Threshold T used to classify pixels is universal to all image segmentation algorithms. The other user defined parameters background proportion B, learning rate β, number of distributions k increase the failure scenarios when the background model is suddenly faced with multiple changes; e.g. a global illumination change while there are a lot of foreground targets. The presence of the foreground targets during what then becomes the retraining sequence will severely reduce accuracy till the model has adapted to the changes.

2.2.2.2 Codebook Background Model

Unlike the previous background modelling algorithm, the algorithm that we will now examine models the background as a cache of colour values [39]. The pixel classification process can be described as follows:

- Every pixel is modelled as a set of codewords in a codebook. The code words for a pixel contain its colour information (RGB) and two intensity values (i_{min} and i_{max}).
- Pixel classification is based on a colour value and intensity range match to a codeword. If the colour matches and the intensity is within a set threshold of either i_{min} or i_{max} then the pixel is considered a match to the current codeword for that pixel.

- Matched codewords are updated with new intensity and colour values information, unmatched pixels are classed as foreground and new codewords added for that pixel.
- Periodically codewords lists for pixels are compacted by removing codewords that have not been used for some time.

For a pixel i with four colour channels (RGB + Intensity) in frame n in a sequence X of images after the background model has been trained, the classification process can be described as:

$$Bg = \begin{cases} 1, X_{i,n} : X_{i,n} \in CB_i \\ 0, \text{otherwise} \end{cases} \tag{2.11}$$

Where, CB_i is a set of code words that describe the colour history of the pixel:

$$CW_{i,j} = \{Pc_{i,j}, aux_{i,j}\} \tag{2.12}$$

$Pc_{i,j}$ is the colour vector in RGB colour space and (2.13)

$$aux_{i,j} = \{\hat{\imath}_{i,j}, \check{\imath}_{i,j}, f_{i,j}, \lambda_{i,j}, r_{i,j}, \alpha_{i,j}\} \tag{2.14}$$

$$j = j : j \in I, j > 0 \tag{2.15}$$

In equation 2.14

$\hat{\imath}_{i,j}$ is the minimum intensity for the codeword,

$\check{\imath}_{i,j}$ is the maximum intensity for the codeword

$f_{i,j}$ the frequency for the codeword

$\lambda_{i,j}$ is the **M**aximum **N**egative **R**un **L**ength

$r_{i,j}$ is the first frame at which that codeword occurred

$\alpha_{i,j}$ the last frame at which that codeword occurred

By separating the intensity from the colour of the pixel and using a minimum and maximum pixel intensity threshold, prevents local luminance changes from affecting the accuracy of the match. It also permits a single codeword from matching a pixel under varying illumination conditions.

To identify global illumination changes and prevent it from affecting classification decisions the pixels are always classified after compensating for colour distortion ϕ between the current pixel colour $X_{i,n}$ and $Pc_{i,j}$. Colour distortion is the change in pixel RGB values that occurs when there are changes to the luminance in the scene. The following matching

technique ensures that only the chroma values are used to match the colour of the current pixel to the codeword.

$$\phi(X_{i,j}, Pc_{i,j}) = \sqrt{||X_{i,n}||^2 - \xi^2} \tag{2.16}$$

$$||X_{i,n}||^2 = R_{i,n}^2 + G_{i,n}^2 + B_{i,n}^2 \tag{2.17}$$

$$\xi^2 = ||X_{i,n}||^2 \cos^2\theta = \frac{(X_{i,n}, v_{i,n})^2}{||v_{i,j}||^2} \tag{2.18}$$

$$||v_{i,j}||^2 = ||Pc_{i,j}||^2 = R_{i,j}^2 + G_{i,j}^2 + B_{i,j}^2 \tag{2.19}$$

$$(X_{i,n}, v_{i,j})^2 = (R_{i,n}R_{i,j} + G_{i,n}G_{i,j} + B_{i,n}B_{i,j})^2 \tag{2.20}$$

Because pixel colour is constant once changes in intensity has been compensated for [37] the classification of pixel can be rewritten as:

$$X_{i,n} \in CB_i :: [\phi(X_{i,n}, Pc_{i,j}) < T_c]_1^j \tag{2.21}$$

Equation 2.21 shows that the model uses a threshold value to classify pixels like the previous background model. The classification mechanism also controls how the background model updates. When a new pixel value that does not match its codebook is encountered, it is added as a new codeword to the at $j+1$ to the codebook for that pixel. The colour information $P_{i,j+1}$ and $aux_{i,j+1}$ (equation 2.12) for the new codeword is initialised using the following values

$$Pc_{i,j+1} = \left(\frac{f_{i,j}R_{i,j} + R_{i,n}}{f_{i,j}+1}, \frac{f_{i,j}G_{i,j} + G_{i,n}}{f_{i,j}+1}, \frac{f_{i,j}B_{i,j} + B_{i,n}}{f_{i,j}+1} \right) \tag{2.22}$$

$$aux_{i,j+1} = \{\hat{i}_{i,j+1}, \check{i}_{i,j+1}, f_{i,j+1}, \lambda_{i,j+1}, r_{i,j+1}, \alpha_{i,j+1}\} \tag{2.23}$$

$$\hat{i}_{i,j+1} = argmax(I(X_{i,n}, \hat{i}_{i,j}) \tag{2.24}$$

$$\hat{i}_{i,j+1} = argmin(I(X_{i,n}, \hat{i}_{i,j}) \tag{2.25}$$

$$\lambda_{i,j+1} = \max(\lambda_{i,j}, n - \alpha_{i,j}), \tag{2.26}$$

$$r_{i,j+1} = \alpha_{i,j+1} = nf_{i,j+1} = 1 \tag{2.27}$$

$$\tag{2.28}$$

If codewords continue to be accumulate for a pixel, in time the background model will contain a lot of codewords and parsing the chain for a match will become impossible. Equation 2.14 identifies one property that can be used to remove transient codewords.

$$CW_{i,j} \notin CB_i \text{ if, } \lambda_{i,j} > 100 \tag{2.29}$$

In the above equation (2.29), the λ threshold 100 is an empirically derived value and is increased for environments where the random background motion is low.

From the classification technique described in equation 2.11 we can see that the classification mechanism is not compute intensive, it is instead memory intensive. Additionally, this background modelling algorithm unlike the previous modelling algorithm

(section 2.2.2.1; B, β and T) has only two parameters (T_c and λ). Only one of which (T_c in 2.11) has a significant impact on the algorithms performance. However, this algorithm has one significant weakness, in that when used as a model for an image stream with a large amount of background motion, the memory footprint of the codebook could become significant.

2.2.3 Performance Metrics

Earlier we examined, in detail, two implementations of background modelling algorithms (sections 2.2.2.1 and 2.2.2.2), from the implementations and literature we can now look at various metrics used to evaluate the performance of the algorithms and how the parameters could be adjusted to improve performance.

The most effective metrics used to evaluate the performance of a background modelling algorithm are the False positive (FP) and false negative (FN) rates. A false positive match occurs when a background pixel is misclassified as part of the foreground. Similarly, a false negative match occurs when a foreground pixel is classified as part of the background. The frame rate and the memory footprint of the model are other metrics that are commonly used.

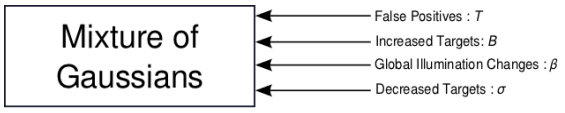
When a background model has a high FP rate, spurious transient targets are presented to pedestrian detection and tracking algorithms. This increases the computing time taken by those modules and may increase false tracks. Large areas of false positive matches are usually caused by global changes in the image stream, by say, a cloud shadowing the region under observation or high winds increasing the magnitude of random background motion.

Conversely, if a background model has a high FN rate then, the regions that are presented as part of the foreground are eroded and some valid targets are completely masked. Fragmented foreground regions may be misclassified as non-pedestrian targets and consequently not tracked or in some cases the feedback mechanism could validate these as segmentation errors and modify the operating parameters to remove the fragmentary targets, exacerbating the loss of tracking information. FN problems usually occur when the classification threshold is set too high or in situations when the background model adapts too quickly (learning rate too high). One of the most challenging image streams for a background modelling algorithm is when the incoming image stream consists of large numbers of similar foreground regions.

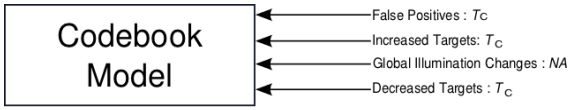
While FN and FP [21, 9, 40] metrics have been used to evaluate a background model performance [19, 22, 14, 12], the frame rate and memory footprint metrics are more subjective and the statistics quoted for one publication may not hold true in all cases.

This uncertainty is reduced when an article implements all the background models on the same or similar hardware and evaluates the performance as illustrated by Chalidabhongse et al.[12]. Using the above described metrics, we can look at how the feedback

mechanisms could modify the performance of the background model at runtime.



(a) Mixture of Gaussians



(b) Codebook Model

Figure 2.4: Feedback Model Control Mechanisms

Mixture of Gaussians : In figure 2.4(a) we can see how the parameters might be changed in response to changes in the environment. If, for example, the number of FP targets increased then the T value could be increased. If the number of foreground targets increased; then the proportion of the image that is part of the background B needs to be decreased, conversely if the number of targets decreases then the variance σ_{min} needs to be increased so that the static background does not make the model unstable. Additionally, if there are lots of rapid global illumination changes, the learning rate β needs to be adjusted to ensure that the colour distortion due to this is accounted for in the background model.

Codebook Background Model Similarly figure 2.4(b), shows the parameters that are modified when modelling the background using the codebook algorithm. As identified previously in section 2.2.2.2, this background model is very versatile and thus the only parameter that needs to be adjusted is the threshold value T_c. This is done in response change in target density as well as in scenarios where the background model is generating FP matches. Whilst this makes the feedback mechanism simpler, this also means that the model cannot be fine-tuned in response to one specific change in the environment.

As we have an understanding of how background modelling algorithms work and have examined two implementations of the same in detail, we can now move onto the next module in a pedestrian tracking system, the pedestrian classifier (figure 2.1). This module takes the foreground regions that have been identified by the background model and processes them to identify any regions that may contain valid pedestrian targets. A variety of techniques are used to achieve this, and the following section examines these techniques in detail.

2.3 Pedestrian Classifiers

In chapter 1, section 1.2.2 and figure 1.2 introduced the functionality of a generic object classifier. The classifier taxonomy in figure 2.1 categorises pedestrian classifiers into the following types:

Template Based : These classifiers attempt to match a pedestrian template, i.e. an exemplar of a valid pedestrian target to the foreground regions. If this match succeeds, then the foreground region is classified as a valid pedestrian target. This type of classifier has successfully been used with infrared image sequences by Dai et al [41], Nanda and Davis [8] Olmeda et al [42] etc.

Appearance Based : These pedestrian classification algorithms, use the distinct appearance of a pedestrian target to identify and localise pedestrians in foreground regions. With infrared images, the most commonly used pedestrian appearance classifiers are symmetry [11, 10, 7] and aspect ratio [13, 7, 6]. Usually, both the appearance metrics are used together to help improve classification accuracy.

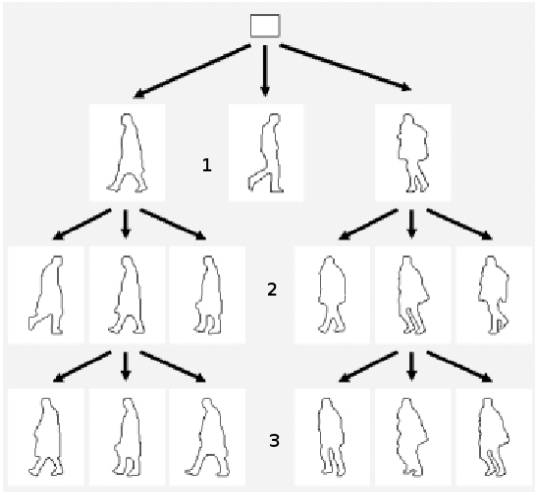
Model Based : Model based classifiers, localise pedestrians in foreground regions using models of pedestrians. Unlike the template based approaches, a exemplar is not used instead physical features are matched to a model. The models used could be 2 Dimensional or 3 Dimensional, these methods have been found to be very effective in classifying and tracking pedestrians in visual image sequences [32]. As identified in section 2.1, infrared images lack texture information for these techniques to be effective. Due to this reason, model based classifiers are not reviewed in detail in this document.

2.3.1 Template Based Pedestrian Classifiers

Most implementations of template based classifiers operate on the image stream directly without background subtraction. As it is impossible to generate a template set that can match all possible pedestrian targets template based classifiers have higher failure rates.

Multiple template based classifiers have been used with infrared images for pedestrian detection; for example, the probabilistic template classifier by Nanda and Davis [8]. This has been improved by Olmeda, de la Escalera and Armingol [42] where they use the probabilistic template to detect pedestrians in infrared images and track the targets using a kalman filter.

Another template technique that has been successfully used with IR images is a coarse-to-fine hierarchical matching algorithm [43] (figure 2.5(a)), this algorithm tests the foreground regions with a three level clustered template tree where the foreground regions are tested against the tree and are classified as valid tracking targets only if they match either the second or third level of the tree.



(a)

Figure 2.5: Hierarchical Templates

2.3.1.1 Probabilistic Template

Probabilistic template tracking was first described by Nanda [8] in 2002 for pedestrian detection. Subsequently, it has also been implemented by Olmeda [42] with a Kalman Filter as the tracking module. The template is generated from a set of training images which contain pedestrian targets and background regions. The assumption for the targets in the training set is that the targets are warmer than the background. The algorithm for probabilistic template classification can be described as follows:

Training :

- Using the training image pixel intensities, identify the mean and standard deviations for the pixels in the foreground and the background.
- The mean and standard deviations for the two distributions generate the training threshold value.
- Threshold the training images with the training threshold to generate the binary equivalent of the training image.
- Resize and rescale the training images so that the geometric center of the foreground target in the training image is in the geometric center of the window.
- Using all the resized training images, for every pixel in the window calculate the probability of its being part of the foreground. If a pixel in a training image is 1 then it increases the probability that it is part of the template. If 0, it reduces the probability that the pixel is part of the template.
- Using the above probability values, a combined probability map for the training images is generated. Using a separate set of images without pedestrians, a

combined probability map for sample images without pedestrians is similarly generated.

- Modelling the pixels in the combined probability maps as distributions, the mean and sigma for the two sets of images is calculated (training images with pedestrians and training images without pedestrians).
- The distributions are then used to generate a classification threshold value that is used to identify pedestrians in images.

Classification :

- To classify pedestrians in an image, the image is scanned using a sliding 128×48 pixel window. As the image is scanned, combined probability maps are generated, centered around the pixels.
- The combined probability maps for the image are thresholded using the classification threshold. Any pixels that are above the classification threshold are labelled as containing a pedestrian.

If the image in figure 2.2(a) is used to generate a training set, then the possible training images that could be extracted are show below (figure 2.6).

To generate the template, using the training images the pixel distributions are calculated. One is used to represent the pedestrian targets (μ_{pf}, σ_{pf}), the second to represent the background (μ_{pb}, σ_{pb}) in the training set. Assuming that both the distributions have equal priors, the classification threshold:

$$threshold = \frac{\sigma_f \sigma_b}{\sigma_f + \sigma_b} \ln\left(\frac{\sigma_f}{\sigma_b}\right) + \frac{\sigma_f \mu_b + \sigma_b \mu_f}{\sigma_f + \sigma_b} \tag{2.30}$$

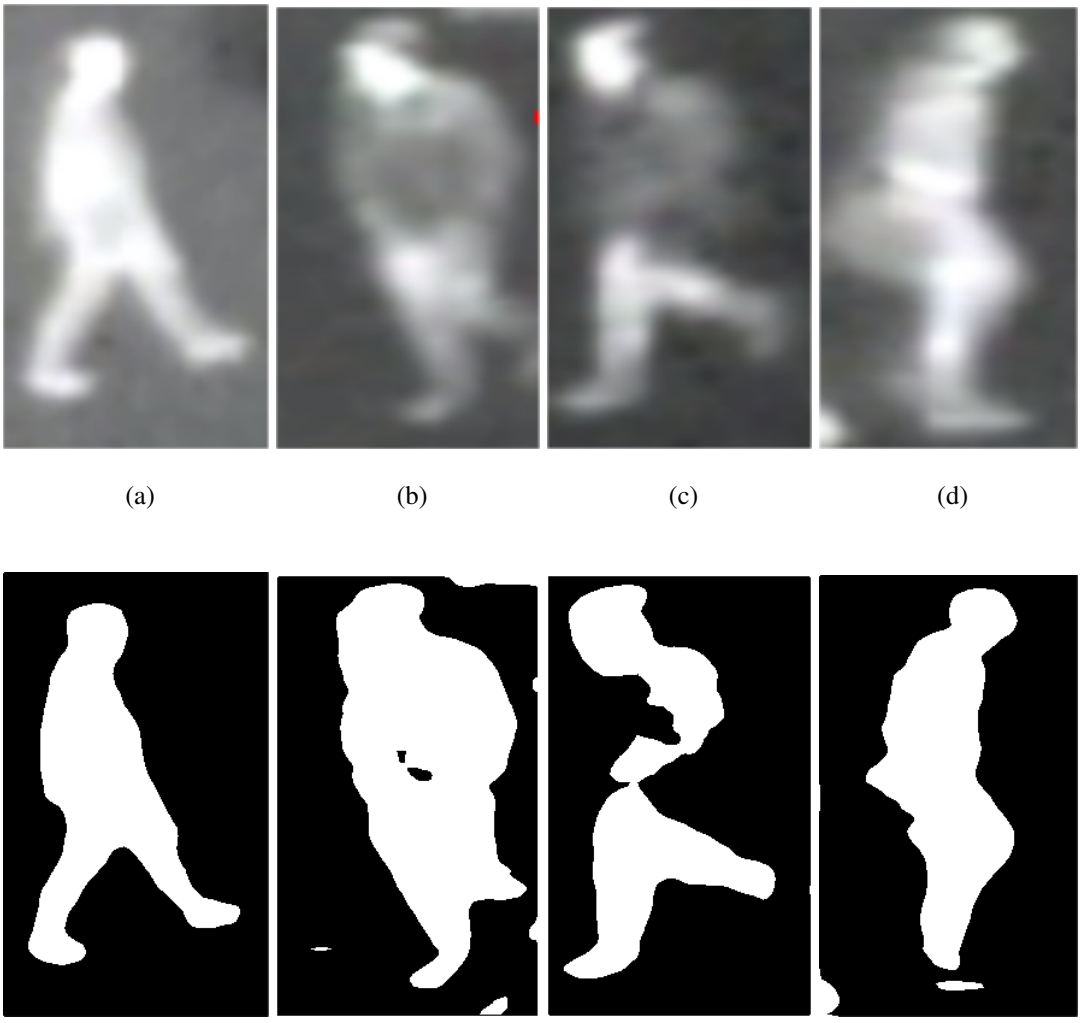
The threshold described in equation 2.30, is used in the original publication by Nanda and Davis in [8] it is an empirically developed threshold that has been found to be effective at thresholding infrared images.

Applying the result of equation 2.30 to the training images in figure 2.6(a) to 2.6(d) the thresholded training images are generated (figure 2.6(e) to 2.6(h)). This is done by classifying the pixels in the training images

$$th_{x,y} = \begin{cases} 1, \text{ if } i_{x,y} > threshold \text{ from 2.30} \\ 0, \text{ otherwise} \end{cases} \tag{2.31}$$

Where th is the thresholded image and i is the input image. After processing the training image using equation 2.31, the pixels that correspond to the objects emitting heat are given a value of 1 and the objects that do not emit heat replaced by 0.

The training images after thresholding, are scaled such that the centroid of the foreground object lies at the geometric center of the image and that they are 48×128 pixels



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2.6: Pedestrian Images for Training Probabilistic Template

in size. After this is done, every pixel is treated as an individual distribution and the probability $p_{x,y}$ of that pixel being part of a pedestrian foreground is calculated by counting how frequently it appears as 1 in the thresholded training set $\frac{\sum_1^n i}{n \times 2}$.

To identify pedestrians in the current image X_n the template is scaled to a size of 128×48 pixels and the combined probability map for the image is calculated.

$$combinedprobablity_{i,j} = \sum_{x=1...48, y=1...128} (th_{x,y} \times p_{x,y} + (1 - th_{x,y}) \times (1 - p_{x,y})) \tag{2.32}$$

where, th is a window of size 128×48 around a pixel $p_{i,j}$ in the image X_n and $p_{x,y}$ is the probability for the pixel being part of the foreground from the thresholded training images. Once the probability map is generated then, image is thresholded using equations 2.30 and 2.31.

This pedestrian detection algorithm has been found to be robust in detecting pedestrians in cluttered infrared images, however it sufferers from some significant weaknesses.

As highlighted by Broggi [10] it is very difficult for a template to match all possible targets. Secondly, the template will not be able to classify occluded and partially occluded targets (region 2c in figure 2.2(a)). Finally classification for pedestrians colder than the threshold will fail as the target will be classed as part of the background.

2.3.1.2 Principal component analysis

Principal component analysis (PCA) when used to identify pedestrians is usually used in conjunction with a shape based classifier . Dai, Zheng and Li [41] use a trained Support Vector Machine (SVM) classifier to detect potential pedestrians that are then localised by the PCA model. A more recent publication by Malagon-Borja and Fuentes [44] will be used to illustrate how PCA classification is achieved.

For a set of n training images of size x, y each image I_i is represented by a vector ϑ_i with the length xy the mean for this set can be represented as

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\vartheta_i \tag{2.33}$$

$$\text{the co-variance matrix } c = \sum_{i=1}^{m}(\vartheta_i - \mu)(\vartheta_i - \mu)^T \tag{2.34}$$

The component vectors of C will be the principal vectors for the image set, a solution to this can be calculated using the QZ algorithm [45]. Sorting the vectors in decreasing order of eigenvalues and selecting a subset P of k vectors from C.

$$k : k \in xy, 1 \leq k \leq xy \tag{2.35}$$

and the projection of an image into a space defined by k vectors can be calculated as follows:

$$p = P(u - \mu) \tag{2.36}$$

$$u' = P^T p + \mu = P^T P(u - \mu) + \mu \tag{2.37}$$

the difference between the current image and the and the PCA model:

$$d = |u - u'| = \sqrt{\sum (u - u')^2} \tag{2.38}$$

The image stream is classified by finding the difference between the current image u and the eigenvector matrix using equation 2.38 as follows

- From the true training images from n to obtain the projection matrix P_{gp} and the mean μ_{gp}
- Using the edge information from true training images in n to obtain a projection matrix P_{ep} and mean μ_{ep}
- From the false training images from n to obtain the projection matrix P_{gn} and the mean μ_{gn}

- Using the edge information from false training images in n to obtain a projection matrix P_{en} and mean μ_{en}

To classify a target image I as a pedestrian, the image is *'reconstructed'* into I' by projecting it into the space defined by the k vectors. The image I and the its edges I_e are projected into the eigenspace defined by k vectors, using the equations 2.36 and 2.37 to obtain, $u'_{(I,gp)}$, $u'_{(I,ep)}$, $u'_{(I,gn)}$ and $u'_{(I,en)}$. The difference between the I and the projections using equation 2.38, then becomes $d_{t,I} = d_{(I,gp)} + d_{(I,ep)} - d_{(I,gn)} - d_{(I,en)}$. The total difference can then be used to classify the image u as

$$g = \begin{cases} \text{Pedestrian} & d_{t,I} \geq 0 \\ \text{Non-pedestrian} & d_{t,I} < 0 \end{cases} \tag{2.39}$$

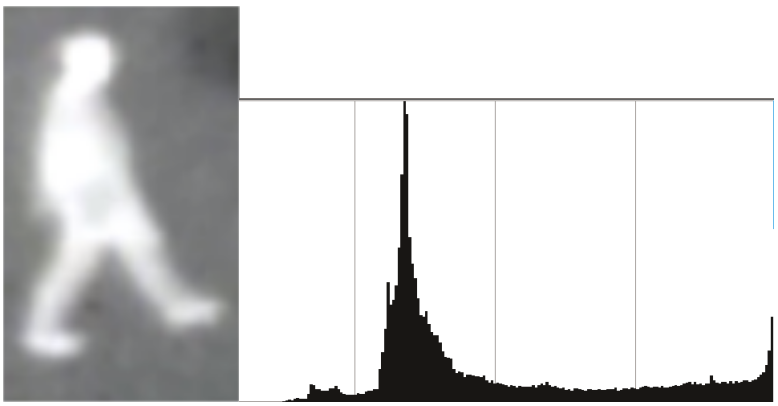
From the above we can see that the algorithm will work with a pedestrian from any perspective included as part of the training image set. It is also very sensitive to the performance of the edge detection algorithm.

2.3.2 Pedestrian Aspect Ratio and Pedestrian Symmetry

As previously shown (section 2.3) a significant body of work exists where the pedestrian detection in IR is achieved using non template based techniques. These techniques rely on looking for all targets matching pedestrian aspect ratio, or, searching for a physical feature (head, arm, legs etc.).

Figure 2.7: Pedestrian Features

One feature initially identified was that in IR images, pedestrians tend to be brighter than the background [8, 9, 46, 13, 47]. This feature was then found to be inadequate as pedestrians are not homogeneous targets [9], as the environmental temperatures change a *'polarity switch'* [48] may occur (pedestrian target colder than background). Due to the



(a) Pedestrian image (b) Corresponding histogram

Figure 2.8: Pedestrian with Corresponding Histogram

problems with using pixel intensity as illustrated, one other feature that becomes obvious when figure 2.7 is examined is that the heads of the pedestrians are distinct, this feature has been used to identify and track pedestrians [49, 50, 47, 42].

Another feature that has been found to be effective with pedestrian detection is the aspect ratio [13, 46] (the height of a pedestrian is twice the width of the pedestrian), aspect ratio is also used in section 2.3.1.1 for the template window (128×48 pixels).

2.3.2.1 Histogram for Pedestrian Detection

Image intensity histograms have been used to detect pedestrians by Bertozzi [13], followed by histograms of image gradients Zhang [51] to classify image regions as pedestrians. For example, if the histograms for pedestrians are examined (figure 2.8), the histogram for a pedestrian is in the form of a symmetrical spike which mirrors the shape of a pedestrian.

To do this, the image space is scanned to locate regions that have intensity peaks, bounding boxes are fitted to those regions, then histogram for only the bounding box is generated. For an image $i_{x,y}$, the bounding box $B_{l,m}$ is size constrained as follows:

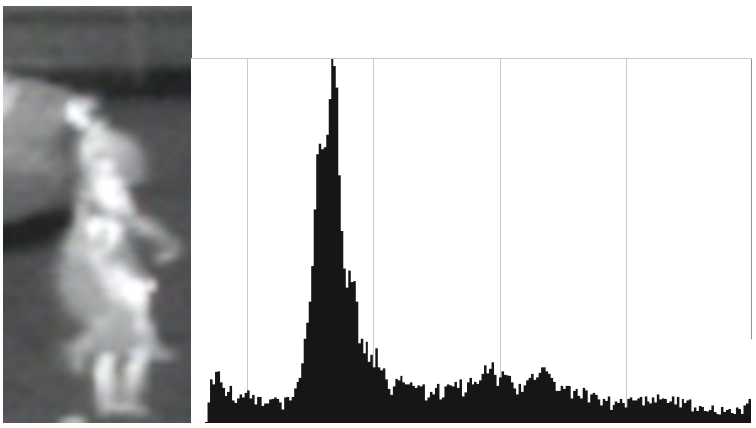
$$l : l \in I, 12 \leq l \leq 42 \text{ and } m : m \in I, 28 \leq m \leq 100 \tag{2.40}$$

$$\text{the mean for the image, } mn_n = \frac{\sum i_{x,y}}{x \times y} \tag{2.41}$$

The mean is then used to threshold the image and generate a thresholded image $j_{x,y}$ retaining the image information,

$$j_{x,y} = \begin{cases} i_{x,y}, & i_{x,y} > mn_n \\ 0, & i_{x,y} \leq mn_n \end{cases} \tag{2.42}$$

Bounding boxes are then placed on the regions left in the image in a coarse to fine format



(a) (b)

Figure 2.9: Illustration of Histogram Failure

i.e. the largest bounding box first followed by finer and finer bounding boxes till the smallest size of bounding box is reached.

When the bounding boxes are applied to the regions of interest the histogram for the bounding box is examined and based on the morphological characteristics of the histogram the boxes are classified as true targets of false targets. The histogram characteristics that are used to eliminate bounding boxes are:

- The centre of the histogram is empty
- More than half the histogram is empty
- When 80% of the histogram is confined to less than 25% of the area.

Any resulting histograms found not to be symmetrical or where the center is empty are discarded as false positives. This system is simple and robust at identifying pedestrians in infrared images.

2.3.3 Pedestrian Classifier - Summary and Metrics

This section introduced the different types of pedestrian classifiers. Section 2.1 introduced literature which precluded the use of 2D and 3D models as classifiers with infrared image sequences. Following this, the review in this section was constrained to template and pedestrian appearance based classifiers (figure 2.1).

From the subset of pedestrian classifiers examined, we can identify the following commonalities:

- The pedestrian aspect ratio is a physical attribute that is easily identified in infrared images. It can also be used to constrain sliding windows while classifying image content (sections 2.3.1.1 and 2.3.2.1).

- Intensity thresholding is a reasonably successful method of identifying regions of interest that may contain pedestrians in infrared images (2.3.2.1).
- The relative low surface texture content in infrared images makes it easier to generate templates for use with infrared images [8, 41] and section 2.3.1.1.
- Templates have higher failure rates [10] when used to validate pedestrians in infrared video sequences.

2.3.3.1 Pedestrian Classifier Metrics

The performance of a classifier module is expressed in terms of True Positive TP_c, True Negative TN_c, False Positive FP_c and False Negative FN_c rates. For the scene in figure 1.1(a),
the number of pedestrians in the scene $N_p = 7$ and
the number of foreground objects is $N_f = 8$. After classification
the number of *'objects of interest'* correctly identified, $N_c = 2$,
the number of other objects correctly identified $N_{oc} = 1$ and
the number of objects mis-classified as *'objects of interest'* $N_{mc} = 0$.
With this information we can now calculate the metrics for the classifier:

TP_c: This is the ratio of pedestrian objects identified accurately, to the number of pedestrians in the foreground, $TP_c = \frac{N_c}{N_p} = \frac{2}{7} = .28$.

TN_c: This is the ratio of non pedestrian objects classified accurately, to the number non pedestrians in the foreground, $TN_c = \frac{N_{oc}}{N_f - N_p} = \frac{1}{8-7} = 1$.

FP_c: This is the ratio of number of non pedestrian objects classified as pedestrian objects, to the non-pedestrian objects in the foreground,
$FP_c = \frac{N_{mc}}{N_f - N_p} = \frac{0}{8-7} = 0$

FN_c: This is the ratio of number of pedestrian objects misclassified to the total number of pedestrians in the foreground, $FP_c = \frac{N_p - N_c}{N_p} = \frac{5}{7} = .71$

A good pedestrian classification algorithm will have a high TP_c and TN_c rates with a correspondingly low FN_c and FP_c rates. Now that we have a basic understanding of how an object classification algorithm works and the metrics that can be used to evaluate the performance of the same. We can now review pedestrian tracking algorithms.

2.4 Pedestrian Tracking

The pedestrian tracking system taxonomy in figure 2.1, classified pedestrian trackers into two main types, which were briefly described in section 1.2.3. Tracking algorithms that

use segmented images tend to be faster than algorithms than those that use the raw images, as image segmentation and the subsequent classification of pixels into foreground and background regions reduces the search space which the tracking algorithm uses to match known targets.

The taxonomy in figure 2.1, identifies three main types of *'identify and match'* pedestrian trackers, they are:

Geometric Model Trackers : Model based trackers like model based pedestrian classifiers, track pedestrians by mapping pedestrians in the image sequence to the geometric model. Like the model based classifiers, there is insufficient texture information in infrared images to be used with model based trackers. There are various types of model based trackers, some examples:
Stick figures: A stick figure representing the human body [52].
2D ribbon model: The braided pathway formed by the limbs [53].
3D volumetric model: A model consisting of rigid surfaces and 25 joints [54].
Elliptical model: Length of the axis and the major and minor axes of the ellipse cross section [55].
Cylinder model: Uses cylinders of fixed ratio lengths to represent the human body [53].

Feature Based Trackers : Feature based trackers map the foreground objects being tracked to *'features'* such as the object centroid, cluster mean etc. and track the features. The features are used by the trackers to differentiate between different foreground objects. Some examples of feature tracking algorithms:
Features using Cluster Tracking [56].
Features using Kernels [57].
Features using Bayesian Networks [58].

Bounding Box Trackers : Bounding box trackers fit foreground regions with bounding boxes and track the bounding boxes. These trackers are similar to feature trackers and in some cases the bounding box could be considered as a feature. An example of this is a bounding box corner tracker by Meuter et al. [59], where pedestrians are tracked using the corners of the bounding boxes containing the pedestrians.

As with background models in section 2.2 and pedestrian classifiers in the previous section a small subset of pedestrian trackers will be examined in detail. Before reviewing the trackers, the challenges that tracking systems need to overcome are identified and reviewed.

2.4.1 Challenges to Tracking in Image Sequences

Successfully tracking pedestrians in image sequences requires the tracking systems to overcome two main challenges. The first is object occlusion and the second is false targets

due to shadows. The two challenges are examined in detail below.

2.4.1.1 Occlusion Events and Recovery

The introduction to pedestrian trackers in section 1.2.3 introduced the concept of *'ground truth'* and one of the ground truth categories is the number of occlusion events. An occlusion event occurs when a foreground object in the area under observation, is no longer identifiable as part of the foreground. This could be due to a background object between the foreground object and the cameras or another foreground object between the occluded object and the camera.

This temporarily reduces the amount of information available about the occluded object to the tracking system (both the classifier and the tracker), this increases the probability of this object being misclassified for the duration of the occlusion event. When an object tracking system identifies a possible occlusion event between two foreground objects, there are few general strategies that could be used for recovery after the event. These strategies are outlined below:

Merge targets: A tracker using this method of occlusion recovery, marks both the occluding and occluded objects as *'merged'*. This merged target can then be tracked as either as an intermediate object or as a group of objects in close proximity. This strategy is usually used when the tracker is unable to differentiate between the constituent foreground objects.

Drop one target: This type of occlusion handling requires the algorithm to be very accurate in terms of object recognition. When another object occludes an object being tracked it recognises the object in the foreground and tracks that, while the occluded target is marked as lost. If the occluded object moves out of occlusion, the tracker recovers and continues to track the previously occluded object, else it records the trajectory of the object till it was occluded. As the tracker continues to maintain track of one of the objects in an occlusion event, the overall track fragmentation will be lower than for a tracker using the *'merge targets'* strategy.

Drop both targets: This is easiest to implement, when two objects occlude it marks both the occluding and occluded objects as lost and tracks the new foreground object as a new target. Using this strategy with image sequences with occlusion events, results in a tracker where there are more tracks than there are objects.

Track both targets: If sufficient information is available about both the targets during the occlusion event, then the tracker could potentially track both targets simultaneously. This occlusion handling mechanism requires the tracker to be able to identify the occluded object with minimal or no information which may be the case for fully occluded objects, therefore this occlusion resolution mechanism is usually used in

trackers that have high levels of accuracy and able to discriminate between targets with very little information.

2.4.1.2 Shadows

Shadows, in visible image sequences or in the case of IR image sequences, reflections, are difficult for some image segmentation algorithms to recognise and classify as part of the background. Due to this, they tend to be classified as part of the foreground and passed on to the classifier/tracking algorithm as a region of interest. This distorts the foreground region being tracked, if the tracking algorithm is a template based approach it might fail to recognise the shadow as a false target, degrading accuracy.

This is one reason for the increased popularity of model based trackers with visual images as the model is able to maintain object track through shadows and occlusion events. However, as stated previously and identified by Fang et al. [7], the intrinsic properties of infrared images precludes the use of these trackers with infrared images. So alternative approaches will need to be considered.

2.4.2 Cluster Analysis

Now that we have an understanding of the challenges that trackers need to overcome before being able to successfully track pedestrians in image sequences, we can examine examples of pedestrian trackers that have been published in literature. The first tracker to be reviewed in detail uses an alternative approach to the one used by the proposed tracker.

The first method uses the expectation-maximisation (EM) algorithm to track foreground objects as clusters in the image sequence by modelling the image as a Gaussian distribution. This algorithm was first described by Pece in 2002 [56], it uses a static image s_t as a background model and the change in the pixel value from this static image to the current image cu to identify and track clusters. A detailed guide to the EM algorithm can be found in the book by Lachlan and Krishnan [60]. The functioning of the tracker is described below:

1. A static background image is used and is modelled as a single cluster (single gaussian distribution).

2. As neighbouring pixels do not affect each other, any gaps in the distribution are assumed to represent new objects in the scene.

3. The gaps are used to generate new clusters, which are tracked as long as there is a gap between the distributions.

4. For new frames, any existing clusters that do not match the clusters in the new frame are merged into the background cluster.

5. To ensure that viable clusters are not merged to the background cluster, the Mahalanobis distance between the cluster to be merged and the background is used to compute the merge cost. As long as this cost is below a set threshold the clusters are merged.

As the gaussian distribution for an image could be generated from any combination of the pixel properties such as colour and/or intensity, the implementation description will refer to the distribution features i.e. distribution mean, distribution variance etc. and the term *'cluster'* in this context refers to a cluster of pixels.

Initially the entire image is treated as a single cluster with index 0. This cluster represents the background, any clusters labelled as part of the foreground are later split and merged from this initial cluster. Assuming that the image has n_t target clusters excluding the background cluster, with indices $j > 0$ because when, $j = 0$ it is the background cluster. The parameters describing cluster j are described by φ_j. The number of pixels in an image is fixed at *pm* (*image resolution*) the number of pixels that cluster j accounts ω_j is one of the parameters that describes the cluster.

From single pixel background modelling algorithms we know that neighbouring pixels do not have any effect on the pixel i in the current frame *cu*, from this the probability that a pixel is a part of cluster j

$$f_j(i) = g(i|\varphi_j).h|\delta(i)|\varphi_j| \text{ where,} \tag{2.43}$$

φ_j are the parameters for the cluster,

$h|\delta(i)|\varphi_j|$ is a distribution of observable differences for the cluster (pixel grey levels).

$g(i|\varphi_j)$ is a gaussian function of the distance from the centroid of the cluster

$$g(i|\varphi_j) = \frac{1}{2\pi\sqrt{|\Sigma_j|}} e^{\frac{1}{2}(i-c_j)^T \Sigma_j^{-1}(i-c_j)} \text{ where,} \tag{2.44}$$

c_j is the centroid of cluster j Using the Bayes' theorem [61] the posterior probability of pixel i being generated by j can be estimated as:

$$po_j(i) = \frac{\omega_j f_j(i)}{f(i)} \text{ where,} \tag{2.45}$$

and ω_j is the probability that pixel i was generated by cluster j.

thus, the fraction of pixels in the image generated by cluster j can be represented by

$$\omega_j^{nt+1} = f(i)^{nt} = \sum_{j=0}^{nt} \omega_j f_j(i) \tag{2.46}$$

The parameters for cluster j can be listed as

$$\varphi_j = \{\omega_j, \mu_j, c_j, \Sigma_j\} \text{ where,} \tag{2.47}$$

μ_j is the mean for cluster j (if $j = 0$ its the mean for the background, else, it is the mean for object j).

Σ_j is the co-variance matrix for the distribution

As the EM algorithm is iterative, an example for estimating the mean at the $k+1$ iteration is shown below

$$\mu_j^{k+1} = \frac{\Sigma_i(\delta(i)p_j^k(i))}{\Sigma_i p_j^k(i)} \tag{2.48}$$

This estimation is done for all parameters (2.47) of all foreground distributions. However this is not undertaken for the background cluster. When a new frame is processed by this algorithm it first attempts to fit existing clusters to the new image. After this is done, any clusters that are not found to fit the new image are merged into the background.

Then clusters that are found to overlap need to be merged, to do this the Mahalanobis distance between the centroids of the two clusters is used to calculate the cost of merging clusters, if the clusters to be merged are j and k then

$$MC(j,k) = pm\omega_j \left[\log \frac{\varpi_k}{\varpi_j} - \frac{1}{2} D_M^2(j,k) - \frac{1}{2} tr(\Sigma_j \Sigma_k^{-1}) + 1 \right] \text{ where,} \tag{2.49}$$

$D_M^2(j,k) = (c_j - c_k)^T \Sigma_k^{-1} (c_j - c_k)$ is the squared Mahalanobis distance between the centroids

$\varpi_j \underline{\underline{def}} \frac{pm\omega_j}{2\pi\sqrt{|\Sigma_j|}}$ the density for pixels originating from cluster j

To split a cluster, the image plane is scanned and if there are gaps in the distribution within the cluster then the cluster is split. The number of new clusters created depend on the number of gaps in the distribution. To identify and create new clusters the background cluster is scanned for changes any clusters of pixels that are detected are added in as new clusters with parameters estimated using the EM algorithm.

This algorithm is robust in tracking targets and ensuring that events such as occlusion and shadows do not affect the tracking accuracy. For every frame in the sequence, the tracking algorithm has to compute the distribution properties for the clusters in the frame. After which the merge costs for any non-matching clusters need to be computed. These operations are compute intensive and as the number clusters increase the speed at which frames are processed degrades.

This pedestrian tracker is representative of trackers that use the predict and verify strategy to track pedestrians in image sequences. Additionally this tracker is accurate and able to recover from target occlusion and merge events while maintaining track of targets. Similar trackers described in literature have used kalman filters [50], Bayesian networks [62], hidden markov models [63] etc. to track pedestrians in image sequences. The next tracking algorithm to be examined uses, the *'identify and match'* strategy to track pedestrians in image sequences. It does so by employing pedestrian attributes to differentiate between pedestrian targets.

2.4.3 Pedestrian Attribute Tracking

Attribute tracking uses unique pedestrian characteristics such as aspect ratio, symmetry, colour, gait etc. to identify and track a pedestrians. However if the goal is to track a pedestrian in an image that contains other pedestrian targets, the challenge is much more complex. As the pedestrian target moves across an image plane, the size of the target changes, the colour/intensity levels will change in response to changes in the environment etc.

If, for example, the colour of clothing is used; there is no guarantee that there will be no other pedestrian targets wearing that same colour on the image plane at the same time. As we have already seen in section 2.3.2 the aspect ratio can be used as a pedestrian classifier, we will now examine a technique that uses the aspect ratio to track pedestrians.

The implementation is one by Pai et al [64] that is designed for pedestrian safety in at cross roads. This implementation uses two pedestrian attributes the aspect ratio [13] and the gait [65, 66, 67] to classify and track targets. To extract this information, the incoming image is thresholded to identify the foreground regions followed by shadow suppression. The pedestrian classification and tracking mechanism used by this algorithm can be described as follows:

- Identify foreground regions and suppress shadows in the image using a background modelling algorithm.
- From the foreground regions, remaining in the image, identify moving pedestrians by checking for foreground objects whose aspect ratio is in flux.
- Within the subset of foreground regions whose areas are in flux, foreground regions found to be deformable are initially classed as pedestrians and a bounding ellipse is fitted.
- For pedestrians fitted with bounding ellipses, gait information is collected by measuring the change in the area between the feet.
- The gait information is unique to a pedestrian and is used to re-establish tracks after occlusion events.
- Additionally, the fourier power frequency of the pedestrians gait is used as a validating classifier for pedestrians.

Pedestrian recognition relies on the change in aspect ratio that occurs as the pedestrian walks. The torso is located by using an axis ratio constrained ellipse that is fit onto the foreground contours.

This is done using by finding the area occupied by the pedestrian feet A_{obj} and the area occupied by the same pedestrian when standing still A_{sil}, the difference between the two A_{ratio}. As a pedestrian walks A_{sil} will change but A_{obj} will stay the same (relatively).

This implies that A_{ratio} will rapidly change if the target is a pedestrian but remain static if the target is not a pedestrian. Shannons' entropy is used as a measure for rate of change:

$$E_{ratio} = -\sum_{1 \leq i \leq N} p(i) \log p(i) \text{ where,} \tag{2.50}$$

N is the number of blocks covered by the foreground region

$p(i)$ is the ratio between the number of contour points and the total number of points covered by the object.

With E_{ratio} is a measure of whether a foreground object is deformable or not. To fit a bounding ellipse, it is initially fitted to the center of the foreground region. To adjust the length, for every point in the upper silhouette the vertical distance between it and the and the upper half of the ellipse is calculated d_u^v (v represents vertical and u upper limit) if this is positive, then ellipse is too small; similarly for the lower difference is also calculated d_l^v, this information is then used to adjust the fit of the ellipse

$$A' = A + (\min\{d_u^v\} + \min\{d_l^v\}) \tag{2.51}$$

$$C'_y = C_y + \frac{\min\{d_u^v\} + \min\{d_l^v\}}{2} \text{ where,} \tag{2.52}$$

A is the long axis of the initial ellipse

C_y is the vertical co-ordinate of the initial ellipse

While the hough transform [68] could also be used for fitting the ellipse, this technique is faster.

As the pedestrian moves gait information that has been shown to a feature that is unique [65, 66, 67] is observed. This information for a target is reliably recorded and is used to re-establishing tracks after occlusion events.

As we already know the area between the feet of pedestrians A_{obj}, the maximum and minimum distance between a targets feet can be identified. This area is used to measure the gait frequency for that target. After collecting the gait frequency for some time (2-3s) using a Fourier transform the power p_a of every frequency can be obtained. The power ratio with an empirically set pedestrian confidence threshold acts as a secondary pedestrian classifier in addition to the bounding box. This is done as follows:

- Set power $p = 0$ if lower than a predefined threshold (insufficient track data)
- Sum the powers p_p where the gait frequency $f : 1.5 \geq f \geq 2.5Hz$
- calculate the power ration by dividing the $p_R = \frac{p_p}{p_a}$
- if the above ratio p_R greater than the threshold, the object under consideration is a pedestrian

The power frequency of this distance measure will be the gait periodicity for that individual target. This gait information helps to help recover track information after occlusion and merge events. This tracker uses two pedestrian attributes, the aspect ratio and the gait to implement a pedestrian tracker and to validate pedestrians and reduce the tracker false positives.

As the algorithm was designed with pedestrian safety, only a very low false negative or false positive rates are acceptable. While the deformable bounding ellipse could potentially be used, the following pedestrian tracker is more suited for use with infrared images for pedestrian tracking.

2.4.4 Matrix Bounding Box Tracker

Like the tracking algorithm examined in the previous section, the tracker that will be detailed in this section uses bounding boxes to track pedestrians. It does so by matching pedestrian (*'blobs'*) in the current image with the pedestrian (*'blobs'*) in the previous images in the sequence. This algorithm uses two way matching matrices to track blobs between image frames as described below:

- Identify all the blobs (foreground regions of interest ROI's) in an image.
- Any overlapping blobs are treated as matched blobs. For any blobs with multiple matches, the *'match string'* is generated.
- This match string is also used to resolve other tracking problems, such as occlusion handling, blob split, merge and the introduction of new blobs to the image space.

The implementation of the tracker is described next:

For two images $X_{x,y}$ and $X+1_{x,y}$ with M and N blobs respectively. Any bounding boxes that overlap between frames are considered as matches [69], multiple bounding boxes may overlap as the pedestrian density increases. To overcome problems with multiple matches *'Matching Strings'* are used to identify matches. This is done by generating two match matrices for the two sets of blobs $B_{(t)} : B_{(t)} \in M$ and $B_{(t-1)} : B_{(t-1)} \in N$

$$m_{t-1}^{t}(i,j) = \text{Matching}\{B_{i,(t-1)}, B_{j,t}\} \tag{2.53}$$

$$m_{t}^{t-1}(i,j) = \text{Matching}\{B_{i,(t)}, B_{j,(t-1)}\} \tag{2.54}$$

$$\text{the matching string, } S_{t-1}^{t} = \bigcup_{j} \frac{j}{m_{t-1}^{t}(i,j)} = 1 \tag{2.55}$$

Where, *i,j* are blobs whose bounding boxes overlap in the two images under evaluation.

The matching string in equation 2.55 will contain a reference for all the blobs that that match between the two frames. Blobs in frame $X_{x,y}$ with only one matching blob in frame $X+1_{x,y}$ are tracked by the change in the x,y co-ordinates of the centroid of bounding boxes.

However, when more than one blob in $X + 1_{x,y}$ matches a blob from $X_{x,y}$ there has been an occlusion event. Occlusion events and new targets are identified as follows:

New Blobs: Any new blobs will have no matches in frame $X_{x,y}$ i.e. $B_{N,t} = S_t^{t-1} = \emptyset$

Blob Merge, Occlusion: When two blobs merge the tracker is able to identify the blobs that merged and also track the merged object $B_{N,t} \equiv B_{i,(t-1)} \bigcup B_{j,(t-1)}$.

Blob Split: When a tracked blob splits into to or more blobs $B_{i,(t-1)} \equiv B_{J,t} \bigcup B_{N,t}$.

Any blobs in frame $X + 1_{x,y}$ that matched multiple blobs in $X_{x,y}$ and not marked as merged or split blobs using the above mechanisms, are marked as lost in the current frame ($X + 1_{x,y}$). While this increases the track fragmentation rate, it avoids the problem of the track being assigned to the wrong object.

2.4.5 Pedestrian Trackers - Summary and Metrics

In this section we examined, in detail, a small subset of tracking algorithms from literature. Examples of the different genus of trackers identified in figure 2.1, except for model based trackers were examined. The justifications for excluding model based trackers are well founded in literature [7, 10] and have been outlined in section 2.1.

From the three algorithms examined, we can observe the following:

Occlusion : The occlusion handling mechanisms for the three trackers are not designed as add-on's to the tracking systems. Instead the different occlusion resolution mechanisms are an extension of the trackers themselves (Mahalanobis distance in section 2.4.2; Match Strings in 2.4.4 and finally Pedestrian gait for occlusion recovery in 2.4.3).

Shadows : Only one of the three trackers uses an explicitly defined shadow suppression system (section 2.4.3). Of the other two, the accuracy of the pixel cluster tracker will be degraded by shadows and the bounding box tracker is reliant on the ability of the background model to accurately suppress shadows and their effect on the accuracy.

Taxonomy : Classifying pedestrian trackers into rigid categories is very difficult; for example, this document classifies the aspect ratio constrained tracker from section 2.4.3 as a bounding box tracker. However, as the tracker uses a deformable pedestrian model to fit the bounding box, it could potentially be classed as a model based tracker. Finally, if, the elliptical bounding box used is considered a feature that is tracked, then the tracker could be classed as a feature tracker.

Use of Pedestrian Attributes : Using attributes unique to the pedestrian class such as gait, appearance etc. to discriminate between pedestrians improves accuracy. As accuracy improves, there is a reduction in occlusion recovery time (section 2.4.3).

Now that we have a better understanding of the different types of pedestrian trackers that have been used in literature, the metrics used to evaluate their performance needs to be reviewed.

2.4.5.1 Performance Metrics for Pedestrian Trackers

To evaluate the performance of a pedestrian tracking system two items are essential; a benchmark image sequence and the *'ground truth'* for that sequence. The standard metrics that are used to evaluate the performance of pedestrian tracking systems are well understood have been described by Desurmont [70], Fernandez-Garcia [71], Black [72] and Renno [40] and can be summarised as:

1. Tracking Success Rate (T_{sr}): The ration of the number of tracked objects to the total number of ground truth objects.

2. Track Detection Rate (T_d): Ratio of the number of detected ground truth points for an object to the total number of ground truth points for the object.

3. Tracker Detection Rate (T_{dr}): This is a ratio of the total number of positive tracks detected to the number of ground truth points that were generated.

4. Track Fragmentation (T_f): Ratio of the number detected tracks matched to one ground truth track.

5. False Alarm Rate (T_{fa}): The ratio of the total number of false positive (FP) tracks to all the tracks detected by the algorithm including false positives.

6. Occlusion Success rate (T_o): Ratio of number of occlusions successfully detected to the total number of occlusion events.

Of the above six metrics, five are different ways of measuring the ability of the tracker to recover after occlusion events.

To illustrate the use of ground truth, tracker metrics and the the performance of an object tracker; the sequence of images in figure 1.5(a) to figure 1.5(c) will be used with a simulated tracker. The objects to be tracked are numbered (figure 1.5(a), 1 to 4), the simulated track data for the objects is shown in figure 1.5(g). Table 2.1 summarises the ground truth and simulated track information for the image sequence.

The data in the table is as follows:

Image: Image from sequence in figure 1.5.

GT_i: Number of ground truth objects in current image.

GT_o: Occlusion events in current image.

T_{No}: Number of objects identified in current image by simulated tracker.

Obj_i: Object number in current image.

$GT_{p,i}$: Number of ground truth points for current objects.

$GT_{t,i}$: Ground truth track points for current object in image.

$M_{GT,i}$: Do the simulated tracked location and the ground truth location match?
1: *yes*; **0**: *No*.

T_m: Do the simulated tracked location and the ground truth tracked location match?
1: *yes*; **0**: *No*.

$O_{s,i}$: Did the simulated tracker successfully track the object during an occlusion event?
- : *NA*; **1**: *yes*; **0**: *No*.

From this information we can calculate the performance metrics for the object tracker in figure 1.4,

Tracking Success Rate: $T_{sr} = \frac{T_{No}}{GT_i}$.

Track Detection Rate: $T_d = \frac{\sum M_{GT,i}}{\sum GT_{p,i}}$.

Tracker Detection Rate: $T_{dr} = \frac{\text{Count of all non zero elements in } T_m}{\sum GT_{t,i}}$

Track Fragmentation T_f, only object two has a fragmented track which is not recovered. Hence, cannot be demonstrated for this example.

False Alarm Rate: T_{fa}, there are no false targets in the example, so this will be zero.

Occlusion Success rate: $T_o : \frac{\sum O_{s,i}}{\sum GT_{o,i}}$

Using the above information and the simulated performance data in table 2.1, the performance metrics obtained are in table 2.2. From the information in this table we can infer the following about the performance of the simulated tracker:

- The tracker performs very well when tracking distinct objects, $T_{sr} = 1$ for figure 1.5(d).
- Occlusion handling in the tracker works when there is a significant colour difference between the objects in the occlusion event. T_o for 1.5(f) is better than the T_o for 1.5(e).
- The track fragmentation rate for object two will be high as the tracker is not able to recover after the occlusion event (figure 1.5g).
- Though the tracker is able to track, what it thinks is object 4 in figure 1.5(f). The intersection between the identified location and the ground truth is insufficient for it to be considered a valid match.

Table 2.1: Ground Truth and Track Data for Image Sequence in Figure 1.5

Image	GT_i	T_{No}	Obj_i	$GT_{p,i}$	$GT_{t,i}$	$GT_{o,i}$	$M_{GT,i}$	T_m	$O_{s,i}$
1.5(d)	**4**	4	1	**1**	**1**	**0**	1	1	-
			2	**1**	**1**	**0**	1	1	-
			3	**1**	**1**	**0**	1	1	-
			4	**1**	**1**	**0**	1	1	-
1.5(e)	**4**	3	1	**1**	**1**	**0**	1	1	-
			2	**1**	**1**	**1**	0	0	0
			3	**1**	**1**	**0**	1	1	-
			4	**1**	**1**	**1**	1	1	1
1.5(f)	**4**	3	1	**1**	**1**	**1**	1	1	1
			2	**1**	**1**	**0**	0	0	-
			3	**1**	**1**	**1**	1	1	1
			4	**1**	**1**	**0**	0	0	-

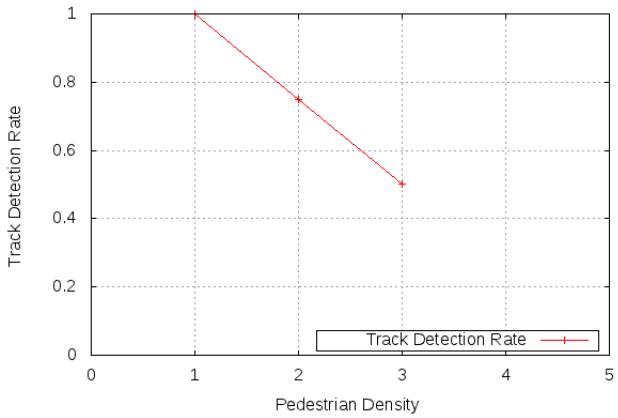
Table 2.2: Performance Metrics for Simulated Algorithm

Image	T_{sr}	T_d	T_{dr}	T_o
1.5(d)	1	1	1	-
1.5(e)	.75	.75	.75	.50
1.5(f)	.75	.75	.50	1

The performance metrics for a simulated tracker, in table 2.2 are difficult to understand. To make this easier to interpret, the performance of pedestrian tracking systems usually is graphed as receiver operator characteristic (ROC) curve. For example, the track data rate (T_{dr}) ROC curve for the data in table 2.2 is shown in figure 2.10(a).

When this is examined, it becomes obvious that the simulated trackers performance degrades drastically from image 1.5(d) to 1.5(f). The precision of a pedestrian classifier and tracker is its ability to localise a pedestrian within a bounding box. When the *'ground truth'* and the locations identified by a pedestrian tracking system for a pedestrian are compared to generate the performance metrics for the tracking system, exact matches between the two locations are not required. Instead a large enough overlap between the ground truth and the tracked location is accepted as valid (usually above 80%).

As the precision of a classifier and tracker improves so does its ability to accurately localise a pedestrian within the bounding box, improving percentage overlap between the ground truth location and the tracked location.



(a)

Figure 2.10: Receiver Operator Characteristic (ROC) Curves for Generic Tracker

2.5 Literature Review Summary

This chapter reviewed the literature are relevant to understanding and evaluating a pedestrian tracking system using infrared image sequences. The beginning of the chapter introduced a simplified taxonomy for use with pedestrian tracking systems derived from taxonomies described in literature.

The first section of this chapter (section 2.1) introduced the properties of infrared images identified in literature that make it a promising area of research for pedestrian tracking systems. This same section also excluded the use of some of the most accurate pedestrian classifying and tracking algorithms (geometric model based techniques).

Following on from this, single pixel background models were identified as being most responsive to changes in image sequences and two examples were detailed. When examining the pedestrian classifiers in use with infrared images we identified that the pedestrian aspect ratio is a promising physical attribute that is identifiable in infrared image sequences.

On further studying pedestrian classifiers, we identified that it is not possible to generate templates for all possible combinations of pedestrian appearances which has a detrimental effect on the accuracy of classifiers using templates. After this we examined the metrics that are used with pedestrian classifiers and how that relates to the ground truth of a sequence of images.

The final section reviewed pedestrian trackers described in literature. Again the taxonomy identified figure 2.1 was used to select the algorithms examined in detail such that they were representative of all the categories in identified in the taxonomy. From the review, we found that the occlusion recovery mechanism for a tracking system is usually inherent in the tracking process. We also found that using unique pedestrian attributes to discriminate between pedestrian targets improves the overall accuracy of the tracking system.

Chapter 3

Imaging System and Dataset

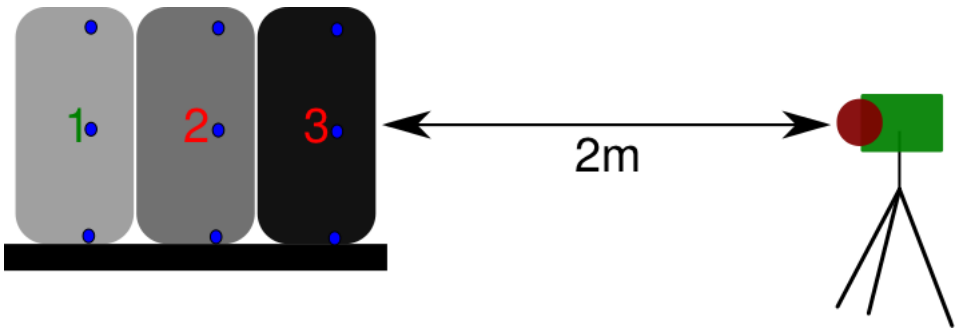
This chapter will help the reader understand the operation of the imaging system (infrared camera) and the pedestrian dataset that was used to benchmark the performance of the pedestrian detection and tracking algorithms in the following chapters. First the infrared cameras response to temperature changes were recorded and these observations will be presented. Following this the locations at which the infrared pedestrian image dataset was collected are described and the segmentation of the dataset is described.

3.0.1 Imaging System

The core of the imaging system that was used for data collection was the infrared camera. Section 1.1 presented a basic introduction to infrared images recorded using microbolometer arrays, the same section also highlighted the importance of texture information in distinguishing between objects at the same temperature.

The normal human body temperature is around 37^o C, but recording some test images revealed the following:

- The grey scale changes in the images were sensitive to the temperature range in the image i.e the the coldest and warmest objects in the scene



(a) Illustration of apparatus used to acquire images

Figure 3.1: Experiment Setup

- The temperature drift of the camera was very low, automatic recalibaration to correct for current drift was infrequent even when the camera was being used in a warm enclosed space.

3.0.1.1 Experiment Setup

To quantify the response of the camera to temperature range, the experimental in figure 3.1(a) was used (. The apparatus consisted of the following:

1, 2 and 3 : Conductive metal cans painted black to decrease reflectivity.

Infrared Camera : Two meters away from the cans and with the center of the lens in line with the center of the cans. The distance of two meters was emperically found to be the distance at which the cans occupied the entire scene without any of the background being visible.

Blue dots : Thermocouples to monitor can temperature.

To change the temperatures of the metal cans, they were filled with water at the following temperatures:

Can 1 : Ice + water, at 0^oC

Can 2 : Water at room temperature, between 12^o and 14^oC

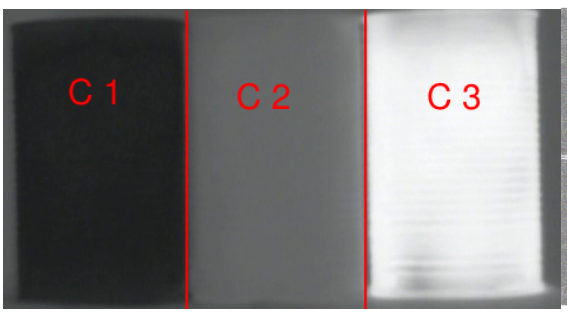
Can 3 : Water at 95^oC

Can 3 was then allowed to cool, while all the cans were stirred periodically to prevent thermal stratification. Thermal stratification in the cans would make them non-uniform. The thermocouples were used to ensure that the temperature gradients within the cans was less than 2^oC.

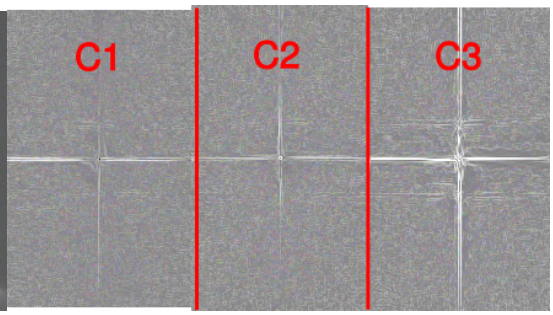
The image sequence in figure 3.2 shows the changes over time. Figure 3.2(a) is the image recorded at the start of the experiment (cans at 0^oC, 11^oC and 95^oC respectively; 0 minute mark in figure 3.2(g)) and the image in 3.2(e) is the image recorded at the end (50 minute mark in figure 3.2(g); cans at 0^oC, 12.5^oC and 50^oC respectively). The chart in figure 3.2(g) is the mean temperature recorded by the thermocouples for the cans over the duration of the experiment.

A method to quantify the surface texture changes for the cans between the three images (figure 3.2(a) , 3.2(c) and 3.2(e)) was necessary . Wavelets have been used to compare the texture information in infrared images to the texture information in visible images [4], the same publication also identified that infrared images have less texture information. As the change in texture was expected to be small and difficult to highlight with wavelet statistics, alternative texture analysis techniques were examined.

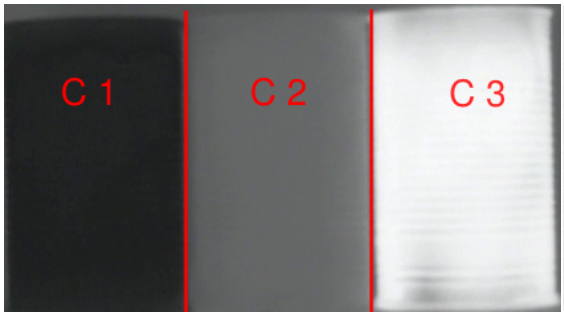
Fourier transform analysis of surface texture well researched and has been used to both match and generate surface texture [73, 74]. The eight neighbour fourier transform



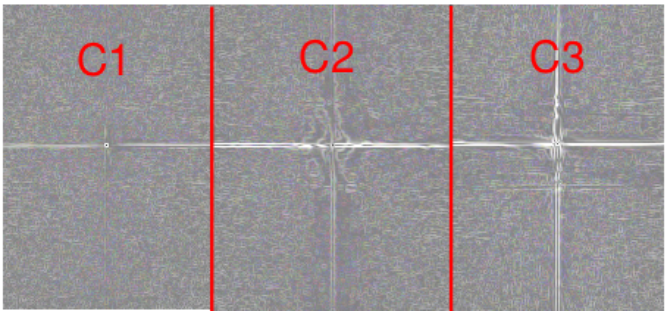
(a) C1 @0^{o}C, C2 @11^{o}C and C3 @95^{o}C



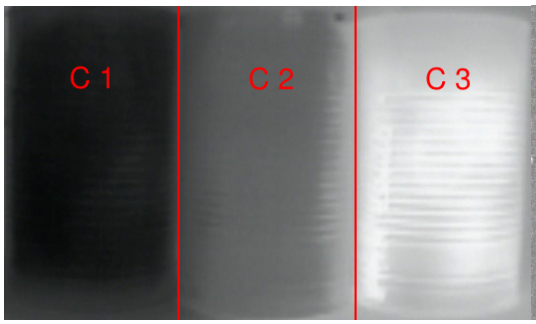
(b)



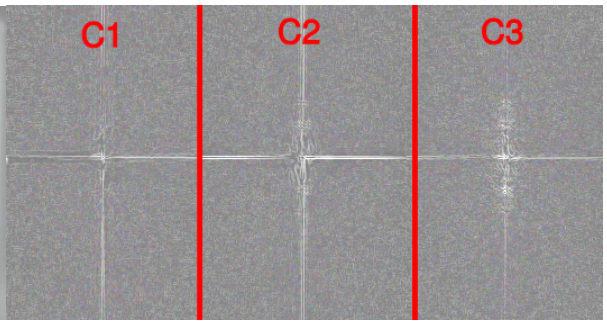
(c) C1 @0^{o}C, C2 @11^{o}C and C3 @70^{o}C



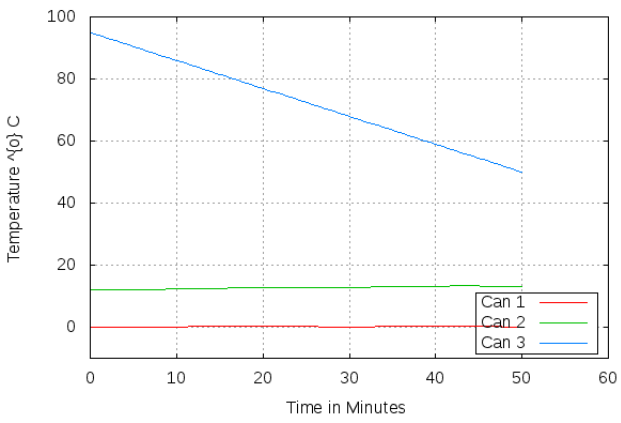
(d)



(e) C1 @0^{o}C, C2 @11^{o}C and C3 @50^{o}C

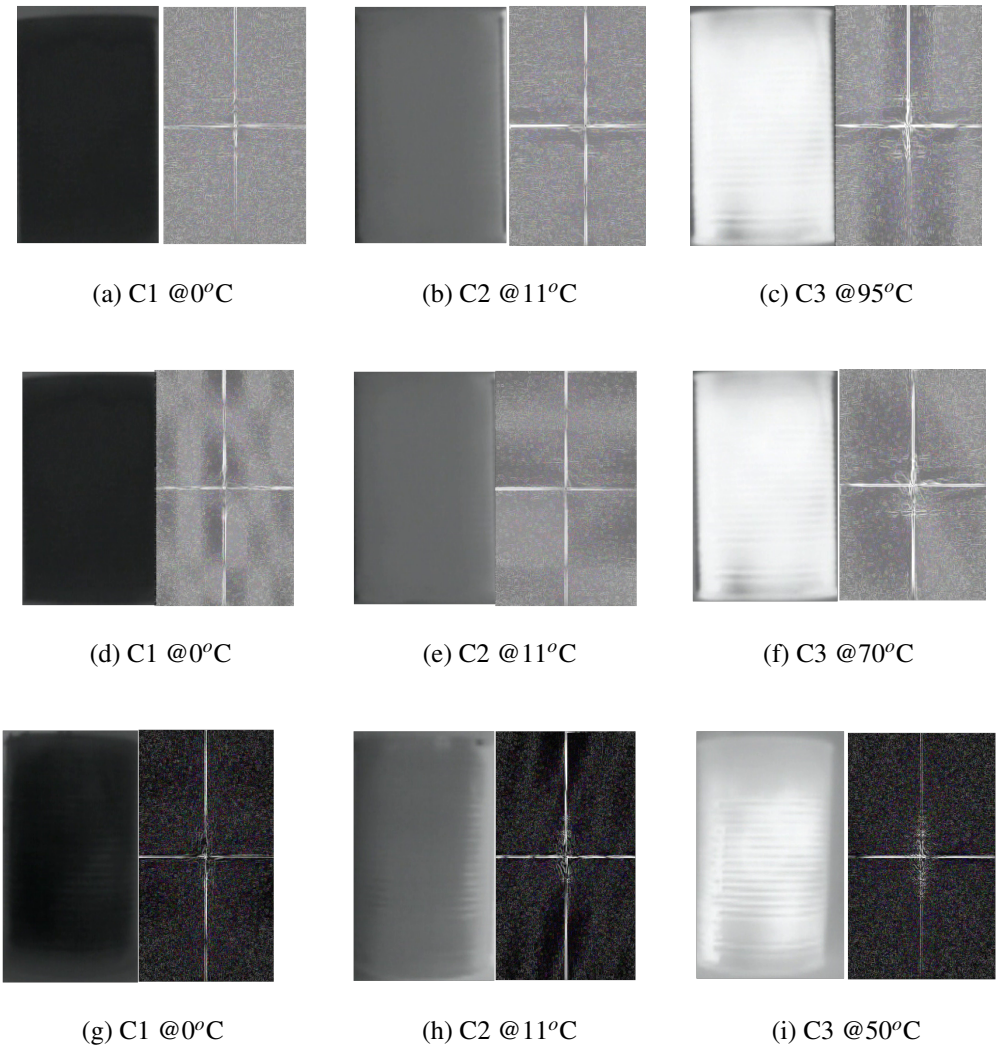


(f)



(g) Temperature change of contents

Figure 3.2: Images Recorded During Experiment



(a) C1 @0^{o}C (b) C2 @11^{o}C (c) C3 @95^{o}C

(d) C1 @0^{o}C (e) C2 @11^{o}C (f) C3 @70^{o}C

(g) C1 @0^{o}C (h) C2 @11^{o}C (i) C3 @50^{o}C

Figure 3.3: Fourier Transforms, Demonstrating Uniformity of Temperature

analysis of texture proposed by Zhou et al. was used compare the texture content of the cans in the three images. Figures 3.2(b) to 3.2(f) are generated applying the fourier transform to the corresponding infrared image.

The surface texture map for a can was generated by using an 10x10 pixel window and is mapped along the vertical axis. The intensity and height of the y-axis in the fourier map is proportional to the surface texture content. The map also shows the temperature change between the cans along the horizontal axis. To demonstrate temperature uniformity, fourier maps were generated by modifying the fourier transform to measure fine texture, figure 3.3. The inverted map is shown because single points along the texture map were difficult to identify when scaled to fit into the page.

From the first image of the sequence, figure 3.2(a) and its corresponding texture map in figure 3.2(b). The fourier transform maps for Cans 1 and 2, indicate that they do not have any recorded surface texture; Can 3, on the other hand has visible and prominent surface texture. This demonstrates the compression and loss of surface texture for Cans 1 and 2.

As the temperature range narrows, the camera begins to record more surface texture for Cans 1 and 2. The change in the vertical axis of the texture map for Can 2 (figure 3.2(b) and figure 3.2(d)) demonstrates this. However, to the casual observer, there doesn't seem to be any change in surface texture for Can 2 between figure 3.2(a) and figure 3.2(c). This indicates that in figure 3.2(c), surface texture for Can 2 is compressed, not lost; hence it is recoverable. In the final image of the sequence, figure 3.2(e), the textures are prominent and easily identified which is reflected in the texture maps for the cans in figure 3.2(f).

From the above images and their fourier transforms, we can infer the following:

- The temperature to grey scale mapping is non-linear and the warmest objects are mapped to a larger proportion of the grey-scale range. This is most obvious in figure 3.2(a). Where, Can 3, has prominent surface texture unlike Cans 1 and 2. If the temperature to grey scale map was uniform, then the texture map would be distributed uniformly for all three cans.

- Some surface texture, irrespective of the temperature range, is present in the final image. But the output medium (grey-scale image) hides this information.

This helps answer the first of the two research questions,
In grey-scale infrared images, due to auto-ranging, some texture information is lost and compressed. How much of this information is lost?
The texture information loss is highest for objects below or at the background temperature range in an image.

3.1 Texture Visibility in Infrared Images

In the previous section, we examined the loss of texture information in an infrared image due to auto ranging. During this we found that the mapping of incident radiation to grey-scale was not linear and that some of the texture information in an image is compressed and could be recovered. Texture recovery in this context means, improving the contrast in the image so that surface textures are more easily perceived.

One possible solution would be to normalise the intensity scale used. Research has shown that this is not effective and increases the noise in the image [4]. Additionally, normalising an image recorded with a non-linear scale will lead to texture loss in the higher intensity range in an image.

3.1.1 Texture Recovery Algorithm

Using the above information the following algorithm was developed to highlight compressed texture in an infrared image:

I. Find the median intensity $\hat{i}$ for the image

II. Find intensity gradients in the infrared image.

III. Mask regions in the image with steep gradients larger than min_{sz} and regions above median intensity ($\hat{i}$)

IV. Check unmasked regions for contiguous areas larger than minimum size min_{sz}.

V. Rescale the contiguous unmasked regions of the image.

The following sections will detail the implementation detail for the algorithm and also the reasons for the control parameters.

Intensity Gradient : Intensity gradients are used to identify textures that are not compressed and mask them from the initial rescaling procedure. Preserving the intensity gradient from changes during the rescaling procedure preserves texture information in the image.

Median Intensity : The median intensity $\hat{i}$ of the image is used to identify the regions of the image that are warmer than the rest of the image and prevent them from being rescaled. As identified in the previous section, the camera uses a non-linear grey scale to temperature map (figure 3.2(a)), masking these regions reduces the noise introduced into the image by rescaling the pixel intensity.

Minimum Area : A minimum area min_{sz} is used to reduce the noise that might be added to the infrared image by rescaling operations that operate on single pixels. This control parameter is camera model dependent, as different models will have different noise profiles.]

Rescaling : All the pixel intensity values between the 0 (black) and median intensity $\hat{i}$ of the image will not be in use. The rescaling operation uses these unused intensities to improve the contrast between the pixels in the image. After the unmasked regions are rescaled, the regions previously masked due to steep intensity gradients but with intensities below the median $\hat{i}$ are remapped to the new intensities while preserving the intensity gradient.

3.2 Results in Sample Images

A sample outdoor infrared image (figure 3.4(a)) will be used to outline the contrast enhancement process. The median intensity for the image is identified by using an intensity histogram (figure 3.4(b)). The intensity gradients in the image are identified and masked, thick lines in figure 3.4(c). The areas to be enhanced are marked (thin white lines in figure 3.4(c)). The contrast enhancement operation, for this image was most effective when using a minimum area of 4x4 pixels.

The processed image is shown in figure 3.4(d), where, texture features such as physical shapes have been enhanced and easier to identify. But the noise in the image has increased. Also, the fine texture details on the warm pedestrians though unaltered, are less prominent. The current contrast ratio to improve the image surface texture visibility is a subjectively set, so more detailed research into the contrast ratios that could be used to improve texture visibility is required.

3.2.1 Preliminary Results

The enhanced visibility of texture content in the final image (figure 3.4(d) indicates that this algorithm is capable of improving texture contrast in infrared images. However, the final image is also very noisy, indicating that the algorithm as described is inadequate and needs further research. Developing the following will help automate the process while reducing noise in the final image:

- Automatic grey scale map selection based on the median intensity in an image.
- An objective measure for surface texture visibility.
- Deformable minimum area windows to better fit image contours.
- Noise suppression mechanism/filter.

With the above implementation we can now answer the third research question *Can this information loss and compression be identified and texture visibility improved?* Yes, the the visibility of the texture details can be enhanced but the algorithm as described introduces too much noise into the resulting image.

3.3 Summary and Implications on Pedestrians in Infrared Images

The first section of this chapter established that surface texture is present in an infrared image. It also established that surface texture loss is the highest for objects below or at the background temperature for an image. The fourier transform map also demonstrated that due to the poor contrast ratios in infrared images, some surface texture in the image is difficult to identify.

However, enhancing the visibility of these surface textures using a linear pixel intensity remapping will lead to a loss of surface texture as the grey scale map used for generating an infrared image is non linear. Instead using this non-linearity as a feature, a contrast visibility enhancement algorithm was described (section 3.1).

The results of processing an image using the proposed algorithm (section 3.2) show that while this algorithm is able to enhance the visibility of surface textures in infrared images, the final images are very noisy. Hence further work is required before the proposed algorithm could be effective at enhancing infrared image contrast automatically.

3.3.1 Implications on Pedestrians Tracking

As it has now been established that the camera uses a non-linear grey scale to temperature map, texture details for pedestrians warmer than the background will be better. This will help identify individual pedestrians in an occlusion event. Conversely, for any pedestrians colder than the background, less texture details will be recorded this may hinder individual pedestrian identification during occlusion events.

3.3.1.1 Contributions and Further Work Summary

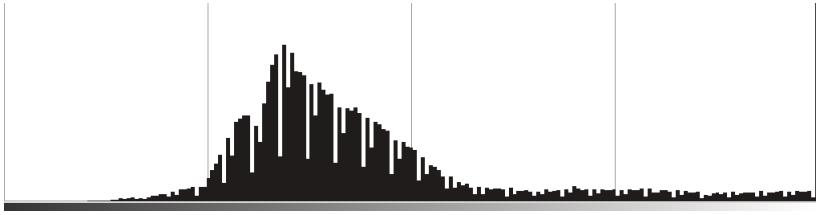
This chapter addressed two of the research questions identified in the first chapter (section 1.5.1),

> In grey-scale infrared images due to auto-ranging, some texture information is lost or compressed. How much of this information is lost?
> Can this information loss and compression be identified and texture visibility improved?

Here we found that for infrared images recorded using the FLIR [1] camera, the grey-scale to temperature mapping is non-linear. This means that finer texture details are recorded for pedestrians at the warmer end of the grey scale range in the image. A novel algorithm to enhance the visibility of the surface texture details was introduced and its effect on a sample image was demonstrated.

The while the texture visibility in the processed image was improved, the measure of change was subjective. Identifying the research needed to improve the usability of this algorithm.

(a) Original Image



(b) Image Histogram

(c) Intensity Gradient Mask

(d) Contrast Enhanced Image

Figure 3.4: Texture Recovery in a Sample Infrared Image

Chapter 4

Performance of Existing Algorithms

This chapter is to present to the results for research into the third research question:

> Is using the time taken to process a single target an effective metric to predict the performance of a tracking system for a specified target (pedestrian) density?

The term *'process'* refers to pedestrian classification, location matching and track data generation. Usually these pedestrian detection and tracking systems are implemented on computing platforms that are state-of-art with plenty of computing resources.

This increases the cost of the platform, making the systems less accessible and increasing associated hardware costs. If pedestrian density and processing speed correlate, platforms used for implementing pedestrian detection and tracking systems can be more accurately specified making the systems more accessible while improving resource utilisation and reducing cost.

If the change in speed (frame-rate) is linear and correlates with the number of targets in the sequence it will be evidence for processing time per target for the system staying the same. Classifiers and trackers implement false target detection and occlusion recovery mechanisms (partially occluded target detection in classifiers, occlusion recovery mechanisms in trackers). These require processor time to execute, hence as the target density increases in an image sequence, these events may occur more frequently resulting in these mechanisms being executed more often which should lead to a non-linear response.

This chapter is organised as follows, the first section describes the dataset and the locations at which the dataset was collected. Following this the performance data for three pedestrian classifiers and a tracker from literature is used to validate the use of 'time per target' as a metric for pedestrian tracking systems. The final section summarises the findings and introduces the next chapter.

4.1 Dataset for Evaluation

In chapter 3 we examined the properties of infrared images recorded using the micro-bolometer camera [1]. There we found that texture information for objects in infrared images does not consist solely of temperature data (section 3.0.1.1) and that the infrared camera being uses a non-linear temperature to grey scale map.

This has the following effects when classifying and tracking pedestrian targets in infrared image sequences:

- The image sequence will contain more details for pedestrian targets warmer than the background.
- Pedestrian classification and tracking algorithms described in literature for use with visual image sequences will have sufficient texture (chapter 3 and [4]) information to be effective when classifying or tracking pedestrians in infrared image sequences.

To characterise the performance of a pedestrian tracking system, a standard dataset is necessary. Two infrared pedestrian image sequence datasets previously used in literature by Davis and Keck [75] (OSU [1]) and by Conaire, O'Connor and Smeaton [76] (DCU) and are publicly available. These datasets are limited in the number of combinations of pedestrians and occlusions events in the sequences. These limitations justify the use of a new dataset.

The data set was generated from infrared video recorded indoors at a location with dense pedestrian traffic. The location, camera height, camera angles, pedestrian paths are illustrated in figures 4.1(a) and 4.1(b). Sample images for the two camera heights are in figures 4.1(c) and figure 4.1(d). The pedestrian tracks identified in figures 4.1(a) and 4.1(b) are the most common paths followed by pedestrians in the dataset.

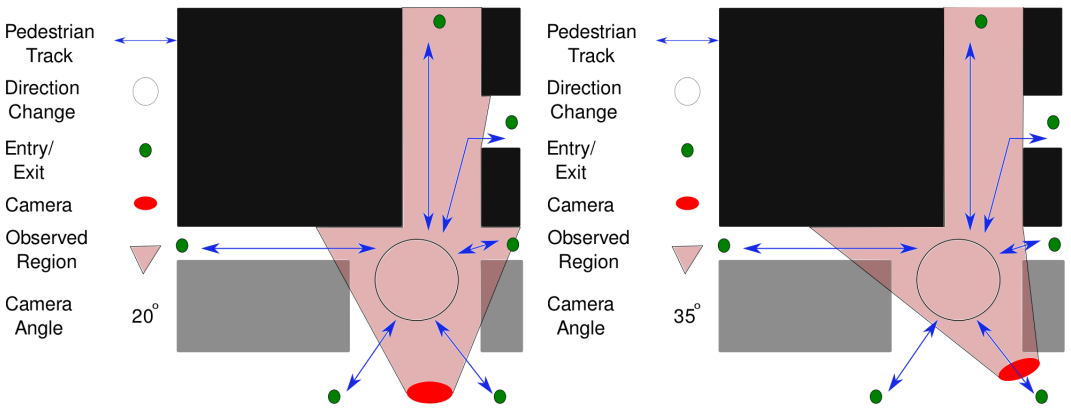
4.1.1 Ground Truth for the Dataset

The infrared video recorded at the above locations edited into shorter sequences. These sequences were then grouped according to the pedestrian density, the number of cold and warm targets and the number of occlusion events. To generate the ground truth these sequences were annotated by manually fitting bounding boxes to pedestrians. The ViPER [2] ground truth authoring and viewing tool was used to manage the meta-data for the sequences the same tool was also used to measure the accuracy metrics for the sequences.

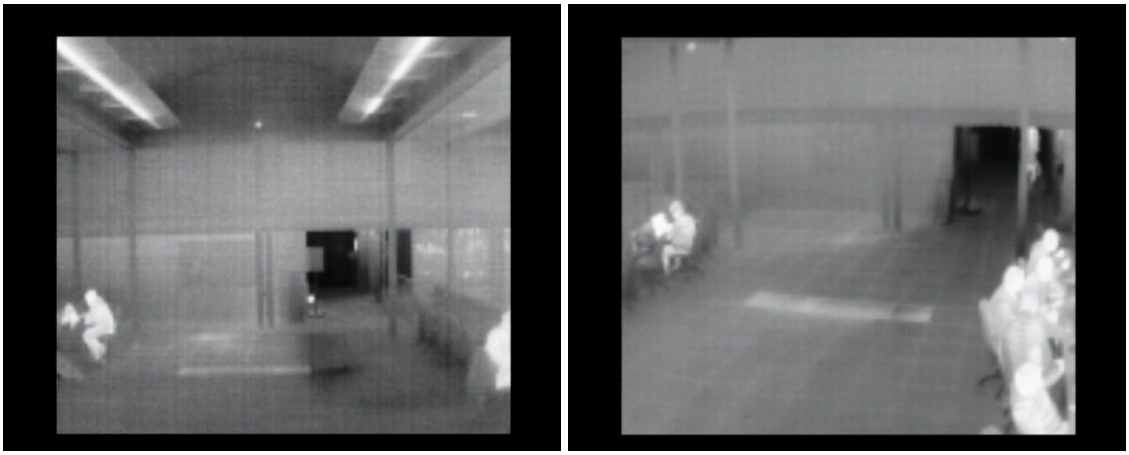
The annotated sequences were grouped into two sets of sequences; the first set consists of sequences with few pedestrian targets (at most 4 pedestrians), these sequences were exclusively used during the design of the classifier and the tracker and were **not** used

[1] http://www.cse.ohio-state.edu/otcbvs-bench/

[2] website: viper-toolkit.sourceforge.net



(a) (b)



(c) JKCC Camera @ 2.5m (d) JKCC Camera @ 4m

Figure 4.1: Indoor Sequences Layouts

for collecting performance metrics. The second set of sequences was used to collect performance and accuracy metrics.

4.1.2 Qualitative Sequences in Dataset

The first dataset as stated previously was used to develop the classifier and tracker and had the following target types:

Room Temperature Targets: As the dataset was collected indoors, some pedestrians are at room temperature. These were used to evaluate the classifiers ability to work with eroded targets. Targets at room temperature are usually eroded in infrared image sequences, as these pixels are more difficult for the background model to classify.

Cold Targets: Some of the dataset was recorded in winter, when the outdoor temperature was below room temperature. Due to this some of the pedestrians are wearing clothing below room temperature and consequently have less texture information.

Occlusion Recovery: Image sequences with long term (> 5 seconds), short term (< 5 seconds) occlusion events.

Table 4.1 groups the sequences in the dataset using the following attributes:

- N_s Number of sequences.
- T_n Total number of targets.
- T_c Number of targets below room temperature .
- T_w Number of targets at or above room temperature.
- T_o Minimum number of occlusion events.

Table 4.1: Dataset for Development

N_s	T_n	T_c	T_w	T_o	*Notes*
16	1	-	1	-	Single warm target
17	1	1	-	-	Single cold target
20	2	-	2	1	Occlusion event with two warm targets
23	2	1	1	-	Independent cold and warm targets
18	-	1	1	1	Warm target occludes cold target
17	-	-	1	1	Partial occlusion events
19	-	2	2	2	Long term occlusion events

4.1.3 Quantitative Sequences in Dataset

The sequences from this dataset were used to produce the metrics presented in this document. The sequences in the dataset are grouped according to the pedestrian density in the sequence. The dataset comprises of 45 sequences as outlined in table 4.2. In which,

- N_s Number of sequences.
- T_{mn} Minimum number of pedestrians for sequences in this group.
- T_{mx} Maximum number of pedestrians for sequences in this group.

As the location used to record the video has uncontrolled pedestrian access, multiple sequences with exactly the same pedestrian densities were not recorded. Instead, sequences with near identical pedestrian densities are used to measure the effect of pedestrian density on system performance.

4.1.4 Pedestrian Density

Pedestrian density refers to the number of pedestrians in at-least 65% of the frames in a sequence. So, if the pedestrian density for a sequence is 5, and the sequence has 20 frames. Then 15 frames of that sequence have 5 pedestrians present in them. The 65% rule is to allow for natural entry and exit of pedestrians into the field of view of the camera at higher pedestrian densities.

Table 4.2: Dataset for Performance Evaluation

N_s	T_{mn}	T_{mx}
12	4	6
13	9	11
10	14	16
10	19	21

4.2 Performance of Pedestrian Tracking Systems from Literature

The literature review in chapter 2 examined the current state of the art for pedestrian detection and tracking with infrared images. As identified in section 2.1, not all trackers implement a discrete background subtraction, pedestrian classifying and object tracking mechanisms. As the proposed novel algorithms implement pedestrian classification and tracking independently, benchmarks for the performance of classifiers and a tracker from literature are necessary.

Table 4.3: Systems from Literature

	Segmentation	Detection	Tracking
Probabilistic Template [42]	Intensity Thresholding	Template	Kalman Filter
Histogram Symmetry [13]	Intensity Thresholding	Histogram	N/A
Principal Component Analysis (PCA) [41]	Expectation Maximisation-Algorithm	Support Vector Machine + Principal Component Analysis (PCA)	Hausdorff Distance

The following three pedestrian classifiers and a pedestrian tracker were implemented as they are representative of the state of the art. The classifiers are the probabilistic template model proposed by Olmeda [42], the histogram symmetry based technique proposed by Bertozzi [13] and finally the principal components analysis (PCA) model proposed by Dai [41]. The background model, pedestrian detection and tracking mechanisms used by the authors are listed in table 4.3.

The first is a trained template approach, the second uses the pedestrian symmetry, aspect ratio and the properties of infrared images (with the assumption that pedestrians are warmer than the background). The final system identifies pedestrians by looking for identifying pedestrian components within the foreground regions.

In a large body of work with pedestrian tracking in infrared image sequences assumes that a pedestrian target will be higher in intensity than the background (table 4.3 and section 2.3). But, as the dataset was collected indoors some of the targets are colder than the background (pedestrians entering a warm building on a cold day). Therefore, the algorithms may need to be modified to be able to classify and track these targets.

4.2.1 Background Model

In table 4.3, two systems use intensity thresholding to identify foreground regions, this, will not work with the dataset recorded as all pedestrian targets are not above ambient temperature. To identify these targets a more flexible background model was necessary. Using information from a background model review published by Chalidabhongse et al [12], a Gaussian background model was used. This model serves two purposes; one, as a

background model and secondly as a noise filter as identified by Dai et al [41].

4.2.2 Probabilistic Template - Modifications

As the texture content for warm pedestrians differs from the texture content for cold targets (section 3.0.1.1), the probabilistic template based detector was modified to use two trained templates, one for targets that are warmer than the ambient (T_w) and a second template for targets that are colder (T_c) than the ambient. Both templates are trained using a training set of 100 manually marked sample images, 50 warm and 50 cold pedestrians.

To identify the appropriate template for a foreground regions, the mean intensity of the foreground region $i_{l,m}$ is checked against the mean intensity for the image $i_{x,y}$ as follows:

$$i_{l,m} = \sum i_{a,b}/(l \times m) \tag{4.1}$$

$$i_{x,y} = \sum i_{a,b}/(x \times y) \tag{4.2}$$

$$\text{the mean intensity ratio, } i_r = i_{l,m}/i_{x,y} \tag{4.3}$$

$$\text{classification template (equation 2.32) } \begin{cases} T_w; 1 \leq i_r \\ T_c, \text{ otherwise} \end{cases} \tag{4.4}$$

In equation 4.3 the mean foreground intensity ($i_{l,m}$) is in the numerator, to avoid a divide by zero error when the foreground region is completely black. Additionally, as the mean intensity check takes less time than matching the foreground region to a template, choosing the appropriate template reduces the time taken to process a sequence with mixed targets.

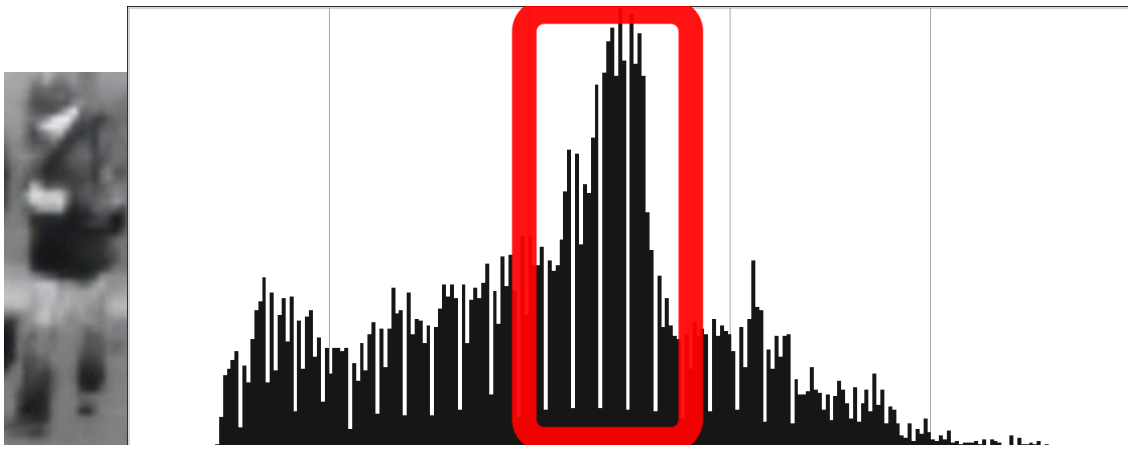
4.2.3 Histogram Symmetry - Modifications

The histogram symmetry measure used by Bertozzi et al [13], assumes that foreground regions that are in the higher intensity ranges are valid targets for pedestrian detection. Some of the pedestrians in the dataset are colder than the ambient room temperature. The histogram for a target that has varying grey levels, but is mostly colder than then ambient is shown in figure 4.2(b).

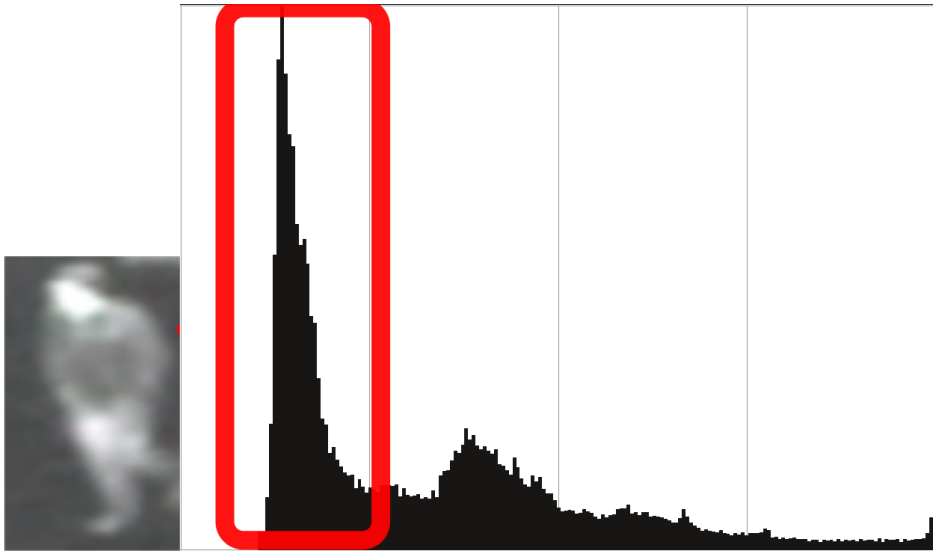
When this is compared to the histogram of a warm pedestrian (figure 4.2(d)) they are similar, sharp peaks in foreground regions with valid pedestrian targets. Highlighted by red boxes in figures 4.2(b) and 4.2(d). These peaks are a proxy for a target with pedestrian appearance (pedestrian aspect ratio, [13] and section 2.3). As this demonstrates that pedestrian targets colder than the ambient can be identified by the stock algorithm, no modification was necessary.

4.2.4 Principal Component Analysis - Modifications

Dai, Zheng and Li [41], use a combination of Support Vector Machine (SVM) and Principle Component Analysis to identify and localise pedestrians in bounding boxes. In



(a) (b)



(c) (d)

Figure 4.2: Histograms for Targets Above and Below Ambient

this dataset, all foreground regions contain pedestrians and the challenge is separating individual pedestrians under occlusion. Hence, this implementation will use only PCA classification to localise the pedestrians in foreground regions.

To identify individual pedestrians in foreground regions, two pedestrian templates, 110×40 and 60×24 pixels in size were generated. Both the templates were generated using the same, manually labelled, sample images used to train the probabilistic template. As in the original implementation, a 30% overlap was used to merge targets.

4.2.5 Tracking Module used for Benchmarking

As we can see in table 4.3, different tracking algorithms were used with the classifiers in literature. As the goal is to evaluate classifier performance with infrared image sequences, a simple but effective tracker is sufficient. While a bounding box tracking [49] algorithm would have been effective, Fuentes et al [77] described useful enhancements that helped with occlusion recovery.

The tracker is a bounding box tracker that uses pixel intensity information to recover from occlusion events. The bounding boxes and the contents of the two blobs (foreground regions with valid pedestrian targets), are resized so they are the same size and binary foreground matrices are calculated for the bounding boxes. The binary matrices are generated by setting every foreground pixel to 1 and the background pixel to 0 in the bounding box.

The two binary matrices are then XOR'ed and the sum of the elements of the resulting matrix is used to check if there is a match between the two bounding boxes. The implementation of the tracker is described next.

To match two overlapping blobs i and j in consecutive frames $X_{x,y}$ and $X+1_{x,y}$, the bounding boxes of the blobs $i = B_{a,b}$ and $j = B_{c,d}$ are modelled as matrices. When an object is moving, the area of the bounding box changes between frames, the larger of the two bounding boxes ($B_{a,b}$ or $B_{c,d}$) is resized so that they are both the same size.

Resizing the larger bounding box compresses the texture content of that box, this is favoured over the option of enlarging the smaller of the two bounding boxes as the interpolation required to enlarge the smaller bounding box increases the noise in the foreground. After the resize operation, the foreground pixels in the matrices are ex-ored to match the two blobs. This illustrated below:

Assuming that the area of $B_{c,d} > B_{a,b}$, the bounding box $B_{c,d}$ is resized to generate $R'_{a,b}$. From the two ($R'_{a,b}$ and $B_{a,b}$), two new binary matrices are generated $bm1_{a,b}$ from

$B_{a,b}$ and $bm2_{a,b}$ from $R'_{a,b}$ where,

$$bm1_{a,b} = \begin{cases} 1, \text{ if, } B_{a,b} \text{ is part of the foreground.} \\ 0 \end{cases} \tag{4.5}$$

$$bm2_{a,b} = \begin{cases} 1, \text{ if, } R'_{a,b} \text{ is part of the foreground.} \\ 0 \end{cases} \tag{4.6}$$

$$\tag{4.7}$$

To verify if the two blobs match, the two binary matrices are ex-or'ed:

$$E'_{a,b} = bm1_{a,b} \oplus bm2_{a,b} \tag{4.8}$$

$$B_{a,b} \equiv B_{c,d} \begin{cases} \text{if, } \left(\frac{\Sigma E'_{a,b}}{a \times b} \right) < T_1 \\ 0, \text{ otherwise} \end{cases} \tag{4.9}$$

To resolve multiple matches during occlusion events, a pixel intensity measure derived from a similar mechanism described by Fuentes et al [77] was used. The mechanism as described, uses the colour information for the pixels in the bounding boxes. It was modified and adapted as only pixel intensity is recorded in infrared images.

If, on the other hand for a bounding box j in X there are multiple overlapping bounding boxes $i_1 \ldots i_w$ from frame $X+1$, all the boxes are resized as described above and the intensity match is generated as follows:

$$A = \sum \begin{cases} 0, \text{ if the absolute value of, } (B_{a,b} - R'_{a,b}) \leq T_2 \\ 1, \text{ otherwise} \end{cases} \tag{4.10}$$

$$\text{again, } bm1_{a,b} \equiv bm2_{c,d} \text{ if, } \frac{A}{a \times b} < T_1 \tag{4.11}$$

This generates a set of A values, $A_1 \ldots A_w$, of which the bounding box for which the lowest A value was generated is considered the best match. If $A \neq 0$ then the bounding boxes are found to match, and the track data is updated.

When no matching blobs are found in the new frame $(X+1_{x,y})$ for a blob present in the initial frame $(X_{x,y})$ then the blob is assumed to have been lost and is marked and after 25 frames, dropped. Any blobs in the frame that do not match any existing blobs are considered as new targets and are added to the set of targets for the next frame in the sequence $X+2_{x,y}$.

The two threshold values used in equations 4.9, 4.10 and 4.11 (T_1 and T_2) are percentage thresholds. The first threshold value T_1 is frame rate sensitive, for the pedestrians in the dataset T_1 was set at .20.

The second threshold value, T_2 is set according to the environment at which the images are recorded and the variance between the targets. As the pedestrians in the image sequence are recorded with a wide range of temperatures, this threshold was set at .3.

Table 4.4: Systems as Implemented

	Segmentation	Detection	Tracking
Probabilistic Template (P.T) [42]	Gaussian Background Modelling	Template	Bounding Box Tracker
Histogram [13]	Gaussian Background Modelling	Histogram	Bounding Box Tracker
Principle Component Analysis (PCA) [41]	Gaussian Background Modelling	Principle Component Analysis (PCA)	Bounding Box Tracker

Table 4.5: Hardware Platforms

	PC	**PS3**	**Wii**
Processor	x86-64x2 @2GHz	Cell @3.2Ghz	PPC @729Mhz
Ram	2 Gb	512 Mb	64 Mb
Notes	General Purpose	Vector Processor	Resource Constrained

4.3 Evaluation Methodology

Table 4.3 is a summary of the pedestrian classification and tracking systems as described in literature. However, for the tracking system to work with the dataset outlined in section 4.1, the classifiers and trackers were modified as described above (section 4.2). Table 4.4 summarises the pedestrian tracking systems that were implemented.

The metrics described in the literature review sections 2.3.3 and 2.4.5.1 are used to collect the accuracy performance data for the system. The classifier and tracking system was implemented on three hardware platforms, a pc with a dual core x86 based processor, a Playstation 3 and a Nintendo Wii. The specifications for the three platforms are summarised in table 4.5.

The three hardware platforms will help find an answer for the third research question,

> Is using the time taken to process a single target an effective metric to predict the performance of a tracking system for a specified target (pedestrian) density?

If only one hardware platform was used for this experiment, then the results are relevant only to hardware platforms of the same type. The three hardware platforms used in this evaluation on the other hand are quite dissimilar, the x86 platform is representative of general purpose systems, the Playstation 3 a specialist hardware platform with a fairly powerful processor but limited memory and finally the Wii a resource restricted system with low power processor and limited memory.

The aim is to identify if any link exists between pedestrian density and the frame rate at which a sequence is processed. Similar links could potentially be identified between pedestrian density and memory usage, or, pedestrian density and processor utilisation. As we are interested in the overall frame rate, these metrics were not recorded and no analysis was done on the same.

The results after processing the 45 sequences of the quantitative dataset (section 4.1.3) with the systems in table 4.4 are grouped into two types. They are:

Accuracy Baseline : The baseline performance metrics are the traditional metrics that have been used with pedestrian tracking algorithms. Previously reviewed in sections 2.3.3.1 and 2.4.5.1.

Frame Rate Performance : Baseline statistics measure the performance of the entire system both the tracking system on the whole and the classifier. However, no difference is expected for these metrics between the three hardware platforms. The frame rate performance, however, will depend on the compute resources available on the platform.

4.4 Performance - System Accuracy

The third research question addresses the relationship between pedestrian density and the speed (frame rate) at which a sequence is processed. Accuracy metrics for the classifiers and the bounding box tracker were also collected, these will be used to evaluate the performance of the novel classifier and tracker.

As stated previously, the sequences in the dataset contain only of pedestrian targets (section 4.1). Metrics for classifiers have been described in section 2.3.3. The effects of a dataset with only pedestrian targets on these metrics is described below.

Of the four metrics described, *true positive* (TP_c), *true negative* (TN_c), *false positive* (FP_c) and *false negative* (FN_c), two of the four are not relevant for use with the image sequences in the dataset. These are TN_c and FP_c rates, because all the targets in an image sequence are pedestrians.

Additionally, as all the targets in the image sequence are pedestrians, the TP_c rate and the FN_c rates will be complementary. That is, a change in the TP_c rate will have a proportional change in the FN_c, as there are no non-pedestrian targets. From published literature [13, 42, 41], we know that the accuracy for the classifiers is at-least 70%.

Ground Truth Overlap:
A less than 80% overlap between the ground truth location for an object and the bounding box was recorded as false negative. The results for the classifiers are in tables 4.7 to table 4.10, where, $\bar{TP_c}$ is the mean true positive rate for N_s sequences, with pedestrian density between T_{mn} and T_{mx} (from table 4.2), $\hat{TP_c}$ the median true positive rate and σ_T the standard deviation for the sequences in the set.

No difference was found in the accuracy of the classifiers on the three hardware platforms. To calculate the TP_c values for a set of sequences, find the average pedestrian density for the set,

$$T_{av} = \frac{T_{mn} + T_{mx}}{2}$$

$$TP_c = 1 - \frac{\bar{T}}{T_{av}} \tag{4.12}$$

$$\sigma^2_{TP_c} = 1 - \frac{\sigma^2_T}{T_{av}} \tag{4.13}$$

The general trend is that the accuracy of the classifier degrades as the pedestrian density increases, sequences where the pedestrian density is between 10 and 15, have the most misclassified pedestrians (figure 4.3).

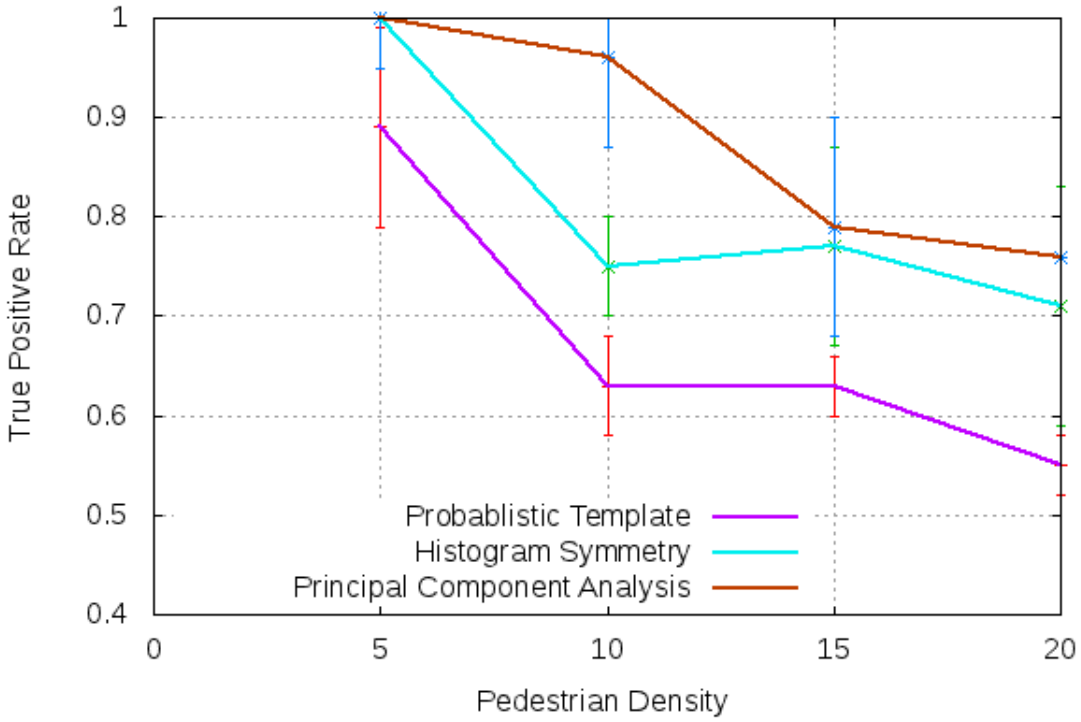
The TP_c curve of the classifier makes the tracking systems performance seem worse than it actually was, more complete picture emerges when the track detection rate T_{dr} is examined. The ROC curve for the T_{dr} is in figure 4.3(b), this was generated from the performance of the tracking system from table 4.6. Again, there was no difference in track detection rates for the three hardware platforms.

4.4.1 Classifier Performance - Probabilistic Template

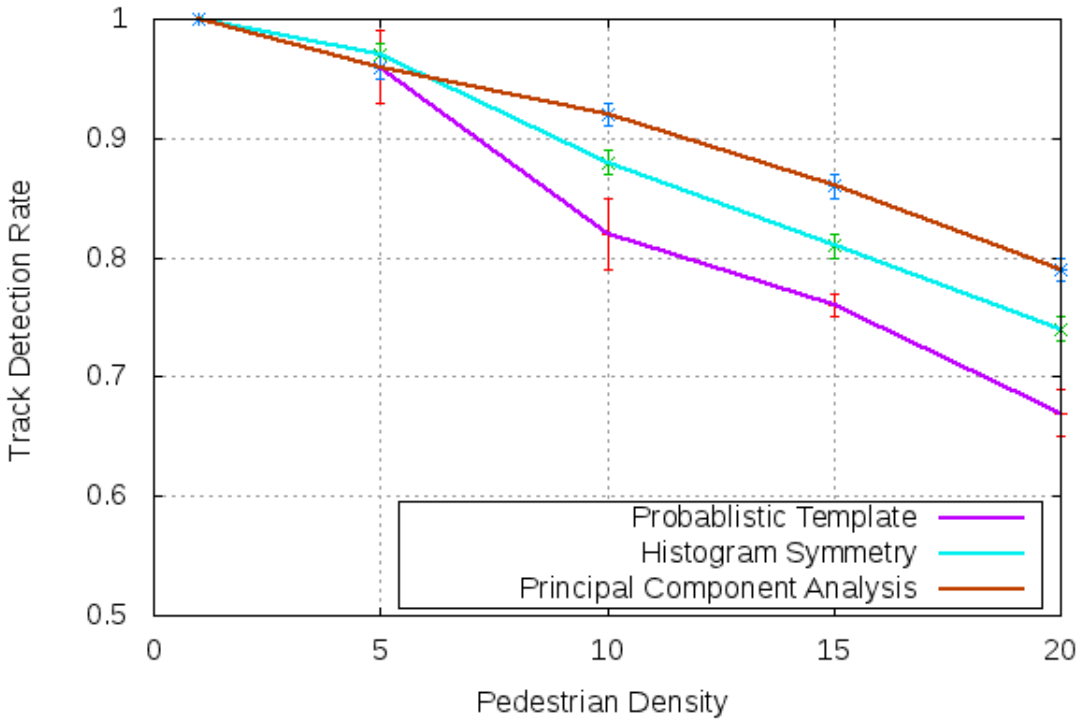
When the pedestrian density in a sequence is between 10 and 15, this classifier had the most misclassified pedestrians ($TP_c = 0.60$ in figure 4.3, table 4.7). This means that the classifier misclassified 40% of pedestrians for these sequences. Using the individual pedestrian sequences described in section 4.1.2, it was found that this classifier was failing when there were partial occlusion events.

Figure 4.4 will be used to illustrate a sample failure event. Figure 4.4(a), is the ground truth; dark green boxes are for non-occluded pedestrians, the lighter green boxes are for partial/occluded pedestrians. The blue arrows indicate the two pedestrians that are misclassified, figure 4.4(b) shows the three pedestrians more clearly.

Figure 4.4(c) is generated after background segmentation. When the template is matched to this foreground region, only one of the three present classified as a valid pedestrian the other two are misclassified as false negatives. As the pedestrian density in the sequence increases, these events become more frequent, hence the FN_c rate increases.



(a) The True Positive Rate TP_c Graph (tables 4.7, 4.8 and 4.10)



(b) The Track Detection Rate T_{dr} Graph (from 4.6)

Figure 4.3: Graphs for Data in Tables 4.7, 4.8, 4.10 and 4.6

Table 4.6: System performance - Track Detection Rate T_{dr} (from table A.2)

Classifier	$\bar{T_{dr}}$	$\hat{T_{dr}}$	$\sigma_{T_{dr}}$	T_{mn} and T_{mx}	N_s
P.T	0.96	0.96	0.02		
H.S	0.96	0.96	0.02	**4-6**	12
P.C.A	0.97	0.97	0.01		
P.T	0.81	0.82	0.02		
H.S	0.91	0.90	0.01	**9-11**	13
P.C.A	0.91	0.91	0.02		
P.T	0.75	0.74	0.01		
H.S	0.84	0.84	0.02	**14-16**	10
P.C.A	0.86	0.86	0.01		
P.T	0.67	0.67	0.02		
H.S	0.76	0.76	0.02	**19-21**	10
P.C.A	0.79	0.80	0.02		

P.T: Probabilistic Template classifier from [42]

H.S: Histogram symmetry classifier from [13]

P.C.A: Principle component analysis classifier from [41]

The change in the TP_c curve for the probabilistic template classifier is steep (figure 4.3(a)), but the change in the T_{dr} curve for the tracking system is more gradual (figure 4.3(b)). When the two curves are compared, they appear to contradict each other. *As only about 60% of pedestrian targets are classified correctly for in sequences with pedestrian densities between 14 and 16 resulting in a TP_c rate of* .6*, how can the system have a T_{dr} rate of* .82*?*

Analysis showed that this was because the classification failure was intermittent, the tracker is able to maintain track of an object without changing the systems T_{dr} rate. The following example illustrates this:

For a sequence of 20 frames, with 5 pedestrians each,

and a total of $20 \times 4 = 80$ ground truth points were processed by the tracking system.

Three of the pedestrians in the sequence were in an occlusion event; and the classifier was unable localise them for 3 frames,

the number of ground truth points not found: $3 \times 3 = 9$.

But the tracker is able to recover after the occlusion event, thus, no tracks are lost. So, for the sequence the $T_{dr} = 1$,

the T_P rate on the other hand: $TP_c = \frac{80-9}{80} = \frac{71}{80} = .88$.

This example illustrates how the T_{dr} curve is to some extent independent of the TP_c

Table 4.7: Probabilistic Template Classifier - Accuracy TP_c Rates (from table A.1)

$\bar{TP_c}$	$\hat{TP_c}$	σ_{TP_c}	T_{mn} and T_{mx}	N_s
0.89	0.80	0.10	**4-6**	12
0.63	0.60	0.05	**9-11**	13
0.63	0.60	0.03	**14-16**	10
0.55	0.55	0.03	**19-21**	10

curve, which is the case for the tracking system using the probabilistic template classifier. This example also illustrates that even if a classifier is unable to localise pedestrians in the foreground, using a robust tracking algorithm improves overall system accuracy.

4.4.2 Classifier Performance - Histogram Symmetry

The performance profile for the histogram symmetry classifier is similar to that of the probabilistic template tracker (table 4.8). However, from figures 4.3(a) and 4.3(b) we can see that this classifier has fewer intermittent failures when compared to the probabilistic template classifier which improves overall system accuracy.

Table 4.8: Histogram Symmetry Classifier - Accuracy TP_c Rates (from table A.1)

$\bar{TP_c}$	$\hat{TP_c}$	σ_{TP_c}	T_{mn} and T_{mx}	N_s
1.00	1.00	0.00	**4-6**	12
0.75	0.80	0.05	**9-11**	13
0.77	0.73	0.10	**14-16**	10
0.71	0.65	0.12	**19-21**	10

The failure mode identified for this classifier was that just before one target completely occluded another, the physical location of these pedestrians in relation to the camera lead to the histograms for that foreground region to have an empty center. From section 2.3.2.1, we know that when the histogram for a foreground region has an empty center, it is classified as a false target. Figure 4.5(a) is one frame in which this failure occurred.

As this event only occurs in one or two frames bracketing an occlusion event (when the pedestrians occlude at specific angles), the overall accuracy of this classifier was better than that of the probabilistic template classifier (figure 4.3(a)).

4.4.3 Classifier Performance - Principal Component Analysis

The PCA template classifier had the best accuracy metrics for the dataset (table 4.10). The reason for this is that the template generation mechanism is more efficient storing pedestrian template features and matching them to the pedestrian foreground regions in

(a) Ground Truth

(b) Targets

(c) Segmented Foreground

Figure 4.4: Probabilistic Template Classifier failure

(a)

Figure 4.5: Example of Histogram Symmetry Classifier Failure

Table 4.9: System performance - Speed on PC (from table A.4)

Classifier	$\bar{S_{pc}}$	$\hat{S_{pc}}$	$\sigma_{S_{pc}}$	T_{mn} and T_{mx}	N_s
P.T.	20.33	20.00	1.15		
H.S.	19.25	19.50	0.87	**4-6**	12
P.C.A	16.42	16.00	0.51		
P.T.	17.00	17.00	0.91		
H.S.	17.69	18.00	0.48	**9-11**	13
P.C.A	15.31	15.00	0.48		
P.T.	14.50	14.50	0.53		
H.S.	14.80	15.00	0.42	**14-16**	10
P.C.A	13.70	14.00	0.48		
P.T.	11.40	11.00	0.52		
H.S.	11.80	11.50	0.92	**19-21**	10
P.C.A	9.70	10.00	0.48		

P.T: Probabilistic Template classifier from [42]

H.S: Histogram symmetry classifier from [13]

P.C.A: Principle component analysis classifier from [41]

S_{pc}: Speed of system on a **PC**.

the dataset. Unlike the previous two classifiers the PCA classifier had no distinct failure modes, instead the drop in accuracy was due to an increase in the number of misclassified pedestrians as the template failed to classify pedestrians during partial occlusion events, where more than 50% of one pedestrian is occluded.

Table 4.10: Principal Component Analysis Classifier - Accuracy TP_c Rates (from table A.1)

$\bar{TP_c}$	$\hat{TP_c}$	σ_{TP_c}	T_{mn} and T_{mx}	N_s
1.00	1.00	0.00	**4-6**	12
0.96	1.00	0.05	**9-11**	13
0.79	0.73	0.09	**14-16**	10
0.76	0.70	0.11	**19-21**	10

4.4.4 Tracker Performance - Accuracy and Pedestrian Density

The bounding box tracker is effective at tracking pedestrian targets in infrared image sequences. The bounding box pixel intensity match described in equation 4.11 was found to reasonably accurate at resolving occlusion events.

As the pedestrian density increased the frequency of occlusion events with four or more pedestrians increased, where one failure mode of the tracker was identified. During occlusion events with more than four pedestrian targets, the tracker is unable to maintain track of last two targets to be occluded. After the occlusion event, it classifies them as new targets.

The cause was identified as foreground model maintained for a bounding box by the tracker. Normally the foreground model updates after matching in every frame. During short occlusion events, the change in appearance of a pedestrian is small and the foreground model for a bounding box is still usable.

For occlusion events involving four or more pedestrians, it was found that the change in the appearance of the pedestrian was sufficient to render the model ineffective, which leads to the bounding box for a pedestrian being marked as lost. Decreasing the T_{dr} rate for the tracking system. The effect of this failure mode, was mitigated by the fact that the camera was positioned overhead when recording the dataset. This reduced the number of frames in a sequence where the entire pedestrian was occluded, usually the upper third (head and neck) of a pedestrian were only occluded for short periods of time ($\approx 10-15$ frames).

The above condition (head and neck not long term occluded) is only true for sequences with low pedestrian densities, for image sequences with higher pedestrian densities the accuracy of the tracking system falls (figure 4.3(b)).

4.4.5 Classifier Performance - Summary of System Accuracy

In section 2.3.3, we identified that for infrared video sequences templates can fit a broader number of pedestrian targets as there are fewer surface texture features. Conversely, the same section also identified that templates have higher failure rates when validating pedestrians.

The accuracy results in figure 4.3(a) for the classifiers reflects both these conditions. The probablistic template model is unable to accurately model all possible pedestrian shapes hence has higher failure rates; whereas, the PCA template model is better able to model pedestrians in the image sequence. Additionally, the PCA model has the advantage of not having a single distinct failure mode which helps improve accuracy.

Also, we identified a unique failure mode for the tracker which will help improve the design of the novel tracker described later in this document.

4.5 System Performance - Processing Speeds

In the previous section, the accuracy of the classifiers as well as the overall accuracy for the tracking system was evaluated. This information will be useful in validating the proposed classifier and tracking algorithms in the next chapter. But it does not help answer the question:

> Is using the time taken to process a single target an effective metric to predict the performance of a tracking system for a specified target (pedestrian) density?

Recording the frame rates at which sequences in the dataset are processed by the pedestrian tracking systems, on the three hardware platforms; will generate the required frame rate observations that can be tested to establish whether or not there is a statistically significant correlation between pedestrian density and frame rate.

4.5.1 Frame Rate Performance - All Hardware Platforms

The baseline frame-rate for a platform is the speed at which the system processed the image sequence and generated the track data. For a set of sequences from the dataset, the speed data is recorded here as the mean ($\bar{S_X}$), median ($\hat{S_X}$) and standard deviation (σ_{S_X}) of the frame rates. This is repeated for all three hardware platforms i.e. *PC*, *PS*3 or *Wii*, the X in the previous sentence refers to the hardware platform.

This information is in table 4.9 for the PC, table 4.11 for the PS3 and table 4.12 for the Wii. The frame rate data for sequences is in Appendix A.

As the data in the tables is difficult to interpret, using the variance as the error line and the median for a set of sequences as the plot value for that set, a combined frame rate

graph was generated. The graph (figure 4.7) shows the frame rates for the three pedestrian tracking systems on all the hardware platforms.

From the graph we can see that all three hardware platforms maintain reasonable frame rates ($>$ 10 frames-per-second (*fps*)) as long as the pedestrian density is less than 10. As the pedestrian density increases, the frame rate for the tracking system decreases. The slowest hardware platform, as expected, is the Wii. The lowest frame rate recorded is 2.5 frames a second for image sequences with at-least 19 pedestrians.

The overview of the change in frame rate (figure 4.7), paints a general picture and is not very useful. Instead, when the speed of the trackers was grouped according to the classifier used by the tracking system a more nuanced picture emerged.

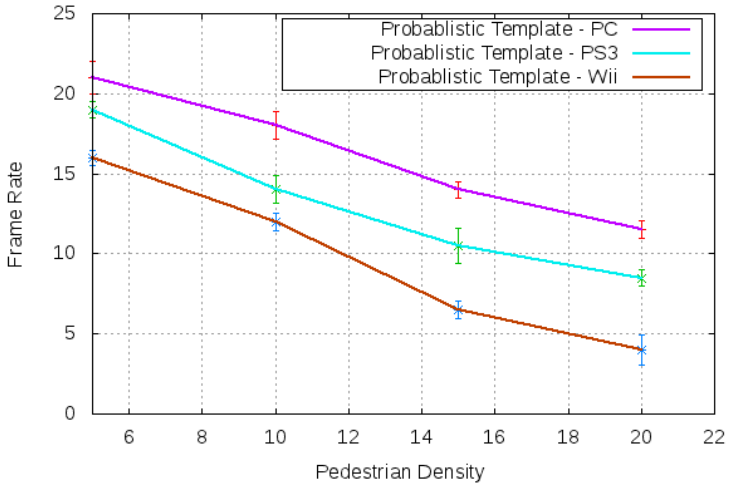
4.5.2 System Frame Rate - Probabilistic Template Classifier

The plots in figure 4.6(a) are the frame rates recorded for the tracker on the three hardware platforms at different pedestrian densities. From the image, we can see that the difference between the frame rates for the three hardware platforms is relatively constant. As the pedestrian density in the sequence increases, the frame rate decreases, but the difference in frame rates between the three platforms stays relatively constant. The throughput is as expected, the frame rates on the PC are the highest. The Wii has the lowest frame rates with the PS3 in between.

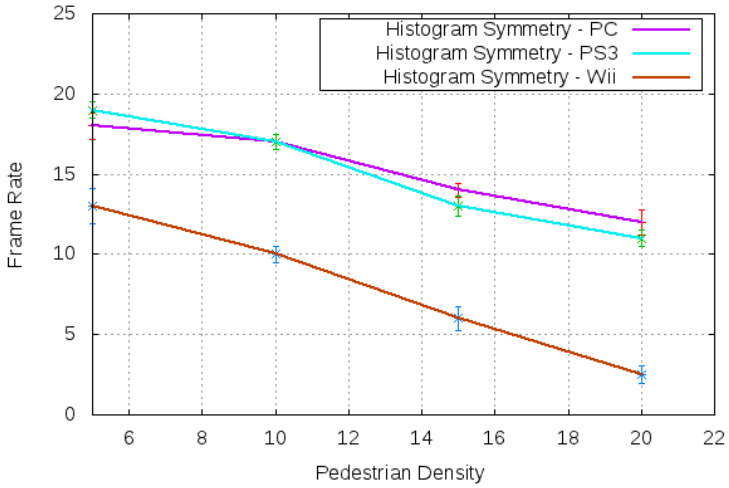
4.5.3 System Frame Rate - Histogram Symmetry Classifier

The frame rate graph for the tracking system using this classifier is in figure 4.6(b). From this figure we can see that unlike the previous tracking system, the throughput of the tracker on the PS3 with this classifier at low pedestrian densities is better than that on the PC. Additionally, the frame rate for tracking system using this classifier on the PS3 does not fall below 10 *fps* even at higher pedestrian densities. This indicates that the histogram symmetry classifier requires less time to evaluate foreground regions on the PS3.

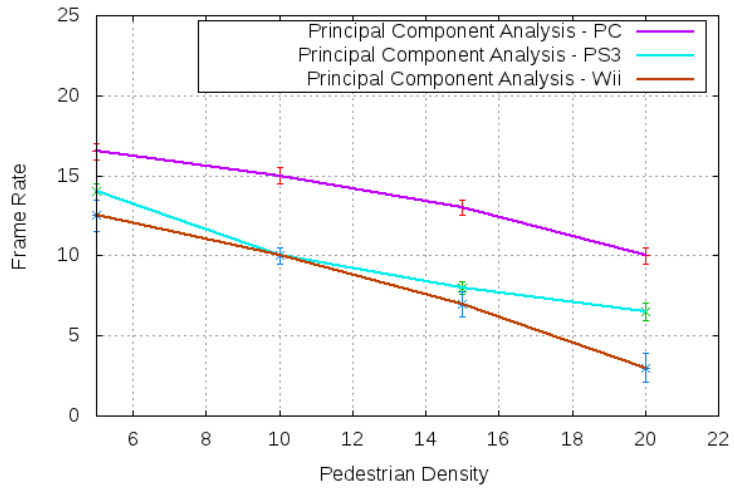
On the PC and the Wii, the response of the tracking system using this classifier, is similar to the response of the tracking system using the probabilistic template and principal component analysis classifiers (figures 4.6(a) and 4.6(c)). Unlike the tracker using the probablistic template classifier, the difference in frame rate changes between platforms is not constant.



(a) Frame Rates for Tracker using Probabilistic Template Classifier



(b) Frame Rates for Tracker using Histogram Symmetry Classifier



(c) Frame Rates for Tracker using Principal Component Analysis Classifier

Figure 4.6: Frame Rates for Individual Trackers

4.5.4 System Frame Rate - Principal Component Analysis Classifier

In section 4.4.3, we identified that for this dataset, the tracker when using the principal component analysis classifier, was the most accurate at identifying and tracking pedestrians. Consequently the tracker had a high T_{dr} rate (figure 4.3b), on examining the frame rates for the tracking system using this classifier, figure 4.6(c) we see that this was the slowest on all three platforms. Like the tracker using the histogram symmetry classifier, the change in frame rate is not linear nor is it constant between hardware platforms.

4.5.5 System Frame Rate - Summary

The throughput on the three hardware platforms decreases as the pedestrian density of the sequence increases (figure 4.7). However, the drop in frame rate for all three pedestrian trackers is not linear (figures 4.6(c) and 4.6(b)).

Of the three trackers implemented, the tracker using the probabilistic template classifier has the most consistent response to pedestrian density changes (figure 4.6(a)). This means that the time taken to process a single pedestrian target stays constant at all pedestrian densities for this tracker.

Of the three pedestrian trackers, the tracker using the histogram symmetry classifier, had the highest throughput on both the PC and the PS3. At lower pedestrian densities the throughput for this tracker on the PS3 is better than the throughput for this tracker on the PC (figure 4.6(b)).

The third research question from section 1.5.1,

> Is using the time taken to process a single target an effective metric to predict the performance of a tracking system for a specified target (pedestrian) density?

The answer to the above will be a qualified , **no**. The time taken to process a single pedestrian target on a hardware platform, cannot be used to predict the performance of all pedestrian tracking system at a specified pedestrian density. For some pedestrian tracking systems, it might work. However, as only one tracking system used in this evaluation had a consistent response to the change in pedestrian density there is insufficient information to generalise it to any particular genera of pedestrian tracking systems (section 4.5.2).

As stated in the introduction, the speed is used as a proxy for the compute resource utilisation by a pedestrian tracking system. On the PC and PS3 the histogram symmetry based tracker required fewer compute resources. The most accurate tracker on the other hand required the most resources, this is reflected in the low throughput for the same (figure 4.6(c)).

4.6 Summary - Performance of Existing Algorithms

This chapter reviewed the performance of existing algorithms from literature. To do this, a dataset of 45 infrared image sequences was used. The ground truth meta-data for this dataset was managed using the ViPER toolkit and the sequences were grouped according to their pedestrian densities (section 4.1).

As some of the pedestrians in the dataset are colder than the background, the pedestrian classifiers from literature were checked and any modifications necessary were implemented and described (section 4.2). A simple bounding box tracker was then described and implemented to complete the pedestrian tracking system (section 4.2.5).

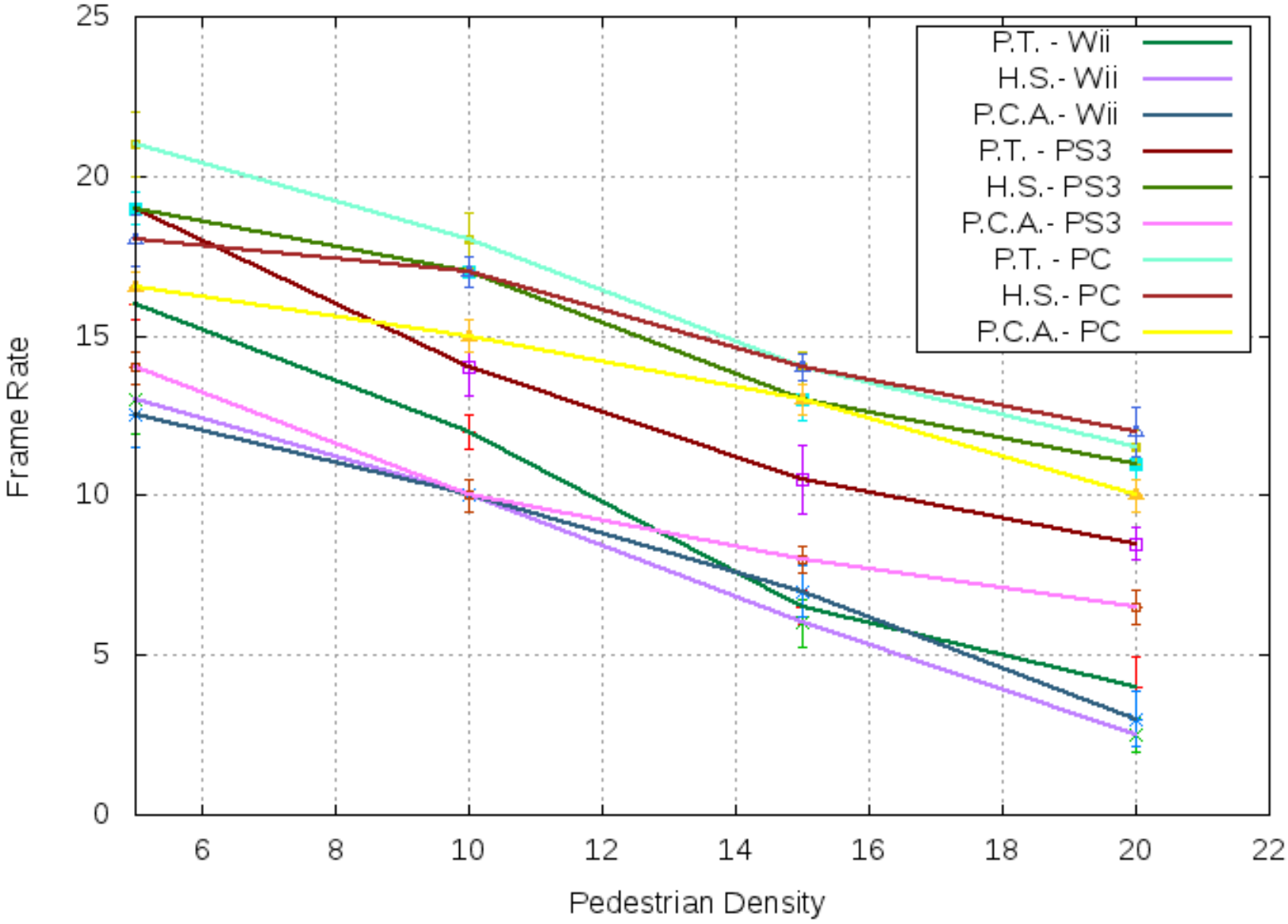
To ensure that these modifications did not affect the accuracy of the classifier, the accuracy metrics were collected for the three trackers (section 4.4). During this evaluation we identified and verified failure modes for two of the classifiers (sections 4.4.1 and 4.4.2). Overall the tracker using the principal component analysis classifier was found the most accurate.

After this, the throughput (frame rates) of these trackers on three hardware platforms were recorded. From this we found that the time taken to process a single pedestrian will not be an effective metric to predict the performance of all pedestrian tracking systems , but could be valid for use with one pedestrian tracking system (section 4.5.5).

4.6.1 Contribution to Body of Knowledge and Novelty

This chapter addressed the third of the six research questions from chapter 1. As this chapter addresses pedestrian classifiers and a tracker previously described in literature there isn't much novelty in the content of this chapter. However, with existing algorithms, this chapter has identified and demonstrated failure modes for two of the classifiers from literature that have a significant impact on their accuracy (section 4.4.1 and 4.4.2). The slowest classifier was identified as the most accurate of the three classifiers on all three hardware platforms.

Using the three hardware platforms we also collected data about the change in frame rate and its relationship to pedestrian density. This data was used to disprove the use a pedestrian density as a metric for use with two of the three tracking systems. As the linear relationship between pedestrian density and frame rate was evident in only one tracking system, the sample size is too small to be viable.



P.T - X: Probabilistic Template Tracker on a Wii, PS3 or PC

H.S. - X: Histogram Symmetry Tracker on a Wii, PS3 or PC

P.C.A. - X: Principal Component Analysis Tracker on a Wii, PS3 or PC

Figure 4.7: System Frame Rates on all Hardware Platforms

Table 4.11: System performance - Speed on PS3 (from table A.6)

Classifier	$\bar{S_{ps3}}$	$\hat{S_{ps3}}$	$\sigma_{S_{ps3}}$	T_{mn} and T_{mx}	N_s
P.T.	18.58	19.00	0.51		
H.S.	19.33	19.00	0.49	**4-6**	12
P.C.A	13.58	14.00	0.51		
P.T.	13.92	14.00	0.86		
H.S.	16.69	17.00	0.48	**9-11**	13
P.C.A	10.38	10.00	0.51		
P.T.	10.50	10.50	1.08		
H.S.	12.80	13.00	0.63	**14-16**	10
P.C.A	8.20	8.00	0.42		
P.T.	8.50	8.50	0.53		
H.S.	11.40	11.00	0.52	**19-21**	10
P.C.A	6.50	6.50	0.53		

P.T: Probabilistic Template classifier from [42]

H.S: Histogram symmetry classifier from [13]

P.C.A: Principal component analysis classifier from [41]

S_{ps3}: Speed of system on a **PS3**.

Table 4.12: System performance - Speed on Wii (from table A.8)

Classifier	$\bar{S_{wii}}$	$\hat{S_{wii}}$	$\sigma_{S_{wii}}$	T_{mn} and T_{mx}	N_s
P.T.	16.33	16.00	0.49		
H.S.	13.67	13.00	1.07	**4-6**	12
P.C.A	12.08	12.50	1.00		
P.T.	11.54	12.00	0.52		
H.S.	10.38	10.00	0.51	**9-11**	13
P.C.A	9.54	10.00	0.52		
P.T.	6.50	6.50	0.53		
H.S.	6.10	6.00	0.74	**14-16**	10
P.C.A	6.80	7.00	0.79		
P.T.	4.00	4.00	0.94		
H.S.	2.50	2.50	0.53	**19-21**	10
P.C.A	2.90	3.00	0.88		

P.T: Probabilistic Template classifier from [42]

H.S: Histogram symmetry classifier from [13]

P.C.A: Principal component analysis classifier from [41]

S_{wii}: Speed of system on a **Wii**.

Chapter 5

Binary Matrix Pedestrian Classifier

This chapter describes and evaluates the performance of first of two novel algorithms described in this document. A pedestrian aspect ratio constrained binary matrix template to classify pedestrians, as identified by the fourth research question
Is a simplified matrix pedestrian aspect ratio classifier effective and accurate at classifying pedestrians in infrared image sequences?

An aspect ratio constrained classifier was implemented due to the following reasons:

- Model based classifiers and trackers, which have been effective with visual image sequences, cannot be used with infrared images (section 2.3).
- Pedestrian symmetry and appearance based classifiers have been demonstrated as being effective at classifying pedestrians in infrared image sequences (section 2.3 and 4.4.2).
- The pedestrian symmetry classifier based tracking system evaluated in the previous chapter, was shown to require fewer compute resources to classify and track pedestrians (section 4.5.3).

This chapter is organised as follows; first, the aspect ratio constrained classifier is described. After this, the performance of a pedestrian tracking system using this module is compared to the benchmark results (accuracy and speed) from the previous chapter. Finally, the effectiveness of the novel classifier is reviewed and any failure modes, if any, are identified.

5.1 Aspect Ratio and Appearance Based Pedestrian Classifier

The binary classifier works on foreground pixels from a segmented image. The algorithm for the classifier is as follows:

1. In the current frame set the foreground pixels to 1 and the background pixels to 0.

2. Scan the current frame from the top right to bottom left.

3. In the current row record the first 0 to 1 (background to foreground) pixel transition and the first 1 to 0 (foreground to background) transition.

4. If the pixel count is insufficient to generate large enough foreground region, look for the next 1 to 0 transition (foreground to background).

5. Generate a binary matrix template using the pixel count, constrained to the pedestrian aspect ratio and generate a template count value.

6. Xor the template with the the foreground region, used to generate the template and generate a count for the resulting matrix.

7. The count of the resulting matrix is used to measure the similarity between the template and foreground pixels. If this value is between 10% and 20% of the count for the template, then the bounding box contains a pedestrian.

8. If the count for the resulting matrix is less than 10% of the template, the template is too small. The pixel count is resized to the next foreground to background transition, the template rescaled (i.e goto step 2).

9. If the count is more than 20%, then the template is too large and a partial match is attempted.

10. If the partial match fails, then the pixels are assumed to be misclassified background and are ignored. The classifier moves onto the next background to foreground transition.

The description of the classifier is divided into the following sections, template generation and target matching.

5.1.1 Aspect Ratio Constrained Template Generation

Scanning the binary segmented image from top left to bottom right, the template is generated as follows:

The width of the template is constrained to:

$$m = 2I, \text{ where, } I : I \varepsilon N; I \geq 2 \tag{5.1}$$

I is the count of the foreground pixels between two background pixels in the current row. The and template is generated as a matrix of the form

$$T_{[n,m]} \text{ where, } n = 2m \tag{5.2}$$

$n = 2m$, constrains the generated template matrix to the pedestrian aspect ratio (height is twice the width).

After this is done all the elements of the template matrix are set to one, except for the elements at the following locations:

$$T_{[1,1]} \dots T_{[I,\frac{I}{2}]} \text{ and } T_{[1,(m-\frac{I}{2})]} \dots T_{[I,(m-\frac{I}{2})]} \tag{5.3}$$

I is the count of continuous foreground pixels in the current row of the image. This template is given a count value is equal to the number of elements in the matrix that are **not zero**: T_v

5.1.1.1 Template Generation Example

To illustrate how the matrix is generated, assuming that $I = 2$. From equations 5.1 and 5.2 we can calculate the size of the template,

$m = 2I; m = 4$ and $n = 2m; n = 8;$

$T_{[m,n]} = T_{[8,4]}$

Using equation 5.3, the elements at the following locations are set at zero,

$T_{[1,1]}, T_{[2,1]}, T_{[4,1]}, T_{[4,2]}$

Figure 5.1 is the aspect ratio constrained template generated when $I = 2$.

0	1	1	0
0	1	1	0
1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

I row's

Figure 5.1: Binary Template Example

5.1.2 Foreground Selection and Classification

For a frame F, if the background to foreground transition is found at pixel x. Then to match a template generated using a subsequent foreground to background transition at pixel $x + a$.

Find I, $I = (x + a) - x$, using the value of I calculate m and n (equations 5.1 and 5.2).

If, $n_{min} \geq n$, then identify the next foreground to background transition along the current row. Keeping x as the start of the foreground region but $x+a$ is moved to the right along row.

If on the other hand, $n_{max} \leq n$, then the foreground there is not a pedestrian, so x is moved to the next background to foreground transition. The foreground pixels $x, x+a$ are suppressed into background.

If, $n_{min} \geq n \leq n_{max}$, the foreground region can be localised and a bounding box fitted.

When the first foreground pixel is at x with the last foreground pixel at $x+a$; then $I = a$, $m = 2I$ and $n = 2m$ (from equations 5.2 and 5.1).

The corners of the the foreground bounding box will be at the following pixels,

$x - \frac{I}{2}$, the top left pixel of bounding box

$x + a + \frac{I}{2}$, the top left pixel of bounding box

$x - \frac{I}{2}, n$, the top left pixel of bounding box

$x + a + \frac{I}{2}, n$, the top left pixel of bounding box

Once the bounding box is created a template of the same size is generated $T_{[n,m]}$ as outlined in section 5.1.1.

5.1.2.1 Foreground Classification - Full Template

To classify the contents of bounding box, the bounding box is modelled as a matrix, $F_{[n,m]}$. This matrix is generated by setting all the foreground pixel in the bounding box to 1 and the background pixels to 0. The foreground matrix is Xor'ed with the template matrix.

$$R_{[n,m]} = F_{[n,m]} \oplus T_{[n,m]} \tag{5.4}$$

$$F_{[n,m]} \equiv T_{[n,m]} \begin{cases} 1, \text{ if, } T_{min} \geq \frac{R_v}{T_v} \leq T_{max} \\ 0, \text{ otherwise} \end{cases} \tag{5.5}$$

where, T_{max} and T_{min} are empirically set thresholds and R_v is the count value for the result matrix and T_v the count value for the template.

5.1.2.2 Foreground Classification - Partial Match

If the foreground matrix $F_{[n,m]}$ is found not to match the template matrix $T_{[n,m]}$ then a partial match is attempted. For this, only I rows from the foreground are matched to I rows of the template. This is done using the same foreground $F_{[n,m]}$ and template $T_{[n,m]}$ except only I rows are matched. The count threshold T_v is modified to count only I rows.

$$R_{[I,m]} = F_{[I,m]} \oplus T_{[I,m]} \tag{5.6}$$

$$F_{[I,m]} \equiv T_{[I,m]} \begin{cases} 1, \text{ if, } T_{min} \geq \frac{R'_v}{T_v} \leq T_{max} \\ 0, \text{ otherwise} \end{cases} \tag{5.7}$$

5.1.2.3 Foreground Classification - Match Failure

If both the full match and partial match fail, then, the foreground region $F_{m,n}$ is resized. The start of the foreground, x is kept at the same location, but, $x+a$ is moved along the current row to the next transition from foreground to background (pixel value 1 to 0) i.e. $x+a'$.

Using the new value of a'; n' is calculated, If, $n' < n_{max}$ then the template match is re-attempted for the foreground $x, x+a'$ (section 5.1.2 and 5.1.2.1).

If no transition to background is found before the end of the row, then the foreground bounding box is set at the end of the row and a template match attempted for that foreground object.

If $n' > n_{max}$ then, foreground region top left corner is moved from x to the next background to foreground transition x'. The foreground pixels $x, x+a$ are considered misclassified background or a non-pedestrian target and suppressed.

5.1.3 Classification Parameters and Description Summary

The classifier uses three classification parameters that affect how the tracking matrix will perform. The reason for using these parameters are:

n_{min} : This threshold is required to ensure that any objects that are too small to be classified are not processed. For use with the dataset from the previous chapter, this was set to 40 pixels.

T_{min} and T_{max}: A pedestrian target is assumed to occupy between 70% to 85% of a bounding box, two threshold values are needed to ensure that eroded pedestrian targets do not get misclassified as two or more targets.

n_{max} : This is required to ensure that non pedestrian targets do not get classified as one pedestrian target. For use with the dataset from the previous chapter, this was set at 160 pixels.

The sample template in figure 5.1, the binary template is a pedestrian aspect ratio constrained matrix. The algorithm described also uses elements of appearance and pedestrian symmetry based classification. Appearance based classification occurs in when the result matrix R is validated for T_{min}] and T_{max}. The pedestrian foreground region is expected to 80 to 90% of the the template due to two reasons:

- The square bounding box will contain some background pixels in the head and neck region of the pedestrian in the bounding box.
- Due to the low contrast between the foreground and background in infrared images (section 2.1), the edges of the foreground objects will be eroded.
- Pedestrian symmetry is used to generate the non-zero elements in the classifier.

5.2 Binary Template Classifier Performance Metrics

The previous section described a novel pedestrian classifier, the effectiveness of which must now be evaluated. The performance review in the previous chapter, collected the metrics for both accuracy and speed for three tracking systems using different pedestrian classifiers. These data will be used as a benchmark for a pedestrian tracking system using the novel classifier.

Of the three tracking systems implemented, the results for the tracker using the probabilistic template classifier will be omitted from the benchmark as it had the lowest accuracy of the three systems. The tracker using the principal component analysis classifier had the highest accuracy (section 4.4.5); the tracker using the histogram symmetry classifier was the fastest (section 4.5.5). The metrics will be presented as follows, first only the metrics for the tracking system using the novel classifier (both accuracy and speed) will be presented.

5.2.1 Binary Template Classifier - Accuracy

As with the benchmark data for the tracking systems in the previous chapter (section 4.4) the data for the tracker using the novel classifier will be presented as a distribution for a set of sequences with different pedestrian densities. As these metrics are for the classifier, its performance will be most obvious in the FN_c rates and the frame rates (speed) for the system. The overall T_{dr} rate for the system is not expected to improve as the accuracy of the bounding box tracker degrades at higher pedestrian densities (figure 4.3(b)).

The accuracy metrics (TP_c) for the binary template classifier are in table 5.1. As with the metrics in table 4.10, $\bar{TP_c}$ is the mean true positive rate for the N_s sequences with pedestrian density between T_{mn} and T_{mx} (from table 4.2). $\hat{TP_c}$ the median number true positive rate, σ_{TP_c} the standard deviation. Figure 5.2(a), sets the context for the TP_c rate, by comparing the true positive rates for all the classifiers.

We can use the graph to compare the performance of the binary template classifier to the other classifiers. From this we can infer the following:

Failure Modes: Testing with the design dataset (section 4.1.2) and gradual change in the TP_c rate for this classifier indicates that this pedestrian classifier has no distinct failure modes.

Accuracy: Of the three pedestrian classifiers used for benchmarking, the classifier using the PCA template model had the highest TP_c rate (section 4.4.3). The accuracy of binary template classifier matches the PCA classifier for sequences with pedestrian densities less than 15. After which it gradually degrades till it matches that of the histogram symmetry classifier.

Partial Pedestrian Match: The T_{dr} rates for the tracker using this classifier (figure 5.2(b)),

indicates that the partial match described above (equation 5.7) improves the overall track accuracy for the system. As the pedestrian density crosses 15 per frame, the classifiers ability to localise pedestrians decreases, this has a proportional drop in both the TP_c and T_{dr} rates.

In image sequences with pedestrian densities above 15 in a frame, both the pedestrian symmetry and appearance based classifiers have comparable changes in TP_c rates. This could be indicative of pedestrian density limitations for pedestrian appearance based classifiers, but a sample size of two is insufficient to confirm this observation. Data from more pedestrian appearance based classifiers with infrared image sequences will need to be collected confirm this.

Table 5.1: Binary Template Classifier - Accuracy (from table A.1)

$\bar{TP_c}$	$\hat{TP_c}$	σ_{TP_c}	T_{mn} and T_{mx}	N_s
1.00	1.00	0.00	**4-6**	12
0.90	0.90	0.06	**9-11**	13
0.78	0.73	0.10	**14-16**	10
0.74	0.70	0.11	**19-21**	10

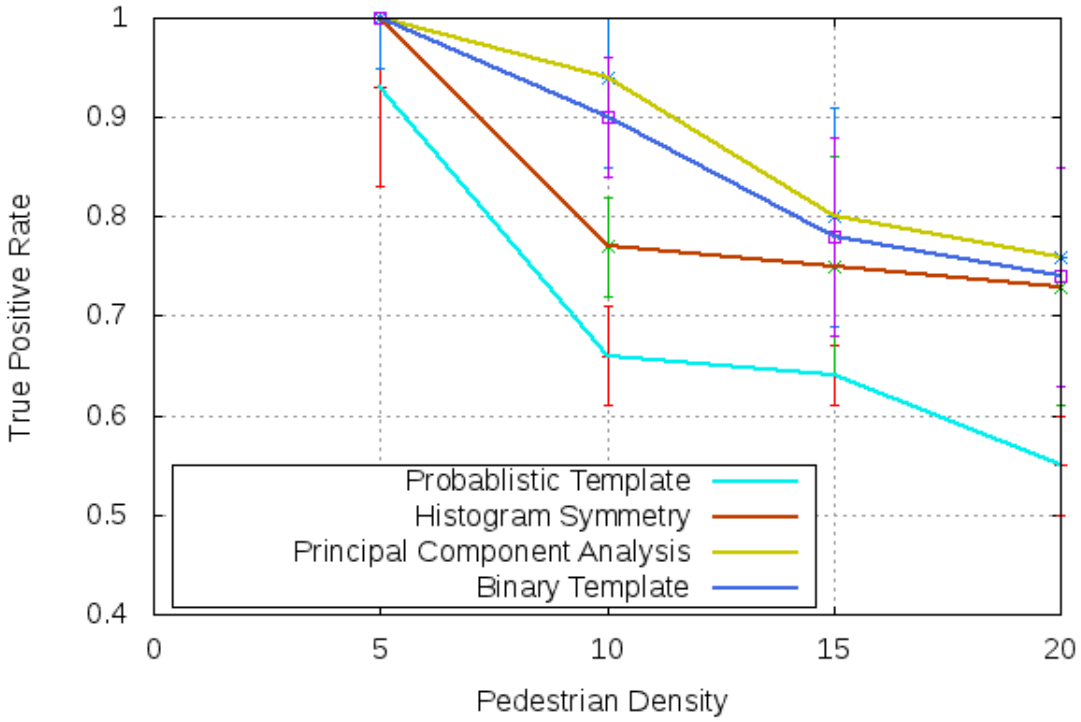
Table 5.2: Binary Template Classifier performance - T_{dr} (from table A.2)

$\bar{T_{dr}}$	$\hat{T_{dr}}$	$\sigma_{T_{dr}}$	T_{mn} and T_{mx}	N_s
0.97	0.97	0.01	**4-6**	12
0.92	0.92	0.01	**9-11**	13
0.82	0.82	0.01	**14-16**	10
0.73	0.74	0.01	**19-21**	10

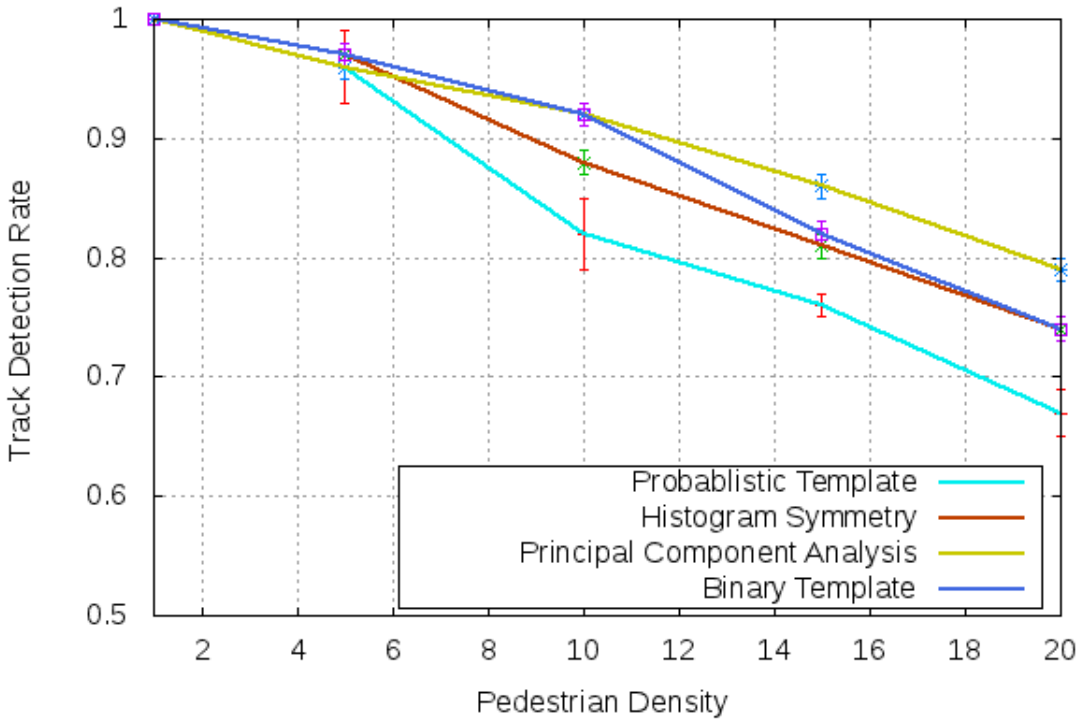
5.2.2 Binary Template Classifier - Speed

The frame rates achieved by the tracking system using the novel binary template classifier are in table 5.3. Figure 5.3(a) plots the frame rates for this the tracking system using this classifier and that of the fastest tracking system from the previous chapter (the tracker using the histogram symmetry classifier, section 4.5.3).

From the frame rate graph in figure 5.3(a) we can observe the following, the overall frame rate for the tracker using the binary template classifier is comparable to the fastest tracker identified in the previous chapter (section 4.5.3). Initially the tracking system using the histogram symmetry classifier is faster, but as the pedestrian density increases the change in speed for the tracker using the binary template classifier is seen to outperform the tracker using the histogram symmetry classifier.



(a) The True Positive Rate TP_c Graph



(b) The Track Detection Rate T_{dr} Graph

Figure 5.2: Graphs from data in tables 5.1, 4.10, 5.2 and 4.6

Table 5.3: System performance - Speed on All Three Platforms (from tables A.4, A.6 and A.8)

$\bar{S_{PC}}$	$\hat{S_{PC}}$	$\sigma_{S_{PC}}$	T_{mn} and T_{mx}	N_s
19.08	19.00	0.90	**4-6**	12
17.00	17.00	0.71	**10-11**	13
14.10	14.50	0.99	**14-16**	10
12.90	13.00	0.74	**19-21**	10
$\bar{S_{PS3}}$	$\hat{S_{PS3}}$	$\sigma_{S_{PS3}}$	T_{mn} and T_{mx}	N_s
19.00	19.00	0.74	**4-6**	12
15.54	15.00	0.88	**10-11**	13
12.20	12.00	0.42	**14-16**	10
10.40	10.00	0.52	**19-21**	10
$\bar{S_{wii}}$	$\hat{S_{wii}}$	$\sigma_{S_{wii}}$	T_{mn} and T_{mx}	N_s
12.67	12.50	0.78	**4-6**	12
10.23	10.00	1.09	**10-11**	13
6.90	7.00	0.74	**14-16**	10
3.30	3.50	0.82	**19-21**	10

This is true for all three hardware platforms and in case of the Wii, with image sequences at high pedestrian densities ($\approx$ 15 pedestrians per frame) the tracker using the binary template tracker is the only one able to sustain frame rates more than $5fps$ for all sequences.

5.3 Summary - Binary Template Classifier

This chapter described a novel aspect ratio constrained pedestrian classifier for use in infrared image sequences. The classification algorithm was described and this was followed by the implementation details. This was followed by the speed and accuracy metrics for the classifier.

Where we observed the following, the accuracy of the binary template classifier is as accurate as the best from literature for pedestrian sequences where the pedestrian density is less than $\approx$ 15 per frame(figure 5.2(a)). The accuracy for the novel binary template at degrades for pedestrian densities above 15 per frame (figure 5.2(a)), as the same degradation is observed in the TP_c curve for the histogram symmetry classifier, which also uses pedestrian symmetry based classification there might be a potential limitation for these types of classifiers.

Finally, the binary template classifier is demonstrated as requiring the fewest resources. The evidence for this is in figure 5.3(a) and table 5.3, where the tracker using this classifier has the highest frame rates on all three hardware platforms.

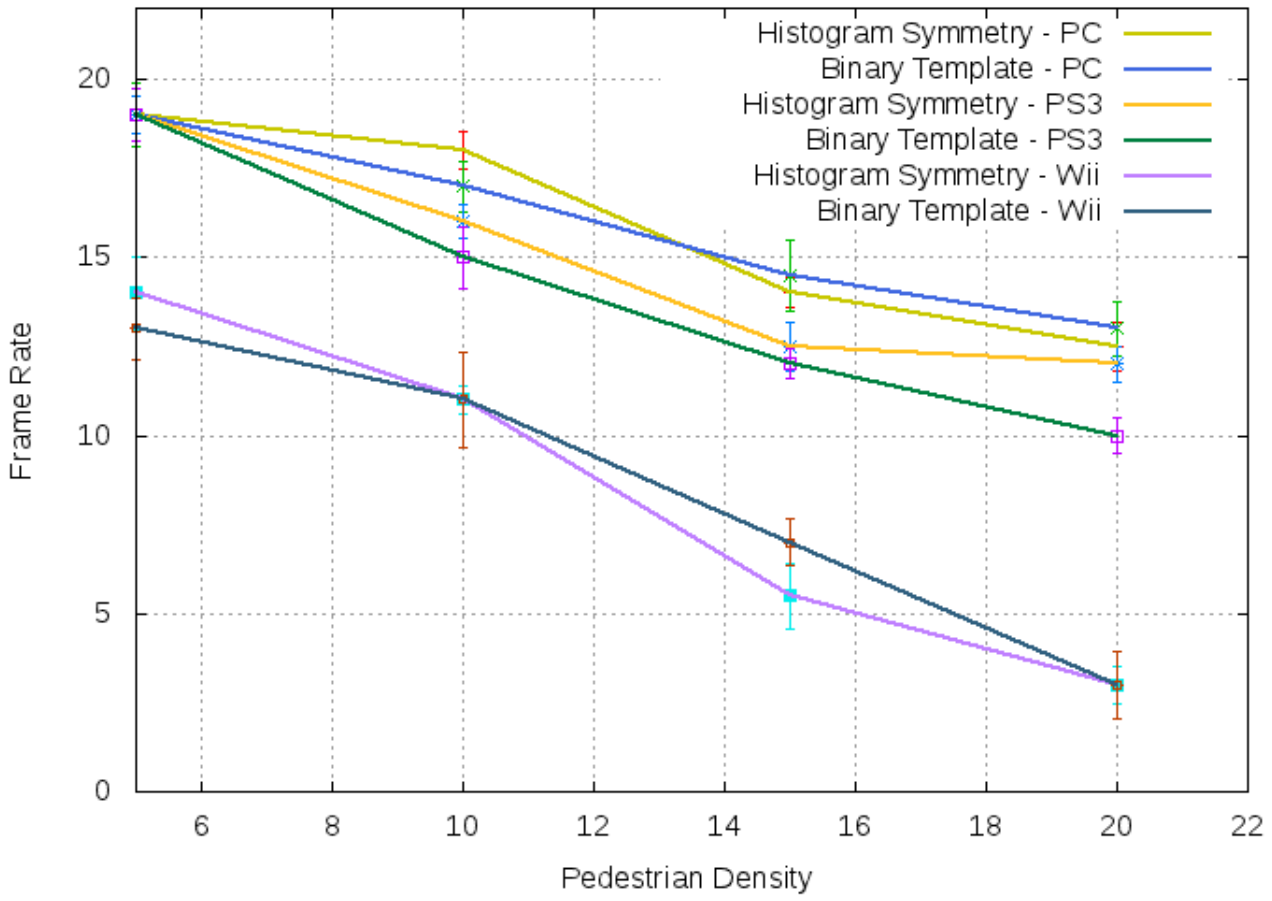
5.3.1 Contributions to Body of Knowledge and Novelty

Now that the binary template for pedestrian classification has been described, the contributions to the body of knowledge in this chapter can be identified. Using these, the novelty and further avenues of research will be highlighted. The third research question required the validation of the used of an aspect ratio constrained binary matrix for pedestrian classification.

Is a simplified matrix pedestrian aspect ratio classifier effective and accurate at classifying pedestrians in infrared image sequences?

The answer to the above question is yes, a simplified matrix classifier is effective at classifying pedestrians in infrared image sequences. This was demonstrated in by describing a novel pedestrian classifier, whose performance was analysed in section 5.2. Where this classifier was shown to have the highest accuracy while matching the throughput of the fastest classifier in literature.

While measuring the performance of the novel classifier, a potential accuracy limitation in terms of pedestrian density for symmetry and aspect ratio based classifiers were identified. As the sample size of two appearance and symmetry based classifiers is insufficient and further testing of symmetry based pedestrian classifiers is required.



(a)

Figure 5.3: Speed for Tracking Systems, Histogram Symmetry and Binary Template

Chapter 6

Intensity Codebook Pedestrian Tracker

This chapter describes and evaluates the performance of the second of the two novel algorithms introduced in this document, an intensity codebook to track pedestrians in infrared image sequences. A bounding box tracking approach was selected as a promising avenue for a tracker in infrared images as:

- Infrared images do not contain sufficient texture information to be effective when used with model based trackers.
- A simple intensity based bounding box tracker described in section 4.2.5, was able to effectively track pedestrians in infrared image sequences.

The design of the novel tracker is predicated on the following assumptions, improving the bounding box intensity model will improve the trackers ability to discriminate between pedestrian targets during occlusion events involving 3 or more pedestrians. Secondly, using the intensity codebook to model bounding box pixel history is not expected to increase the compute requirements of the pedestrian tracking system excessively.

This chapter is organised as follows; first, the codebook based tracker is described and the performance of the pedestrian tracking system using this module is compared to the benchmark results from the chapter 4. Finally, the effectiveness of the novel tracker is reviewed and any failure modes are identified.

6.1 Codebook Based Bounding Box Tracker

The bounding box tracking algorithm described in section 4.2.5, for track recovery under occlusion, compared known bounding box intensity values to current bounding box intensity values to identify the best match (equation 4.11). This was demonstrated as being effective at tracking pedestrians through occlusion events (section 4.4).

But for sequences with pedestrian densities above 10, the above occlusion recovery mechanism does not work effectively (section 4.4.5). As the properties of infrared images

precludes the use of geometric models [7], a different mechanism to model the appearance of a bounding box is necessary.

As the codebook mechanism has been shown to effectively model the colour and intensity histories of pixels independently [39], a codebook bounding box model was developed and implemented. This section will describe the algorithm and the implementation details of the same.

6.1.1 Codebook Tracker - Algorithm and Implementation Details

The codebook mechanism is used to model and track pedestrians in foreground regions are follows:

1. In the initial frame, generate the intensity codebooks for foreground pixels in classified bounding boxes.
2. For subsequent frames, first attempt a foreground Xor match for overlapping bounding boxes.
3. For multiple overlapping bounding boxes use the intensity codebook to identify the most effective match.
4. Update the codebooks for all bounding boxes, both matched by the Xor match and those matched by the codebook match
5. Compact the codebooks for all the bounding boxes

6.1.1.1 Codeword Creation, Matching and Update

The intensity history for a pixel is stored in a set of j codewords, each of which stores the following details:

$$CW_{[m,n],j} = \{\hat{i}_j, \check{i}_j, f_j, \lambda_j, [m,n]_j\} \tag{6.1}$$

$$\tag{6.2}$$

Where,

$j : j \in N$

$\hat{i}_j$ is the minimum intensity for the codeword,

$\check{i}_j$ is the maximum intensity for the codeword,

f_j the number of frames the pixel has not been part of the foreground,

λ_j is the Maximum Negative Run Length (MNRL) for the codeword,

$[m,n]_j$ is the location of the codeword within the bounding box,

Codeword Creation:

To create a codeword for a pixel $k_{[m,n]}(i)$,

$$\hat{i} = i, \check{i} = i, f = 0, \lambda = 0, [m,n] = [m,n]. \tag{6.3}$$

Codeword Intensity Match:

The current pixel intensity $k(i)$ matches the j^{th} codeword if,

$$\hat{i} \leq k(i) \leq \check{i} \tag{6.4}$$

$$k(i) > \check{i} \text{ and } (k(i) - \hat{i}) < T_1 \tag{6.5}$$

$$k(i) < \hat{i} \text{ and } (\check{i} - k(i)) < T_1 \tag{6.6}$$

Codeword Intensity Update:

The current pixel intensity $k(i)$ is used to update the j^{th} codeword,

$$\hat{i}_j \leq k(i) \leq \check{i}_j, \text{ then },\hat{i}_j = \hat{i}_j \text{ and } \check{i}_j = \check{i}_j \tag{6.7}$$

$$k(i) > \check{i}_j \text{ and } (k(i) - \hat{i}_j) < T_1, \text{ then },\check{i}_j = k(i) \tag{6.8}$$

$$k(i) < \hat{i}_j \text{ and } (\check{i}_j - k(i)) < T_1, \text{ then },\hat{i}_j = k(i) \tag{6.9}$$

$$f_j = 0 \tag{6.10}$$

The T_1 value is the intensity threshold, and was empirically determined and set at 10.

6.1.1.2 Bounding Box - Codebook Tracking

For two overlapping bounding boxes, k and l in two consecutive frames $X_{x,y}$ and $X+1_{x,y}$. The bounding boxes are rescaled so that the larger of the two (k or l) is the same size as the smaller bounding box and the Xor bounding box match is attempted (previously described in section 4.2.5). Assuming $bm1_{a,b}$ and $bm2_{a,b}$ are the resized foreground binary matrices for k and l, adapting equation 4.9,

$$E'_{a,b} = bm1_{a,b} \oplus bm2_{a,b} \tag{6.11}$$

$$k \equiv l \begin{cases} \text{if, } \left(\frac{\sum E'_{a,b}}{a \times b}\right) < T_2 \\ 0, \text{ otherwise} \end{cases} \tag{6.12}$$

If the Xor match fails; or, for a bounding box k in $X+1$ there are multiple overlapping bounding boxes $l_1 \ldots l_w$ from the previous frame X. A codebook match is used:

$$A_w+ = \begin{cases} 0, \text{If, } bm1_{a,b} : bm2_{a,b} \in CB_{a,b}) \\ 1, \text{ otherwise} \end{cases} \tag{6.13}$$

$$\text{again, } k \equiv l_w, \text{ if, } \frac{A_w}{a \times b} < T_2 \tag{6.14}$$

The the bounding box pair with the lowest A_w are considered as the best matches. A T_2 value of .20 was empirically found to be optimum.

6.1.1.3 Bounding Box - Codebook Update

To update the codebook for the bounding boxes, the pixel intensity information for $k_{[m,n]}(i)$ needs to be added to the codebook models. The procedure for this is described below.

For bounding boxes from X with no matching bounding boxes in $X+1$:

Check and mark lost any bounding boxes, where a minimum of 70% of the codewords have an $MNRL > MN_T$.

For all other unmatched bounding boxes, increment the $MNRL$ and f values for all codewords by 1.

Discard all existing codewords where $f_j > T_3$.

For the matched bounding boxes :

Set the MNRL, $\lambda_{[m,n],j}$ for all the codewords in that bounding box to 0.

Update the codewords for the bounding boxes with $k_{[m,n]}(i)$ using equations 6.7 to 6.10.

If the pixel intensity $k_{[m,n]}(i)$ does not match any of the j codewords, add a new codeword using equation 6.3.

If no codewords are found for a foreground pixel at location m,n then add a new codeword for $k_{[m,n]}(i)$ using equation 6.3.

If $k_{[m,n]}(i)$ does not match any existing codeword, add a new codeword using equation 6.3.

Discard all existing codewords where $f_j > T_3$.

For new bounding boxes :

Generate the codebook for that bounding box, by creating new codewords for all foreground pixels in the bounding box.

6.1.2 Codebook Tracker - Parameters and Summary

This section described in detail the implementation of a bounding box intensity codebook tracker. The bounding box tracker as described above implements the track all targets solution to occlusion recovery (refer to section 2.4.1.1 for other solutions to occlusion recovery). The bounding box model is dynamically updated while preserving the most frequent intensity profile for the object within the bounding box. The model update rates are controlled by T_1 and T_3.

T_1 is set according to the intensity profiles of the pedestrians in the foreground. If all possible pedestrian targets occupy a narrow intensity range (all warm or all cold pedestrians), then the value of T_3 needs to be smaller. As the pedestrians in the dataset occupy a wide temperature profile (both cold and warm pedestrian targets), after testing with the design dataset (section 4.1.2) 10 was found to be most effective threshold value.

T_3 on the other hand controls how quickly the intensity model for a pedestrian in a bounding box is updated. If the pedestrian orientation in relation to the camera is not expected to change rapidly (orientation change from head-on to sideways in less than 40 frames), the update process can be slow. After testing with the design dataset (section 4.1.2), for this evaluation, T_3 was set at 70.

MN_T controls the removal of bounding boxes from being actively tracked. MN_T is the number of frames that less than 30% of a previously tracked bounding box has been positively identified in the foreground. This is an empirically determined parameter and for the image sequences in this dataset, 150 (5 frames at $30fps$) was optimum.

The 30% *MNRL* threshold is used as it was found to be large enough to avoid problems with *'track jumping'*. Track jumping occurs when parts of two pedestrians targets are similar enough for the tracker to cross assign the bounding boxes. As most pedestrians in infrared image dataset occupy the same temperature range, reducing the probability of this occurring increases the accuracy of the tracking system.

Compared to the codewords used for background modelling in visual images (equation 2.12), the codewords used to model the pedestrian bounding box contain much less information (equation 6.1). The accuracy will be improved without too much of a change in the frame rates. This will be demonstrated in the next section.

6.2 Intensity Codebook Tracker - Performance

The final research question in the introduction was,

> How effective is a codebook tracker at tracking pedestrian targets in infrared image sequences?

So far, this chapter has outlined the need for an improved bounding box tracker and described an intensity codebook tracker for use with infrared image sequences. Using the benchmark data from chapter 4, we can evaluate the performance of an intensity codebook tracker.

In the previous chapters, this document analysed pedestrian tracking systems using four different pedestrian classifiers. For evaluating the performance of the codebook tracker, only two of the four classifiers will be used with the intensity codebook tracker. The two classifiers that will not be used are the histogram symmetry classifier and the probablistic template classifier, as they had the lowest accuracy of the four pedestrian classifiers evaluated so far.

Table 6.1: Systems as Implemented

	Segmentation	Detection	Tracking
Binary Template (B.T)	Gaussian Background Modelling	Binary Template	Intensity Codebook Tracker
Principle Component Analysis (PCA) [41]	Gaussian Background Modelling	Principle Component Analysis (PCA)	Intensity Codebook Tracker

Additionally, the two classifiers that will be used have been identified as having the best accuracy of all four classifiers evaluated so far. These two classifiers also represent the extremes of the resource requirement curve. The PCA classifier tracking system requiring the most resources and the BT classifier requiring the least amount of resources. The accuracy metrics were recorded in terms of the T_{dr} rates for two tracking systems, one with a PCA classifier and the second with a BT classifier. Any accuracy gains by the intensity codebook tracker over the bounding box tracker will be most visible for pedestrian sequences with pedestrian densities greater than ≈ 15.

As the intensity codebook model dynamically models every bounding box, this potentially may affect the frame-rates at which the sequences are processed on the three hardware platforms. The change in frame rates between the bounding box tracker and the codebook tracker on a hardware platform will indicate a change in resource utilisation.

6.2.1 Intensity Codebook Tracker - Accuracy

To collect the accuracy metrics for the codebook tracker, tracking systems as outlined in table 6.1 were implemented on all three hardware platforms. No change in accuracy was recorded between hardware platforms, the T_{dr} rates for the image sequences from the dataset with the two classifiers is recorded in table 6.2.

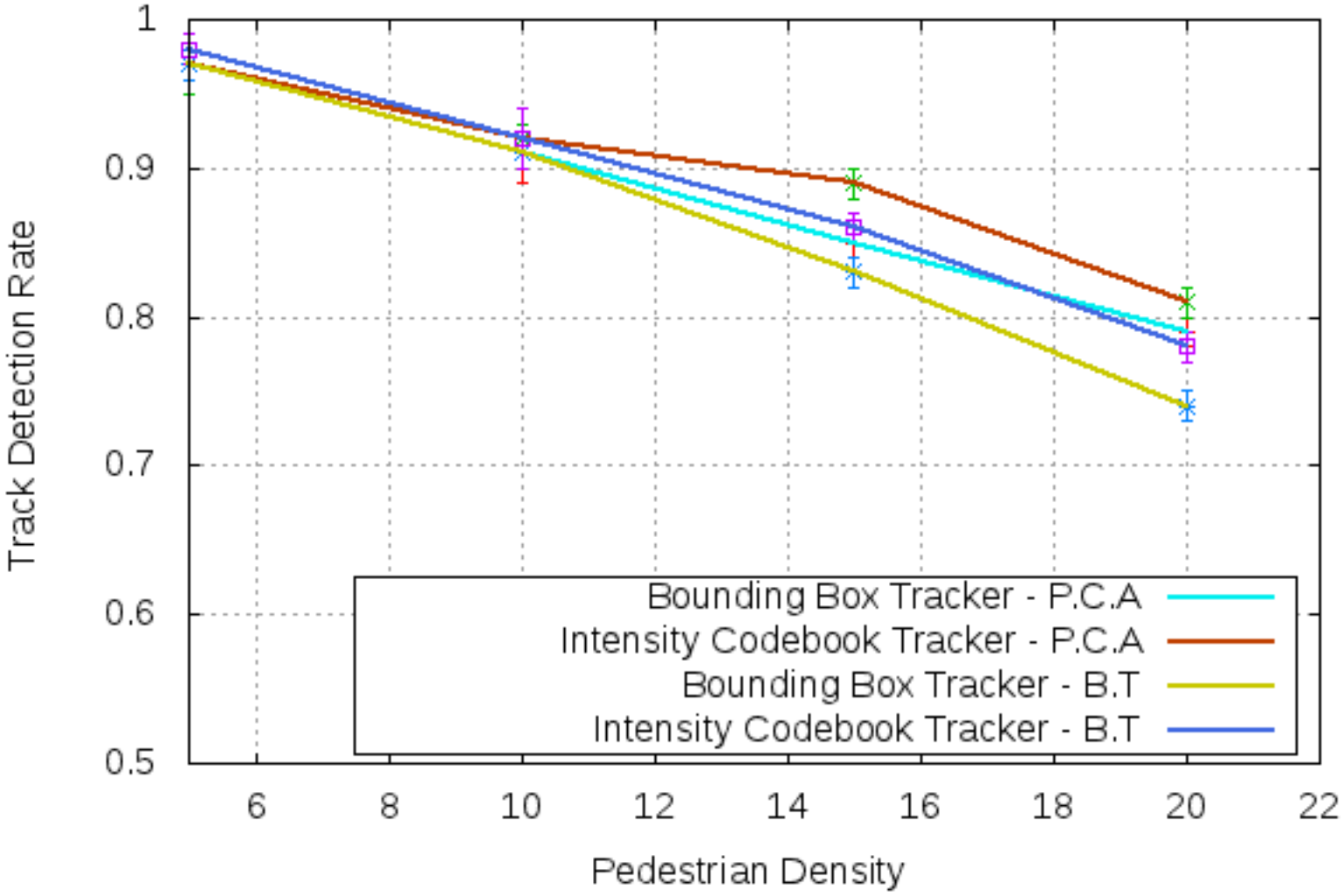
The graph comparing the T_{dr} rates of the intensity codebook tracker and the bounding box tracker is in figure 6.1(a). From the comparative graph, we see that the systems using the codebook tracker outperform the tracking systems using the bounding box tracker at all pedestrian densities.

Table 6.2: Intensity Codebook Tracker Accuracy, T_{dr} (from table A.3)

Classifier	$\bar{T_{dr}}$	$\hat{T_{dr}}$	$\sigma_{T_{dr}}$	T_{mn} and T_{mx}	N_s
PCA	0.97	0.97	0.02	**4-6**	12
	0.92	0.92	0.01	**9-11**	13
	0.89	0.89	0.01	**14-16**	10
	0.81	0.81	0.01	**19-21**	10
BT	0.98	0.98	0.01	**4-6**	12
	0.92	0.93	0.02	**9-11**	13
	0.86	0.87	0.01	**14-16**	10
	0.78	0.79	0.01	**19-21**	10

P.C.A: Principle component analysis classifier from [41]

B.T: Binary template classifier from 5



(a)

Figure 6.1: Comparison Between T_{dr} Rates for the Trackers

6.2.2 Intensity Codebook Tracker - Speed

The speed at which the intensity codebook tracking systems processed the sequences is recorded in table 6.3. As with the speed metrics in the previous chapters, the results for the intensity codebook tracker are recorded here as frame rate distribution. Figure 6.2 compares the frame rates for the novel codebook tracker to that of the bounding box tracker from literature. As no significant changes in speed were identified on any of the hardware platforms, the increased tracking accuracy did not require more system resources.

Figure 6.2(a), compares the frame rates on all three hardware platforms for tracking systems using the bounding box tracker to those using the intensity codebook tracker; both with this principal components analysis classifier. Though the intensity codebook tracker is more accurate than the bounding box tracker (section 6.2.1), there is no change in the frame rates between the trackers. A similar result is seen when the frame rates for the tracking systems using the binary template classifier are examined (figure 6.2(b)).

As the frame rates for the pedestrian tracking system using the intensity codebook tracker does not change from the systems using the bounding box tracker, we can conclude the that processing and memory requirements of the intensity codebook mechanism do not excessively tax the resources on any of the hardware platforms.

6.3 Intensity Codebook Tracker - Summary

This chapter has described a novel intensity codebook mechanism for tracking pedestrians in infrared video sequences. The first section, introduced the background for the development of the tracker after which the implementation details of the tracker were described. After this, while analysing the accuracy metrics of the tracking system we found the following:

- The accuracy of the pedestrian tracking system at higher pedestrian densities is increased.
- The increased accuracy does not detrimentally affect the processing speeds.
- The low foreground visibility requirement, of 30% is sufficient to ensure that occluded targets are successfully tracked through occlusion events.
- Using a dynamic process to evaluate and discard bounding boxes ensures that viable targets are not removed from the tracker prematurely.

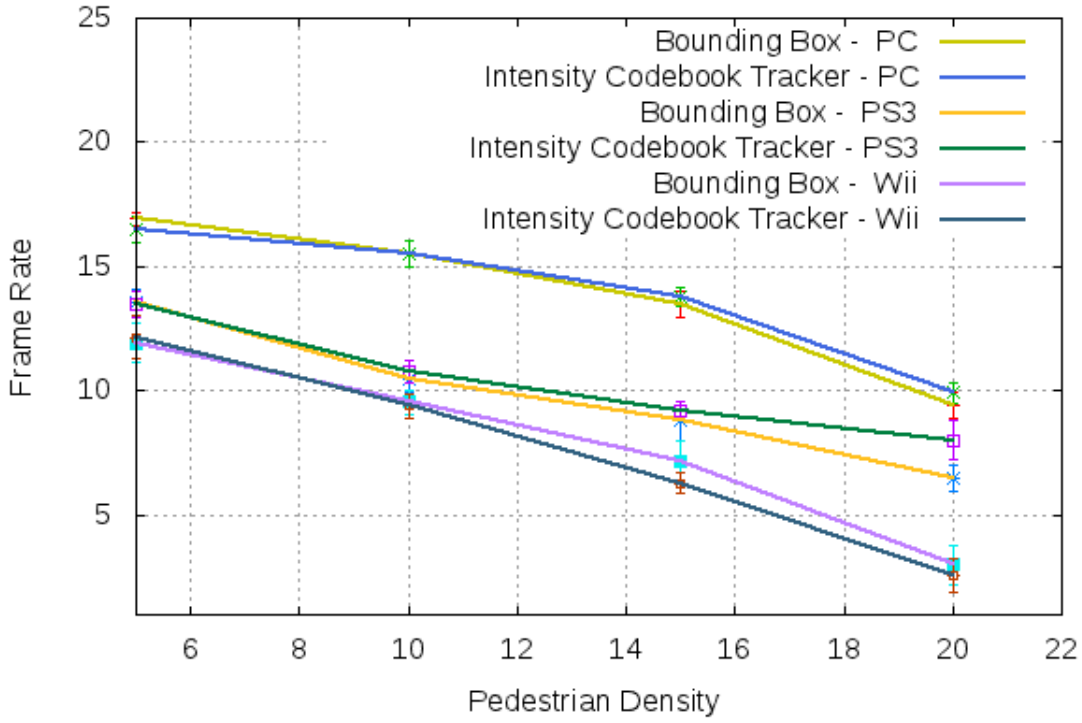
6.3.1 Contributions to Body of Knowledge and Novelty

This chapter described and implemented a novel intensity codebook based pedestrian tracking algorithm, providing sufficient evidence to be able to definitively answer the last

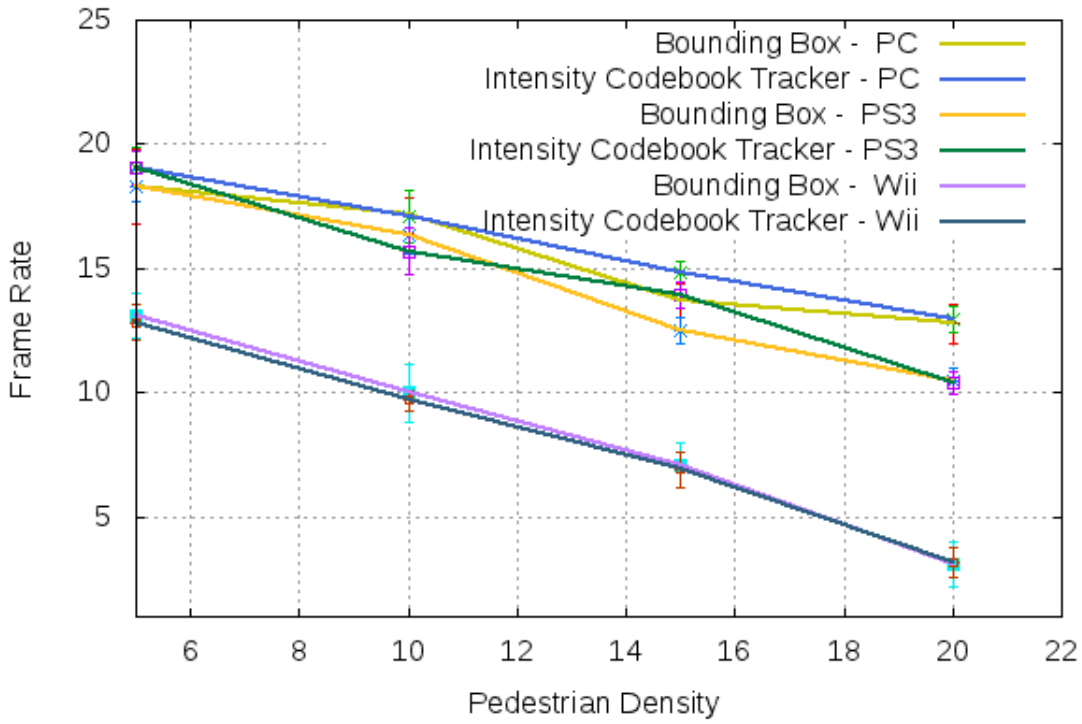
of the research questions from section 1.5,
How effective is a codebook tracker at tracking pedestrian targets in infrared image sequences?

In response to which the evidence shows that the novel codebook tracker is more accurate than the bounding box tracker from literature at higher pedestrian densities (table 6.2 and figure 6.1(a)). This accuracy does not come at the cost of increased compute demands, evidence for which is in figure 6.2.

As the codebook tracker uses pedestrian intensity texture information to discriminate between targets in occlusion events, the fact that the tracker demonstrates higher accuracy than the bounding box tracker is again evidence for the presence of surface texture in infrared images. The improved accuracy of the codebook intensity tracker also indicates that the use of geometric models for pedestrian classification in infrared images needs to be revisited. As the resolution and sensitivity of current infrared cameras might be recording sufficient texture information to enable the use of a rudimentary geometric model.



(a) Principle Component Analysis Classifier



(b) Binary Template Classifier

Figure 6.2: Frame Rates for the Tracker on Different Hardware Platforms

Table 6.3: Intensity Codebook Tracker Speed, Frame Rates (from tables A.5, A.7 and A.9)

$\bar{F}$	$\hat{F}$	σ_F	T_{mn} and T_{mx}	N_s
PCA classifier - PC				
16.50	16.50	0.52	**4-6**	12
15.38	15.00	0.51	**9-11**	13
13.51	13.50	0.43	**14-16**	10
10.37	10.30	0.46	**19-21**	10
PCA classifier - PS3				
13.50	13.50	0.52	**4-6**	12
10.69	11.00	0.48	**9-11**	13
9.48	9.65	0.41	**14-16**	10
8.36	8.50	0.69	**19-21**	10
PCA classifier - Wii				
12.25	13.00	0.97	**4-6**	12
9.33	9.40	0.42	**9-11**	13
6.05	5.85	0.56	**14-16**	10
3.35	3.50	0.57	**19-21**	10
BT classifier - PC				
18.67	19.00	1.07	**4-6**	12
17.08	17.00	0.86	**9-11**	13
15.15	15.20	0.25	**14-16**	10
13.36	13.45	0.53	**19-21**	10
BT classifier - PS3				
18.50	18.00	0.67	**4-6**	12
16.00	16.00	0.71	**9-11**	13
13.74	13.70	0.59	**14-16**	10
10.72	10.90	0.49	**19-21**	10
BT classifier - Wii				
12.67	12.00	0.89	**4-6**	12
10.25	10.30	0.51	**9-11**	13
7.32	7.35	0.93	**14-16**	10
3.39	3.25	0.61	**19-21**	10

P.C.A: Tracking system using the principal component analysis classifier.

B.T: Tracking system using the binary template classifier.

Chapter 7

Summary and Further Work

The six research questions about pedestrian tracking in infrared image sequences from section 1.5 form the backbone of this thesis. The literature review identified that thermal infrared images had some features that simplified pedestrian detection and tracking in infrared image sequences. This review also identified that infrared images recorded insufficient surface detail for geometric models to be usable with infrared images. The features of infrared images relevant to pedestrian tracking can be summarised as follows:

Advantages

Pedestrians are usually distinct in infrared images, usually in occupying the brightest part of the intensity range.

All pedestrians in thermal infrared images occupy the same temperature range, making identification simpler.

Lack of surface texture in infrared images reduces the amount of foreground filtering that needs to be done to classify and localise pedestrians

Disadvantages :

Infrared images are noisy reducing fine temperature gradients.

The contrast between foreground and background regions is limited, making accurate background segmentation difficult.

In infrared images recorded in cold environments, pedestrians wear insulating outer layers of clothing, altering their appearance.

The chapter following the literature review, described the surface texture in infrared images. Here, we found that the temperature to grey-scale mapping in infrared images recorded using the FLIR thermal infrared camera is non-linear (section 3.2). One major implication of this is that pedestrian targets above room temperature will have more texture detail than pedestrians below room temperature. This same chapter also introduced

a novel method for contrast enhancement in infrared images with non-linear temperature to grey-scale maps.

As the level of contrast enhancement is subjective and these processed images were very noisy, further improvement of the algorithm is required. The next chapter (chapter 4 described an infrared image dataset, where the sequences are grouped by the pedestrian density in the sequence. Using this dataset a set of benchmark performance (accuracy and speed) results were obtained.

On analysing the results, two of the pedestrian classifiers from literature were found to have distinct failure modes which affected their ability to classify pedestrians (section 4.4). In the same chapter we identified the limitations of the bounding box tracker at higher pedestrian densities (section 4.4.4) and with long term occlusion.

Chapter 5 described and evaluated the performance of a novel aspect ratio constrained binary matrix for pedestrian classification. The first half of the chapter described and discussed the implementation details of the classifier (section 5.1. The speed and accuracy metrics for two tracking systems from literature, previously recorded in chapter 4 were used as benchmarks for the novel classifier.

On comparing the performance of the three classifiers, the novel classifier was found to match the accuracy of the most accurate classifier from literature (section 5.2.1) while it required the same amount of compute resources as the fastest of the classifiers from literature (section 5.2.2). Additionally, a pedestrian density was identified as a potential limitation for pedestrian symmetry based classifiers, as the sample size with this dataset consists of two classifiers there is insufficient evidence for this.

The next chapter (chapter 6) described and evaluated a novel intensity codebook for bounding box tracking. The first section of the chapter described and discussed the implementation details of the intensity codebook tracker. Where the parameters controlling the performance of the tracker were described. The performance evaluation found that the intensity codebook tracker does not have any distinct failure modes. The intensity codebook model is robust and able to match the bounding boxes during long term occlusion events, which improves its track detection rates while using the same resources as the bounding box tracker from literature.

7.1 Plan for Further Work

In the course of presenting the results of research into pedestrian tracking in infrared sequences the following areas of research were identified (from sections 3.3, 4.6, 5.3 and 6.3):

Infrared Image Texture: Chapter 3.1 described a novel contrast enhancement procedure for texture enhancement for use in infrared images. As currently described, these resulting images are noisy and the contrast ratios used are subjective. Identi-

fying a contrast ratio scale based on the median intensity in an image will automate the process of contrast enhancement and an effective noise filter needs to be identified (section 3.3).

Binary Template Classifier: The Binary Template classifier has been validated with infrared image sequences recorded indoors. While its ability to localise pedestrians in occlusion events has been validated. This has not been tested with image sequences recorded at greater heights where the pedestrian targets are smaller.

Symmetry Based Classifiers: The benchmark results with both the binary template classifier and the histogram symmetry classifier identified an accuracy limitation for this type of classifier at higher pedestrian densities. As the sample size consists of two classification algorithms more data is required before this is validated (section 5.3).

Model Based Classification: Chapter 3.1 established that infrared images contain surface texture details. The same chapter also proposed an algorithm to enhance the visibility of texture in infrared images. As the resolution and sensitivity of infrared cameras has improved, more studies of infrared images have been undertaken. Using this information about texture in an infrared image, the use of model based tracking and pedestrian classification in infrared images must be re-evaluated (section 6.3).

References

[1] FLIR, "Thermovision micron brochure," 2006.

[2] P. Kruse, E. Weber, R. Willardson, and D. Skatrud, *Uncooled Infrared Imaging Arrays and Systems,*, ser. SEMICONDUCTORS AND SEMIMETALS. ACADEMIC PRESS, 1997, vol. 47, ch. 2.

[3] ——, *Uncooled Infrared Imaging Arrays and Systems,*, ser. SEMICONDUCTORS AND SEMIMETALS. ACADEMIC PRESS, 1997, vol. 47, ch. 3, pp. 47–85.

[4] N. J. W. Morris, S. Avidan, W. Matusik, and H. Pfister, "Statistics of infrared images," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, june 2007, pp. 1–7.

[5] N. Nandhakumar, "Integrated analysis of thermal and visual images for scene interpretation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 469–481, 1988.

[6] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern Recognition*, vol. 36, no. 3, pp. 585–601, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/B6V14-4771RWD-2/2/41d0f27ca154e3140e27fdf68a9df519

[7] Y. Fang, K. Yamada, Y. Ninomiya, B. Horn, and I. Masaki, "Comparison between infrared-image-based and visible-image-based approaches for pedestrian detection," in *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, june 2003, pp. 505–510.

[8] H. Nanda and L. Davis, "Probabilistic template based pedestrian detection in infrared videos," in *Intelligent Vehicle Symposium, 2002. IEEE*, vol. 1, june 2002, pp. 15–20 vol.1.

[9] Y. Fang, K. Yamada, Y. Ninomiya, B. Horn, and I. Masaki, "A shape-independent method for pedestrian detection with far-infrared images," *Vehicular Technology, IEEE Transactions on*, vol. 53, no. 6, pp. 1679–1697, nov. 2004.

[10] A. Broggi, A. Fascioli, M. Carletti, T. Graf, and M. Meinecke, "A multi-resolution approach for infrared vision-based pedestrian detection," in *Intelligent Vehicles Symposium, 2004 IEEE*, june 2004, pp. 7–12.

[11] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, and M. Porta, "Artificial vision in road vehicles," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1258–1271, jul 2002.

[12] T. Chalidabhongse, K. Kim, D. Harwood, and L. Davis, "A perturbation method for evaluating background subtraction algorithms," in *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. Nice, France: IEEE, 2003.

[13] M. Bertozzi, A. Broggi, A. Fascioli, T. Graf, and M.-M. Meinecke, "Pedestrian detection for driver assistance using multiresolution infrared vision," *Vehicular Technology, IEEE Transactions on*, vol. 53, no. 6, pp. 1666–1678, nov. 2004.

[14] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. S. Davis, "Background modeling and subtraction by codebook construction," in *Image Processing, 2004. ICIP '04. 2004 International Conference on*, vol. 5, oct 2004, pp. 3061–3064 Vol. 5.

[15] A. Lipton, H. Fujiyoshi, and R. Patil, "Moving target classification and tracking from real-time video," in *Proceedings of The Fourth IEEE Workshop on Applications of Computer Vision, 1998. WACV '98.* Princeton, NJ: IEEE Computer Society, 1998, pp. 8–14.

[16] M. Piccardi, "Background subtraction techniques: A review," in *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, oct. 2004, pp. 3099–3104.

[17] F. Kristensen, P. Nilsson, and V. Owall, "Background segmentation beyond rgb," in *Computer Vision – ACCV 2006*, ser. Lecture Notes in Computer Science, P. Narayanan, S. Nayar, and H.-Y. Shum, Eds. Springer Berlin / Heidelberg, 2006, vol. 3852, pp. 602–612, 10.1007/11612704_60. [Online]. Available: http://dx.doi.org/10.1007/11612704_60

[18] O. Javed, K. Shafique, and M. Shah, "A hierarchical approach to robust background subtraction using color and gradient information," *Motion and Video Computing, IEEE Workshop on*, vol. 0, p. 22, 2002.

[19] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 2, 1999, pp. 2 vol. (xxiii+637+663).

[20] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Seventh International Conference on Computer Vision*, vol. 1, 1999, p. 255.

[21] S. ching S. Cheung and C. Kamath, "Robust techniques for background subtraction in urban traffic video," S. Panchanathan and B. Vasudev, Eds., vol. 5308. SPIE, 2004, pp. 881–892. [Online]. Available: http://link.aip.org/link/?PSI/5308/881/1

[22] A. Elgammal, R. Duraiswami, and L. S. Davis, "Non-parametric model for background subtraction," in *Computer Vision — ECCV 2000*, ser. Lecture Notes in Computer Science, D. Vernon, Ed. Springer Berlin / Heidelberg, 2000, vol. 1843, pp. 751–767, 10.1007/3-540-45053-X_48. [Online]. Available: http://dx.doi.org/10.1007/3-540-45053-X_48

[23] M. Karaman, L. Goldmann, D. Yu, and T. Sikora, "Comparison of static background segmentation methods," S. Li, F. Pereira, H.-Y. Shum, and A. G. Tescher, Eds., vol. 5960, no. 1. SPIE, 2005, p. 596069. [Online]. Available: http://link.aip.org/link/?PSI/5960/596069/1

[24] L. M. Fuentes and S. A. Velastin, "Foreground segmentation using luminance contrast," in *Speech, Signal and Image Processing, 2001 WSES Interational conference on*, sept 2001.

[25] A. Elgammal, R. Duraiswami, and L. S. Davis, "Efficient non-parametric adaptive color modeling using fast gauss transform," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 2, p. 563, 2001.

[26] ——, "Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 1499–1504, 2003.

[27] M. Harville, "A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models," in *Computer Vision — ECCV 2002*, ser. Lecture Notes in Computer Science, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Copenhagen, Denmark: Springer Berlin / Heidelberg, 2002, vol. 2352, pp. 37–49, 10.1007/3-540-47977-5_36. [Online]. Available: http://dx.doi.org/10.1007/3-540-47977-5_36

[28] J. W. Davis and V. Sharma, "Background-subtraction using contour-based fusion of thermal and visible imagery," *Computer Vision and Image Understanding*, vol. 106, no. 2-3, pp. 162–182, 2007, special issue on Advances in Vision Algorithms and Systems beyond the Visible Spectrum. [Online]. Available: http://www.sciencedirect.com/science/article/B6WCX-4MWY05H-2/2/f1ebd2733578866335c02199c836ec19

[29] T. Parag, A. Elgammal, and A. Mittal, "A framework for feature selection for background subtraction," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, 2006, pp. 1916–1923.

[30] N. M. Oliver, B. Rosario, and A. Pentland, "A bayesian computer vision system for modeling human interactions," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 831–843, aug. 2000.

[31] Y. Sheikh and M. Shah, "Bayesian object detection in dynamic scenes," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. 74–79, 2005.

[32] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 90–126, 2006, special Issue on Modeling People: Vision-based understanding of a person's shape, appearance, movement and behaviour. [Online]. Available: http://www.sciencedirect.com/science/article/B6WCX-4M1DB7H-1/2/8da6f6e7a8c8e07d9331bc7738c6d499

[33] M. Seki, T. Wada, H. Fujiwara, and K. Sumi, "Background subtraction based on cooccurrence of image variations," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 2, p. 65, 2003.

[34] B. Han, D. Comaniciu, and L. S. Davis, "Sequential kernel density approximation through mode propagation: Applications to background modeling," in *In proceedings. Asian Conference on Computer Vision 2004*, jan 2004.

[35] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 2, aug 2004, pp. 28–31 Vol.2.

[36] A. Azarbayejani and A. Pentland, "Real-time self-calibrating stereo person tracking using 3d shape estimation from blob features," in *International conference on Pattern Recognition*, vol. 3. Vienna: IEEE, 1996, pp. 627–632.

[37] T. Horprasert, D. Hardwood, and L. S. Davis, "A statistical approach for real-time robust background subtraction and shadow detection," in *International Conference on Computer Vision*, Corfu, Greece, 1999.

[38] X. Gao, T. Boult, and V. Ramesh, "Error characterization of detection and morphological filtering," in *Sixth International Symposium on Mathematical Morphology*, April 2002, pp. 347–359.

[39] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[40] J. Renno, N. Lazarevic-McManus, D. Makris, and G. Jones, "Evaluating motion detection algorithms: issues and results," in *IEEE International Workshop on Visual Surveillance*, May 2006, pp. 97–104.

[41] C. Dai, Y. Zheng, and X. Li, "Pedestrian detection and tracking in infrared imagery using shape and appearance," *Computer Vision and Image Understanding*, vol. 106, no. 2-3, pp. 288–299, 2007, special issue on Advances in Vision Algorithms and Systems beyond the Visible Spectrum. [Online]. Available: http://www.sciencedirect.com/science/article/B6WCX-4MK5GP7-2/2/55c87f808168e496e6853a82394609ef

[42] D. Olmeda, A. de la Escalera, and J. M. Armingol, "Detection and tracking of pedestrians in infrared images," in *Signals, Circuits and Systems (SCS), 2009 3rd International Conference on*, nov. 2009, pp. 1–6.

[43] D. M. Gavrila, "A bayesian, exemplar-based approach to hierarchical shape matching," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 8, pp. 1408–1421, aug. 2007.

[44] L. Malagon-Borja and O. Fuentes, "Object detection using image reconstruction with pca," *Image and Vision Computing*, vol. 27, no. 1-2, pp. 2–9, 2009, canadian Robotic Vision 2005 and 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0262885607000820

[45] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1997, pp. 207–214.

[46] Y. Owechko, S. Medasani, and N. Srinivasa, "Classifier swarms for human detection in infrared imagery," in *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, june 2004, p. 121.

[47] M. Yasuno, N. Yasuda, and M. Aoki, "Pedestrian detection and tracking in far infrared images," *Computer Vision and Pattern Recognition Workshop*, vol. 8, p. 125, 2004.

[48] C. Dai, Y. Zheng, and X. Li, "Layered representation for pedestrian detection and tracking in infrared imagery," in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, june 2005, p. 13.

[49] T. Zhao and R. Nevatia, "Tracking multiple humans in complex situations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 9, pp. 1208–1221, sept. 2004.

[50] F. Xu, X. Liu, and K. Fujimura, "Pedestrian detection and tracking with night vision," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 6, no. 1, pp. 63–71, march 2005.

[51] L. Zhang, B. Wu, and R. Nevatia, "Pedestrian detection in infrared images based on local shape features," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, june 2007, pp. 1–8.

[52] A. G. Barath Kumar, K. E. Daigle, M. G. Pandy, Q. Cai, and J. K. Aggarwal, "Lower limb kinematics of human walking with the medial axis transformation," in *IEEE Computer Society Workshop on Motion of Non Rigid and Articulated Objects*. Austin, TX: IEEE, 1994, pp. 70–76.

[53] S. A. Niyogi and E. H. Adelson, "Analysing and recognising walking figures in xyt," in *IEEE Computer Society conference on Computer Vision and Pattern Recognition*, Seattle, 1994, pp. 469–474.

[54] A. Shio and J. Sklansky, "Segmentation of people in motion," in *IEEE workshop on Visual Motion*. Princeton: IEEE, 1991, pp. 325–332.

[55] J. O'Rourke and N. I. Badler, "Model-based image analysis of human motion using constraint propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 522–536, November 1980.

[56] A. E. C. Pece, "Generative-model-based tracking by cluster analysis of image differences," *Robotics and Autonomous Systems*, vol. 39, no. 3-4, pp. 181–194, 2002.

[57] A. Yilmaz, "Object tracking by asymmetric kernel mean shift with automatic scale and orientation selection," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 0, pp. 1–6, 2007.

[58] G. D. Jones, R. E. Allsop, and J. H. Gilby, "Bayesian analysis for fusion of data from disparate imaging systems for surveillance," *Image and Vision Computing*, vol. 21, no. 10, pp. 843–849, 2003.

[59] M. Meuter, D. Müller, S. Müller-Schneiders, U. Iurgel, S.-B. Park, and A. Kummert, "Pedestrian tracking from a moving host using corner points," in *Proceedings of the 3rd international conference on Advances in visual computing - Volume Part II*, ser. ISVC'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 367–376. [Online]. Available: http://dl.acm.org/citation.cfm?id=1779090.1779133

[60] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, 1st ed. Wiley-Interscience, 1997, ch. 3.

[61] S. M. Stigler, "Thomas bayes's bayesian inference," *Journal of the Royal Statistical Society. Series A (General)*, vol. 145, no. 2, pp. 250–258, 1982.

[62] A. J. Abrantes, J. S. Marques, and J. M. Lemos, "Long term tracking using bayesian networks," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 3, june 2002, pp. 609–612 vol.3.

[63] M. Cristani, M. Bicego, and V. Murino, "Multi-level background initialization using hidden markov models," in *First ACM SIGMM international workshop on Video surveillance*, ser. IWVS '03. New York, NY, USA: ACM, 2003, pp. 11–20. [Online]. Available: http://doi.acm.org/10.1145/982452.982455

[64] C.-J. Pai, H.-R. Tyan, Y.-M. Liang, H.-Y. M. Liao, and S.-W. Chen, "Pedestrian detection and tracking at crossroads," *Pattern Recognition*, vol. 37, no. 5, pp. 1025–1034, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320303003959

[65] K. Bashir, T. Xiang, and S. Gong, "Feature selection for gait recognition without subject cooperation," in *British Machine Vision Conference*. Leeds: British Machine Vision Association, 2008.

[66] I. Bouchrika and M. Nixon, "People detection and recognition using gait for automated visual surveillance," in *The Institution of Engineering and Technology Conference on Crime and Security.*, June 2006, pp. 576–581.

[67] C. Y. Yam, M. S. Nixon, and J. N. Carter, "On the relationship of human walking and running: Automatic person identification by gait," *Proc. of Intl. Conf. on Pattern Recognition*, 2002.

[68] S. Maji and J. Malik, "Object detection using a max-margin hough transform," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, jun 2009, pp. 1038–1045.

[69] S. Intille, J. Davis, and A. Bobick, "Real-time closed-world tracking," in *Computer Vision and Pattern Recognition, IEEE Conference on*. IEEE Computer Society, 1997, pp. 697–703.

[70] X. Desurmont, R. Wijnhoven, E. Jaspers, O. Caignart, M. Barais, W. Favoreel, and J.-F. Delaigle, "Performance evaluation of real-time video content analysis systems in the candela project," N. Kehtarnavaz and P. A. Laplante, Eds., vol. 5671, no. 1. SPIE, 2005, pp. 200–211. [Online]. Available: http://link.aip.org/link/?PSI/5671/200/1

[71] N. Fernández-García, A. Carmona-Poyato, R. Medina-Carnicer, and F. Madrid-Cuevas, "Automatic generation of consensus ground truth for the comparison of

edge detection techniques," *Image and Vision Computing*, vol. 26, no. 4, pp. 496–511, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/B6V09-4P4FV2R-1/2/9555eef5a9713cb5a38eecf466b30c63

[72] J. Black, T. Ellis, and P. Rosin, "A novel method for video tracking performance evaluation," in *In Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS*, 2003, pp. 125–132.

[73] J. G. Daugman, "Two-dimensional spectral analysis of cortical receptive field profiles," *Vision Research*, vol. 20, no. 10, pp. 847 – 856, 1980. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0042698980900656

[74] F. Zhou, J. F. Feng, and Q. Y. Shi, "Texture feature based on local fourier transform," in *International Conference on Image Processing, 2001*, vol. 2, 2001, pp. 610 – 613.

[75] J. Davis and M. A. Keck, "A two-stage approach to person detection in thermal imagery," in *In proceedings. Workshop on Applications of Computer Vision*, jan 2005.

[76] C. O. Conaire, N. E. O'Connor, and A. Smeaton, "Thermo-visual feature fusion for object tracking using multiple spatiogram trackers," *Journal of Machine Vision and Applications*, vol. 19, no. 5, pp. 483–494, may 2007.

[77] L. M. Fuentes and S. A. Velastin, "People tracking in surveillance applications," *Image and Vision Computing*, vol. 24, no. 11, pp. 1165–1171, 2006, performance Evaluation of Tracking and Surveillance. [Online]. Available: http://www.sciencedirect.com/science/article/B6V09-4GX6HWV-1/2/b5c21aff308ca6e86776b01da64f4591

[78] P. Kruse, E. Weber, R. Willardson, and D. Skatrud, *Uncooled Infrared Imaging Arrays and Systems,*, ser. SEMICONDUCTORS AND SEMIMETALS. ACADEMIC PRESS, 1997, vol. 47, ch. 2, p. 38.

[79] K. Livingston, "Mayors transport strategy," 2001.

[80] R. Kukla, J. Kerridge, A. Willis, and J. Hine, "Pedflow: Development of an autonomous agent model of pedestrian flow," *Transportation Research Record*, no. 1774 / 2001, pp. 11–17, 2001.

[81] J. Desyllas, E. Duxbury, J. Ward, and A. Smith, "Pedestrian demand modelling of large cities: An applied example from london," p. 17, 2003.

[82] W. Brog, "Walking- a negelected mode in transport surveys." in *Australia: Walking the 21st Century*. Perth: Department for Planning and Transport: Goverment of Western Australia, 2001, p. 12.

[83] K. Teknomo, "Microscopic pedestrian flow characteristics: Devlopment of an image processing data collection and simulation model," Ph.D. dissertation, Tohoku University, 2002.

[84] S. Velastin, B. Boghossian, and M. Vicencio-Silva, "A motion-based image processing system for detecting potentially dangerous situations in underground railway stations. , ." *Transportation Research Part C*, no. 14, pp. 96–114, 2006.

[85] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review," *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 428–440, 1999.

Appendix A

Data for Tables - Misclassified Targets and Frame Rates

Table A.1: Missed Targets for All Sequences

Probablistic Template Classifier			
4-6	**9-11**	**14-15**	**19-21**
0.00	3.00	5.00	9.00
0.00	4.00	5.00	8.00
0.00	4.00	5.00	9.00
0.00	4.00	6.00	8.00
0.00	3.00	6.00	9.00
0.00	4.00	6.00	9.00
0.00	3.00	6.00	10.00
1.00	3.00	6.00	8.00
0.00	3.00	6.00	10.00
1.00	4.00	5.00	9.00
1.00	3.00	6.00	10.00
1.00	3.00	5.00	10.00
0.00	4.00		
0.00	4.00		
	4.00		

Histogram Symmetry Classifier			
4-6	**9-11**	**14-15**	**19-21**
0.00	3.00	4.00	7.00
0.00	3.00	5.00	6.00
0.00	2.00	4.00	7.00
0.00	2.00	4.00	7.00
0.00	3.00	4.00	7.00
0.00	2.00	4.00	7.00

Continued on Next Page...

Table A.1 – Continued

4-6	9-11	14-15	19-21
0.00	2.00	5.00	6.00
0.00	2.00	5.00	7.00
0.00	3.00	4.00	7.00
0.00	3.00	5.00	6.00
0.00	3.00	5.00	7.00
0.00	3.00	5.00	7.00
0.00	2.00		
0.00	2.00		
	2.00		

Principal Components Analysis Classifier

4-6	9-11	14-15	19-21
0.00	1.00	3.00	6.00
0.00	0.00	4.00	6.00
0.00	0.00	4.00	5.00
0.00	0.00	4.00	5.00
0.00	1.00	3.00	6.00
0.00	0.00	4.00	6.00
0.00	1.00	3.00	5.00
0.00	1.00	4.00	6.00
0.00	1.00	4.00	5.00
0.00	1.00	3.00	5.00
0.00	0.00	4.00	5.00
0.00	1.00	4.00	5.00
0.00	0.00		
0.00	0.00		
	0.00		

Binary Template Classifier

4-6	9-11	14-15	19-21
0.00	0.00	3.00	6.00
0.00	1.00	4.00	5.00
0.00	0.00	3.00	5.00
0.00	1.00	4.00	6.00
0.00	1.00	4.00	6.00
0.00	1.00	4.00	6.00

Continued on Next Page...

Table A.1 – Continued

4-6	**9-11**	**14-15**	**19-21**
0.00	0.00	3.00	7.00
0.00	1.00	4.00	6.00
0.00	1.00	3.00	6.00
0.00	2.00	4.00	7.00
0.00	1.00	5.00	7.00
0.00	1.00	4.00	7.00
0.00	2.00		
0.00	1.00		
	2.00		

Table A.2: T_{dr} Rates for all Classifiers, Bounding Box Tracker

Probablistic Template Classifier			
4-6	**9-11**	**14-15**	**19-21**
0.97	0.81	0.77	0.69
0.97	0.83	0.74	0.68
0.95	0.81	0.76	0.65
0.95	0.85	0.74	0.68
0.92	0.80	0.75	0.68
0.98	0.81	0.77	0.69
0.97	0.78	0.75	0.68
0.92	0.85	0.77	0.65
0.98	0.83	0.75	0.67
0.92	0.78	0.73	0.69
0.93	0.83		
0.96	0.81		
	0.81		

Histogram Symmetry Classifier			
4-6	**9-11**	**14-15**	**19-21**
0.98	0.88	0.81	0.73
0.96	0.89	0.80	0.75
0.98	0.89	0.81	0.75
0.98	0.88	0.80	0.73
0.96	0.89	0.80	0.72
0.95	0.88	0.82	0.73
0.95	0.88	0.80	0.74

Continued on Next Page...

Table A.2 – Continued

4-6	**9-11**	**14-15**	**19-21**
0.96	0.87	0.80	0.73
0.98	0.88	0.80	0.73
0.96	0.89	0.82	0.75
0.96	0.89		
0.95	0.88		
	0.88		

Principal Components Analysis Classifier

4-6	**9-11**	**14-15**	**19-21**
0.97	0.90	0.86	0.78
0.96	0.91	0.84	0.77
0.95	0.92	0.86	0.80
0.98	0.89	0.84	0.80
0.98	0.90	0.84	0.78
0.98	0.93	0.87	0.80
0.99	0.89	0.86	0.78
0.95	0.89	0.87	0.81
0.96	0.89	0.86	0.81
0.96	0.89	0.84	0.77
0.95	0.90		
0.96	0.93		
	0.91		

Binary Template Classifier

4-6	**9-11**	**14-15**	**19-21**
0.99	0.92	0.83	0.73
0.95	0.89	0.82	0.74
0.97	0.90	0.82	0.73
0.95	0.90	0.83	0.74
0.98	0.93	0.82	0.73
0.95	0.89	0.82	0.73
0.98	0.93	0.82	0.74
0.97	0.92	0.83	0.74
0.95	0.91	0.83	0.73
0.99	0.93	0.82	0.73
0.99	0.93		

Continued on Next Page...

Table A.2 – Continued

4-6	**9-11**	**14-15**	**19-21**
0.98	0.89		
	0.93		

Table A.3: T_{dr} Rates for all Classifiers, Intensity Codebook Tracker

Principal Components Analysis Classifier

4-6	**9-11**	**14-15**	**19-21**
0.97	0.90	0.89	0.82
0.98	0.93	0.87	0.80
0.95	0.91	0.90	0.81
0.95	0.90	0.88	0.81
0.98	0.91	0.90	0.79
0.96	0.93	0.87	0.79
0.99	0.90	0.87	0.82
0.96	0.93	0.88	0.82
0.97	0.90	0.88	0.82
0.98	0.91	0.90	0.81
0.97	0.92		
0.99	0.90		
	0.90		

Binary Template Classifier

4-6	**9-11**	**14-15**	**19-21**
0.96	0.92	0.87	0.76
0.96	0.93	0.85	0.77
0.96	0.93	0.87	0.76
0.99	0.92	0.85	0.77
0.99	0.92	0.86	0.79
0.95	0.91	0.87	0.78
0.99	0.91	0.85	0.78
0.95	0.91	0.85	0.79
0.96	0.92	0.86	0.77
0.97	0.93	0.87	0.77
0.96	0.90		
0.98	0.92		
	0.92		

Table A.4: Frame Rates for Bounding Box Tracker on PC

Probablistic Template Classifier			
4-6	**9-11**	**14-15**	**19-21**
20.00	17.00	15.00	12.00
20.00	18.00	14.00	12.00
21.00	16.00	14.00	11.00
21.00	18.00	14.00	12.00
20.00	17.00	15.00	11.00
19.00	17.00	14.00	12.00
19.00	18.00	14.00	12.00
21.00	16.00	14.00	12.00
22.00	16.00	14.00	11.00
19.00	17.00	14.00	12.00
19.00	16.00		
21.00	17.00		
	16.00		

Histogram Symmetry Classifier			
4-6	**9-11**	**14-15**	**19-21**
20.00	17.00	14.00	13.00
19.00	18.00	14.00	11.00
19.00	18.00	14.00	13.00
20.00	18.00	15.00	12.00
20.00	18.00	15.00	13.00
18.00	18.00	15.00	11.00
19.00	18.00	14.00	12.00
20.00	18.00	14.00	11.00
20.00	18.00	15.00	13.00
18.00	18.00	14.00	11.00
18.00	18.00		
18.00	17.00		
	17.00		

Principal Components Analysis Classifier			
4-6	**9-11**	**14-15**	**19-21**
17.00	16.00	13.00	9.00
17.00	15.00	14.00	9.00

Continued on Next Page...

Table A.4 – Continued

4-6	9-11	14-15	19-21
17.00	15.00	13.00	9.00
16.00	16.00	13.00	9.00
16.00	15.00	13.00	10.00
17.00	15.00	13.00	9.00
17.00	15.00	14.00	10.00
17.00	15.00	14.00	9.00
16.00	15.00	13.00	10.00
17.00	16.00	14.00	9.00
17.00	15.00		
17.00	16.00		
	15.00		

Binary Template Classifier

4-6	9-11	14-15	19-21
20.00	17.00	15.00	12.00
18.00	18.00	13.00	14.00
17.00	17.00	13.00	12.00
20.00	18.00	13.00	13.00
20.00	16.00	15.00	14.00
20.00	16.00	15.00	14.00
19.00	18.00	13.00	12.00
20.00	18.00	13.00	12.00
18.00	18.00	13.00	14.00
17.00	18.00	15.00	12.00
17.00	17.00		
20.00	18.00		
	17.00		

Table A.5: PC Frame Rates for all Classifiers, Intensity Codebook Tracker

Principal Components Analysis Classifier

4-6	9-11	14-15	19-21
16.00	15.00	13.30	11.00
17.00	16.00	13.40	11.00
17.00	15.00	13.20	10.80
16.00	16.00	13.10	10.00
16.00	16.00	13.10	9.90

Continued on Next Page...

Table A.5 – Continued

4-6	**9-11**	**14-15**	**19-21**
16.00	15.00	13.30	11.00
17.00	16.00	14.40	10.70
17.00	15.00	13.10	10.40
16.00	16.00	13.20	10.70
17.00	16.00	13.60	10.80
17.00	16.00		
16.00	15.00		
	15.00		

Binary Template Classifier

4-6	**9-11**	**14-15**	**19-21**
17.00	16.00	14.40	12.60
20.00	17.00	15.20	12.50
17.00	17.00	14.10	12.60
18.00	16.00	14.10	13.00
17.00	18.00	15.40	14.00
19.00	16.00	14.40	13.50
19.00	16.00	14.90	13.80
18.00	16.00	14.90	13.30
18.00	17.00	14.10	13.40
19.00	16.00	14.90	13.20
19.00	17.00		
18.00	18.00		
	17.00		

Table A.6: Frame Rates for Bounding Box Tracker on PS3

Probablistic Template Classifier

4-6	**9-11**	**14-15**	**19-21**
18.00	15.00	10.00	8.00
18.00	15.00	11.00	9.00
18.00	14.00	12.00	9.00
18.00	13.00	9.00	8.00
18.00	14.00	11.00	8.00
18.00	14.00	11.00	8.00
18.00	15.00	11.00	8.00
19.00	14.00	10.00	8.00

Continued on Next Page...

Table A.6 – Continued

4-6	9-11	14-15	19-21
18.00	15.00	9.00	9.00
19.00	15.00	11.00	8.00
19.00	13.00		
19.00	13.00		
	13.00		

Histogram Symmetry Classifier

4-6	9-11	14-15	19-21
19.00	17.00	14.00	11.00
20.00	17.00	14.00	11.00
19.00	17.00	12.00	12.00
19.00	17.00	12.00	11.00
20.00	16.00	13.00	11.00
20.00	17.00	14.00	12.00
20.00	17.00	12.00	11.00
19.00	17.00	12.00	12.00
20.00	16.00	13.00	11.00
19.00	16.00	14.00	12.00
20.00	16.00		
19.00	16.00		
	16.00		

Principal Components Analysis Classifier

4-6	9-11	14-15	19-21
14.00	10.00	9.00	6.00
14.00	11.00	10.00	6.00
13.00	11.00	10.00	6.00
13.00	10.00	8.00	6.00
13.00	10.00	9.00	7.00
14.00	10.00	9.00	6.00
13.00	11.00	8.00	6.00
14.00	11.00	9.00	6.00
14.00	10.00	9.00	7.00
14.00	10.00	9.00	7.00
13.00	10.00		
14.00	11.00		

Continued on Next Page...

Table A.6 – Continued

4-6	9-11	14-15	19-21
	10.00		

Binary Template Classifier

4-6	9-11	14-15	19-21
19.00	17.00	13.00	10.00
18.00	16.00	13.00	10.00
19.00	16.00	13.00	10.00
20.00	16.00	13.00	11.00
19.00	15.00	12.00	10.00
19.00	17.00	13.00	10.00
19.00	17.00	12.00	10.00
19.00	16.00	12.00	11.00
20.00	17.00	13.00	10.00
19.00	17.00	13.00	10.00
18.00	16.00		
20.00	17.00		
	15.00		

Table A.7: PS3 Frame Rates for all Classifiers, Intensity Codebook Tracker

Principal Components Analysis Classifier

4-6	9-11	14-15	19-21
14.00	10.00	9.00	8.30
13.00	10.00	8.90	7.90
14.00	10.00	9.60	9.00
14.00	10.00	10.00	7.80
13.00	11.00	9.90	7.20
14.00	11.00	8.50	7.30
14.00	10.00	8.60	7.40
14.00	11.00	10.00	7.30
13.00	11.00	9.70	7.70
13.00	11.00	9.00	7.70
13.00	11.00		
13.00	10.00		
	10.00		

Continued on Next Page...

Table A.7 – Continued

4-6	9-11	14-15	19-21
Binary Template Classifier			
4-6	**9-11**	**14-15**	**19-21**
20.00	16.00	13.40	10.90
19.00	15.00	14.40	10.50
20.00	16.00	13.70	10.30
18.00	16.00	13.20	10.90
18.00	15.00	14.30	11.30
19.00	15.00	13.70	10.00
20.00	17.00	13.00	11.20
19.00	16.00	13.30	11.40
18.00	15.00	13.40	11.50
18.00	15.00	13.90	10.50
20.00	15.00		
19.00	16.00		
	17.00		

Table A.8: Frame Rates for Bounding Box Tracker on Wii

Probablistic Template Classifier			
4-6	**9-11**	**14-15**	**19-21**
17.00	12.00	6.00	5.00
16.00	12.00	7.00	3.00
16.00	11.00	7.00	3.00
17.00	12.00	7.00	5.00
16.00	11.00	6.00	3.00
16.00	12.00	7.00	5.00
17.00	12.00	6.00	4.00
17.00	11.00	7.00	3.00
16.00	11.00	6.00	4.00
17.00	12.00	6.00	4.00
17.00	12.00		
17.00	11.00		
	12.00		
Histogram Symmetry Classifier			
4-6	**9-11**	**14-15**	**19-21**

Continued on Next Page...

Table A.8 – Continued

4-6	9-11	14-15	19-21
14.00	10.00	6.00	2.00
13.00	11.00	5.00	3.00
13.00	10.00	6.00	2.00
14.00	10.00	6.00	3.00
13.00	11.00	7.00	3.00
12.00	10.00	7.00	2.00
14.00	10.00	5.00	3.00
13.00	11.00	5.00	2.00
15.00	11.00	7.00	3.00
15.00	11.00	7.00	2.00
12.00	11.00		
13.00	10.00		
	10.00		

Principal Components Analysis Classifier

4-6	9-11	14-15	19-21
13.00	9.00	7.00	4.00
11.00	9.00	8.00	2.00
12.00	9.00	7.00	4.00
12.00	9.00	8.00	2.00
12.00	9.00	6.00	4.00
12.00	10.00	7.00	3.00
12.00	10.00	7.00	3.00
11.00	9.00	6.00	4.00
12.00	10.00	6.00	4.00
13.00	9.00	7.00	4.00
11.00	9.00		
11.00	9.00		
	10.00		

Binary Template Classifier

4-6	9-11	14-15	19-21
12.00	11.00	7.00	4.00
12.00	10.00	6.00	2.00
12.00	12.00	8.00	4.00
13.00	9.00	6.00	2.00

Continued on Next Page...

Table A.8 – Continued

4-6	9-11	14-15	19-21
14.00	10.00	6.00	4.00
13.00	12.00	8.00	4.00
14.00	9.00	7.00	4.00
14.00	11.00	8.00	2.00
13.00	11.00	6.00	4.00
14.00	9.00	7.00	3.00
13.00	9.00		
14.00	11.00		
	9.00		

Table A.9: Wii Frame Rates for all Classifiers, Intensity Codebook Tracker

Principal Components Analysis Classifier			
4-6	**9-11**	**14-15**	**19-21**
13.00	9.40	6.80	2.20
13.00	9.40	7.50	2.20
11.00	9.20	6.50	3.90
13.00	9.80	5.70	1.80
12.00	9.70	7.50	3.00
12.00	9.00	6.80	3.30
13.00	8.70	5.90	2.70
12.00	8.70	6.30	3.00
12.00	8.50	5.60	3.80
11.00	8.70	6.30	3.70
11.00	9.80		
13.00	9.60		
	8.90		

Binary Template Classifier			
4-6	**9-11**	**14-15**	**19-21**
14.00	9.00	6.40	3.00
14.00	10.60	8.20	4.30
12.00	10.30	7.40	2.60
14.00	10.10	6.00	4.30
13.00	10.80	7.20	2.90
13.00	10.50	6.60	2.90
13.00	10.10	7.20	4.00

Continued on Next Page...

Table A.9 – Continued

4-6	**9-11**	**14-15**	**19-21**
14.00	11.00	7.80	3.00
14.00	10.50	6.10	4.30
14.00	9.90	7.00	2.80
14.00	10.60		
13.00	10.60		
	10.60		

Appendix B

Infrared Radiation and Sensors

The technology that is used to record infrared (digital) images is complex. An understanding of the technology used for IR imaging will help in understanding the challenges of working with the same. As can be seen in the in Figure B.1 infrared (IR) radiation is

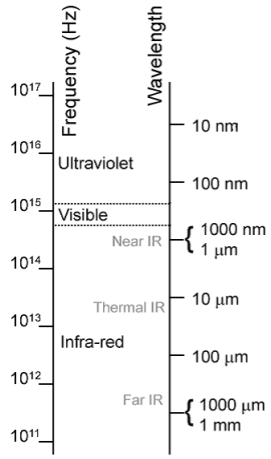


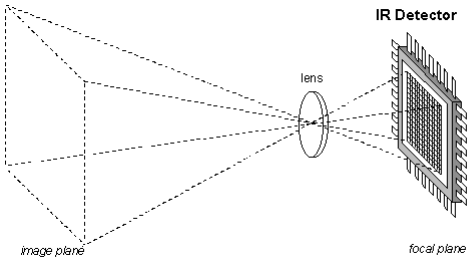
Figure B.1: Electro-magnetic spectrum

classed into three types, near, thermal and far infrared. This classification of the types of radiation is based on frequency and wavelength of the radiation and the most common source of the radiation at those wavelengths.

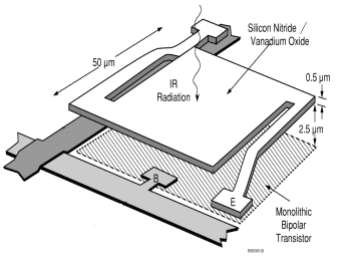
- **Near Infrared**: This is infrared radiation that is at frequencies just below that of visible light.
- **Thermal Infrared**: This is infrared radiation emitted by objects due to their temperature.
- **Far Infrared**: This band of infrared radiation is radiated by extremely distant objects, e.g. galaxies, black-holes etc.

Modern infrared cameras are focal plane array (FPA) cameras [2], i.e. the sensor array is placed at the along the focal plane of the lens (Figure B.2b). The focal plane array used in a thermal camera could either be cooled or uncooled. The arrays used in cooled cameras are usually cooled to cryogenic temperatures. This increases the sensitivity of the sensor

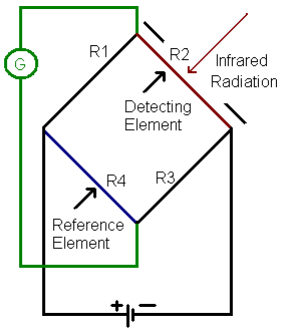
to longer wavelengths, this type of array is usually used for astro-physics to record far IR spectrum images.



(a) Line drawing of image formation



(b) Micro-bolometer schematic [78]



(c) 1 element of the array from B.2b [78]

Figure B.2: Focal Plane Array

Most modern thermal IR cameras are uncooled, this makes them more portable at the cost of sensitivity and decreased signal to noise ratio. Operating the uncooled cameras with narrower bandwidths improves the signal to noise ratio. Figure B.2c is a simplified diagram of a bolometer, it consists of a Wheatstone bridge where one resistive element is exposed to incident radiation. The entire bridge is made of the same material, in this case V_2O_5 (Vanadium Oxide). While machining the sensor elements in micro-bolometers as used in modern uncooled IR cameras the element that is exposed to infrared radiation also acts as a shield for the reference resistive element. A micro-bolometer functions by detecting changes/fluctuations in the current flowing through the Wheatstone bridge, as all the elements are static the only changes in current that occur are those that are caused by the infrared radiation.

The sensitivity of a bolometer is controlled by the sensitivity of the galvanometer G that is connected to the bolometer. However as the elements are all resistive in nature, after some time they begin to heat up and the current drifts and the bias of the bridge changes. To compensate for this drift, the control electronics shutter the camera so that no external IR radiation is incident on the camera. After shuttering the camera a *'zero temperature'* body is rotated in front of the array. The temperature recorded for this object is used to measure the drift in the bias current and filter it out of the final image. One detailed guide

to understanding the workings of a micro-bolometer infrared camera is the book by Kruse et al [2]

Appendix C
Application Areas

This appendix will present a brief overview of application areas for pedestrian tracking systems. Some of the areas in which pedestrian tracking data is currently used are; pedestrian modelling, building safety, pedestrian safety, surveillance and area control and bio mechanics. While some of the application areas are easily identified, others are more esoteric, hence the following paragraphs will elaborate.

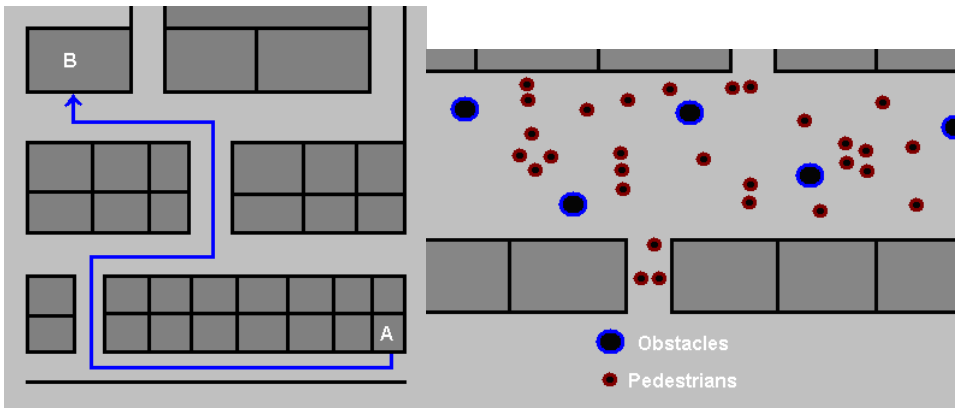
C.1 Pedestrian Modelling

In recent years there has been increasing amount of research into walking as a mode of transport, in view of the need to develop more sustainable modes of transportation. This is due to an increase in the awareness of the impact of short motorised journeys on the environment and the associated increase in vehicular densities on roads; increasing the amount of infrastructure that is needed to support the same [79].

To reduce vehicular traffic an increase in the number of walking journeys is needed. Also to encourage more use of alternate means of transport, planning engineers have been modelling how pedestrian spaces are used [80, 81]. This modelling and simulation needs raw data in terms of surveys of the number of people using walking as a mode of transportation, what types of journeys they use it for, whether they feel comfortable using existing pedestrian spaces [82] etc.

When qualitative data is gathered using surveys of pedestrians, lacks information on behaviour that may not occur due to the presence of people conducting the survey. This missing information affects the accuracy of how the models simulate pedestrian behaviour. At locations where CCTV cameras have been installed, the information from this can be used to collect data for the models, but this is not possible at all locations. Pedestrian environments are modelled as either macroscopic or microscopic.

Macroscopic Modelling: This is when the model is generated with the aim of simulating pedestrian flows across large areas, for example the pedestrian flows in Central London [81]. The aim of this type of simulation is to help integrate pedestrian traffic with other forms of transportation and to minimise the time that is needed by a pedestrian to get from point A to point B (figure C.1a). This is useful while simulating commuting pedestrians and to improve the public transportation infrastructure.



(a) Macroscopic Model (b) Microscopic Model

Figure C.1: Pedestrian Environments

Microscopic Modelling: In this type of model (Figure C.1b) the area in consideration is smaller. It could be a street, or something that is a short distance away from the pedestrian. The goal is pre-determined, but may be flexible. The information that is available to the simulated pedestrian is the destination and whatever is in the line of sight. While this type of modelling is useful in simulating how pedestrians would interact at train stations etc, it is more often used for simulating pedestrian behaviour in shopping malls or on single streets [83].

C.2 Building Safety

In this application area pedestrian tracking is used to generate the profile of normal motion within a designated area, for example, the pedestrian movement in a station [84], or the movement of pedestrians on a pavement etc. This information is then used to flag situations wherein the movement is outside the norm and alert the operator.

This technology can also be used to monitor stretches of busy roads and to generate a profile of normal activity on the road. Using this profile, any incident that affects both traffic flow and hampers pedestrian movement is flagged. This technology could also be capable of warning the operator of any situations that might result in accidents, allowing the operator sufficient time to take preventive measures. The preventive measures could be anything from dispatching the emergency services in case of accidents, or diverting the traffic away from the road if the density is too high etc.

C.3 Access Control and Surveillance

Access control is similar to pedestrian safety. The difference between the two applications is that while the system is monitoring a predefined area, in addition to tracking the movements of people it has to identify people entering and leaving. After identifying the people entering the predefined area it then tracks that person's movement within this area till they exit the area under observation.

Figure C.2: Crowd Scene

This type of tracking needs to be more accurate than that used in either pedestrian modelling or in pedestrian safety. This is because the former is primarily interested in pedestrian flows and the latter is in the interactions. Even if there are some errors in the pedestrian flow data they can be removed as statistical errors by filtering the data while using the information for modelling. For systems used in pedestrian safety applications, the tracking accuracy does not need to be very high; it just needs to be able to detect abnormal movement in pedestrian flow. If an object does get occluded, it doesn't matter if the system temporarily loses one target as long as it is able to identify two objects after the occlusion event is over.

The level of accuracy in use with the former two applications isn't enough to keep a continuous track of pre-identified targets within a designated safe zone. In most cases the number of objects being tracked within a predefined restricted zone will be low, as the access is restricted. This is not always the case, an example of an access restricted areas which require large numbers of objects to be very accurately tracked is an airport. The image quality within an airport is easily controlled, by placing the cameras in locations with good fields of view. The movement of people could also be regulated to make it possible to easily record people entering and leaving. The challenge lies in accurately tracking people *after* they enter.

In the image taken from a protest (Figure C.2) it is obvious that though the camera is placed at a good vantage point the number of targets that need to be tracked is very large. The number of targets and the area that needs to be covered increases the complexity of the system that is needed to accurately track targets from one camera to another.

C.4 Bio-Mechanics

In this area of research automatic capture and analysis of human body motion is used to understand how biological systems work. The images of people in motion are recorded and are then analysed to provide information on the positions of the human extremities such as the arms, legs etc. This area of research is diverse and for more information the reader is referred to Moeslund et al [32], Aggarwal and Cai [85].

Table C.1: Summary of Application Areas

Application Area	Group Size	Tracking		Event Detection	
		Accuracy	Time	Accuracy	Time
Modelling	Small groups to large crowds	$> 80\%$	Offline	75%	$10-15s$
Building Safety	Small groups to large crowds	70%	$5s$	97%	$2s$
Surveillance	Small groups	$> 95\%$	$2s$	90%	$10s$
Bio-mechanics	*N/A*	Body parts	Offline	*N/A*	Offline
Pedestrian Safety	Small Groups	$> 95\%$	$1s$	*N/A*	*N/A*

C.5 Pedestrian Safety

For this application area, the systems are used to identify pedestrians in real time to alert drivers so as to avoid potential accidents. Most of the research in this field use IR cameras as the source of video information [13]. The systems developed for this application area need to be extremely accurate with a low false positive rate.

The fast response time is needed to give the operator of the vehicle sufficient time to respond to changes in pedestrian traffic on a road. People learn to ignore alerts by the system if it has a high false positive rates.

C.6 Application Areas Summary

The information about the application areas is summarised in the table (C.1). From this it is obvious that it is very difficult to make a *'one size fits all'* tracking system. Also engineering one would unnecessarily increase the cost and complexity of the system. The response times and accuracy requirements for the different application areas are estimates obtained by examining literature that address pedestrian tracking systems specifically for that application area.