# Multi-Modal People Detection from Aerial Video


Helen Flynn

Keble College

University of Oxford

A thesis submitted for the degree of

*Doctor of Philosophy*

Hilary 2015

# Abstract

There has been great interest in the use of small robotic helicopter vehicles over the last few years. Although there are regulatory issues involved in flying these that are still to be solved, they have the potential to provide a practical mobile aerial platform for a small fraction of the cost of a conventional manned helicopter. One potential class of applications for these is in searching for people, and this thesis explores a new generation of cameras which are suitable for this purpose.

We propose HeatTrack, a novel algorithm to detect and track people in aerial imagery taken from a combined infrared/visible camera rig. A Local Binary Patterns (LBP) detector finds silhouettes in the infrared image which are used guide the search in the visible light image, and a Kalman filter combines information from both modalities in order to track a person more accurately than if only a single modality were available. We introduce a method for matching the thermal signature of a person to their corresponding patch in the visible modality, and show that this is more accurate than traditional homography-based matching. Furthermore, we propose a method for cancelling out camera motion which allows us to estimate a velocity for the person, and this helps in determining the location of a person in subsequent frames.

HeatTrack demonstrates several advantages over tracking in the visible domain only, particularly in cases where the person shows up clearly in infrared. By narrowing down the search to the warmer parts of a scene, the detection of a person is faster than if the whole image were searched. The use of two imaging modalities instead of one makes the system more robust to occlusion; this, in combination with estimation of the velocity of a person, enables tracking even when information is lacking in either modality. To the best of our knowledge, this is the first published algorithm for tracking people in aerial imagery using a combined infrared/visible camera setup.

# Acknowledgements

First and foremost, I am extremely grateful to my supervisor Stephen Cameron. Stephen originally supervised my MSc thesis and was good enough to recommend I continue for a DPhil, something which I would likely not have considered without his encouragement. His guiding approach to supervision has made my time as a DPhil student all the more enjoyable and rewarding. I am particularly grateful to him for the considerable amount of time he spent in setting up my camera rig for experiments.

Special thanks go to my colleague Victor Spirin for the hours and hours of fun we had in Skylab and at the various RoboCup excursions abroad. My conversations with Victor helped me look at my work from different angles, and to think through things rationally where I might have gotten carried away. When it was time for a break from work, Victor was always there for fun times. I will never forget the time we had to use Stephen's stereoscopic fishing rod to rescue the orange flying fish from the roof of the atrium! I am also grateful to Victor for proof-reading this thesis.

Finally, I would like to thank my family for their love, support and understanding throughout the last four years. Their confidence in me has always provided an extra motivation. Without their support I would not have come to Oxford in the first place, let alone have completed a DPhil.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

There has been great interest in the use of small robotic helicopter vehicles over the last couple of years. Although there are regulatory issues involved in flying these that are still to be solved, they have the potential to provide a practical mobile aerial platform for a small fraction of the cost of a conventional manned helicopter. One potential class of applications for these is in searching for people. Police helicopters are regularly used for this purpose in the UK and elsewhere, where a trained human 'spotter' will typically use a combination of infrared (IR) and visible light imaging to seek people lost or hiding from the authorities. The equipment used for this purpose on manned helicopters is typically large, heavy and expensive, and relies on the spotter to detect the targets. The primary purpose of this DPhil is to explore the use of a new generation of cameras that may be suitable for this task on small unmanned aerial vehicles (UAVs). Communications between such a UAV and a human controller may be unreliable or limited, so it would be useful to be able to use new computer hardware and image processing techniques to at least partially automate the detection process; it would be preferable to have a UAV send back just images of a potential target to the controller for verification, rather than expect the controller to continually stare at a screen for long periods.

The detection of people from small UAVs poses several challenges, including

- Payload - Small research UAVs can typically carry less than a kilogram of load depending on their flight-time. This has to account for cameras, lenses, mounts, and computing and communication equipment. Although a mission-ready UAV

might manage more, hard decisions will have to be made regarding the use of payload. In particular, heavy zoom lenses are unlikely to be viable.

- Camera vibration - Small vehicles are particularly vulnerable to gusts of wind, and there is normally little weight allowance available to provide complete dampening of mechanical vibrations within the UAV. UAVs specifically designed with aerial photography in mind have gimbal mounts which allow the camera to continue pointing in the same direction while compensating for changes in vehicle orientation. Other UAVs use video stabilisation technology which compares consecutive video frames and shifts each frame in order to provide stable output.

- Processing power - Although techniques such as Graphics Processing Unit (GPU) programming may be very effective for image processing, the overall computing power available on a UAV will be limited.

- Infrared picture quality - 'Traditional' quality IR cameras are heavy, particularly if they use active cooling systems. We instead use a micro-bolometer camera, which weighs just 80g including a single lens (but not including all of the required processing electronics). Such cameras are solid state, and the price may be expected to fall[1], but the most expensive (c. $8K) model currently available outside of the military provides an image resolution of only $640 \times 480$.

- Weather conditions - To be fit for purpose a UAV-based solution should be able to deal with at least some variety of weather. The RKM 8X Surveillor by RotorKonzept (Figure 1.1) was designed specifically for surveillance applications, and is capable of withstanding rain, snowfall and wind speeds of up to 40km/h. It is equipped with a thermal camera and a visible light camera, and has a flight time of up to 20 minutes.

[1]FLIR are now marketing a small microbolometer unit with a resolution of $100 \times 80$ for under $500.

Figure 1.1: RKM 8X Surveillor UAV by RotorKonzept [107].

## 1.1 Proposed Approach

This thesis addresses the problem of initially detecting and then tracking a person from frame to frame in video footage captured from aerial views. The primary motivation is to automate the process rather than have a human controller stare at a screen for long periods of time. To solve this problem, a novel algorithm for detection and tracking is proposed: *use the infrared modality to 'guide' the search in the visible band image*. Visible light cameras provide high resolution over three colour channels, whereas the raw output of an infrared camera is monochromatic (indicating apparent surface temperature) and generally of poorer resolution. However, high resolution visible light images can take considerable time to process; most image processing algorithms take time that is at least linear in the image size. We therefore consider an approach in which the infrared images are used to find interesting regions in the image, namely bright (warm) regions of an appropriate size. Such regions can be found quickly, and these are then used to centre a search window for a visual detection algorithm to scan for evidence of a person.

For tracking the person – that is, localising the person in every frame without having to run the more computationally involved detection process each frame – this thesis proposes an algorithm which combines information from both modalities and uses the last known location of the person to predict where they will next appear. The use of a Kalman filter along with the combination of two modalities makes it possible to continue tracking a person in situations where existing visual tracking

algorithms usually fail.

The principal contributions of the thesis can be summarised as follows:

- Integrated infrared/visible light detection of people, extending the prior works of [22, 51, 109] to use a part-based detection algorithm to find people in the visible band image.

- HeatTrack – a novel algorithm for combining measurements from two modalities in order to localise a person in subsequent frames following the initial detection, extending the above-mentioned works which run detection algorithms independently in each frame.

- Motion compensation – an adaptation of the original HeatTrack algorithm to work in the case of a moving camera. This is based on previous algorithms to stabilise videos using interframe 2D homographies to approximate camera motion [63, 80, 84].

## 1.2 Progression of Research

The initial research of this DPhil focused on (i) the problem of creating a camera rig to capture image sequences from two cameras, where the images are as closely synchronised in time as possible, (ii) finding a way to map an infrared image to its corresponding visible light image where the two cameras have different resolutions and fields of view, and (iii) developing an image processing pipeline to highlight the hotspots and then analyse the corresponding regions in the visible light image. Previous work on mapping infrared to visible light images has used global 2D homographies to provide a one to one mapping between pixels in the two images [22, 51, 89, 109]. This thesis presents an improved method of matching the two modalities through the use of unsigned HOG descriptors. Whilst previous work has used monolithic templates to classify humans in aerial imagery, this thesis explores the use of a part-based detection algorithm aimed at detecting articulated people. The use of a part-based detection algorithm is possible because the footage captured from small robotic he-

licopters is closer to the subject than the higher altitude fixed-wing UAVs used in previous work. Hence, the person appears in greater detail in the image.

Initial results of this thesis are presented in [46, 47]. This initial work used temporal consistency as a way to identify people 'tracklets' in a video sequence, using the idea of repeated detections as a confidence measure for the presence of a person. The focus then shifted to developing a robust method to track people – which does not require running the full detection pipeline in each frame – and then focused on making the system work with a moving camera and under occlusion.

From the outset, the intention was to put the handheld camera rig and processor onboard a real UAV and to fly it. Putting the camera rig onto a real UAV posed a significant challenge due to its limited payload. The onboard processor would need to be light enough to fly without sacrificing too much flight time, but still powerful enough to capture from two cameras simultaneously without sacrificing frame rate[2]. Due to expense of the infrared camera and the flight risks associated with current UAV technology, it was decided to conduct experiments using a handheld camera rig capturing multi-modal video sequences in a set of aerial locations around Oxford.

## 1.3 Thesis Structure

A review of relevant literature is presented in Chapter 2, including a history of object detection, followed by a review of specific algorithms for detecting and tracking people, and concluding with a review of how these methods have been applied to infrared imagery and multi-modal setups. The thesis is structured as follows:

- Chapter 3 - *Multi-Modal People Detection* - describes one of the main contributions of this thesis: automatic detection of people in a combined infrared/visual setup, and details various improvements made to the basic method and insights gained from the initial results.

[2]Frame rate refers to the number of video frames captured per second. High frame rates are especially desirable in this application because the camera is expected to be moving and vibrating; if the frame rate is too low this can mean that the imaged scene from two consecutive video frames can look very different.

- Chapter 4 - *Multi-Modal People Tracking* - describes the second contribution of the thesis: tracking people under occlusion by combining information from two modalities, and presents detailed analysis of the characteristic behaviour of HeatTrack in various different scenarios. Chapters 3 and 4 build upon previous research into detecting people from aerial multi-modal setups, in particular that of [22, 51, 109].

- Chapter 5 - *Discussion* - discusses various characteristics of the approach and examines in greater detail its strengths and weaknesses, and Chapter 6 - *Conclusions* - concludes the thesis.

# Chapter 2

# Literature Review

This chapter summarises the state of the art in automatic people detection in both visible light and thermal imagery. It begins with a primer on methods in object detection and the machine learning methods necessary for successful object detection. The current state of the art in people detection and tracking is presented, along with a review of how these methods – originally designed for visible light imagery – have been applied to thermal imagery.

## 2.1 An Overview of Object Detection

The science behind most of today's object detection algorithms can be traced back to the seminal work of David Marr [83], whose research focused on understanding the human visual system. In his 1982 book *Vision*, published posthumously, Marr divided the vision problem into three levels of analysis, starting with the most basic representation and building progressively more complex representations until a full 3D model of the world is constructed. The levels are:

**Primal sketch,** which is concerned with image 'tokens', such as edges and lines, derived from brightness changes. The spatial organisation of these image tokens reflects the structure of the visible surfaces in a scene.

**2.5D sketch,** which represents the orientation and depth of the visible surfaces as well as discontinuities. This is similar in concept to an artist highlighting or shading areas of a scene to give a sense of depth. The '.5' in '2.5D' is intended to describe the idea that, in reality, we do not see all of our surroundings, and so we make assumptions about the structure of objects.

**3D model,** which is a full understanding of the scene. This is intended to describe shapes using a hierarchy of basic features.

Although some original Marr theories about biological vision have been challenged by more recent findings, his basic framework of sequentially building increasingly rich representations, culminating in a 3D representation of the scene, is the normal practice in object recognition today. This section describes the main techniques used in object recognition which came out of research done in the early eighties, beginning with (i) the notion of how an object is to be **represented** in an image, (ii) how its appearance may be **learned** by a computer and (iii) how the object will be **detected** in a new image.

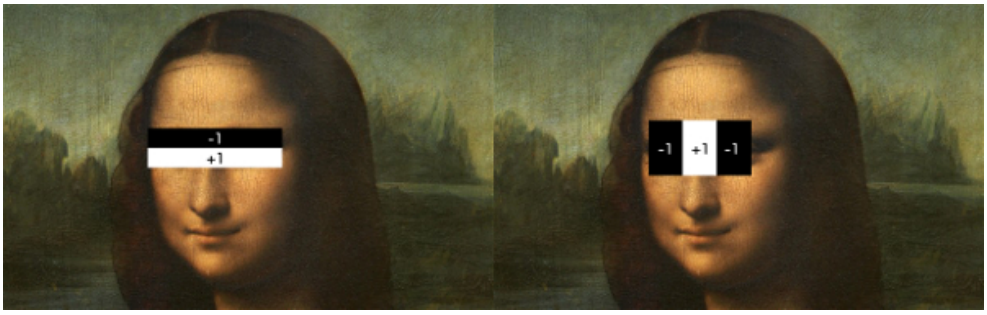



Figure 2.1: An example of an edge feature (left) and line feature (right) and these overlaid on actual images to show where these filters would produce a high response. Values inside the features are the matrix weights.

### 2.1.1 Feature Representations

Local image features are the basic building blocks of any object detection algorithm. Methods for representing an object range from using low-level features like raw pixel values to higher-level representations which combine lower-level features to encode knowledge that is difficult to encode using a vector of pixel values. Image features are derived from brightness (intensity) changes. An image can be viewed as a digital signal where colour changes represent the image frequency. Low frequencies correspond to important structural components of the image, whereas high frequencies correspond to details of the image which are less important and possibly noise. Sharp intensity changes in only one direction signify an edge; sharp changes in two directions simultaneously signify a corner. For a computer, the process of identifying edges or corners in an image is a matter of computing derivatives over the whole image, or, in mathematical terms, *convolving* the image with a derivative filter.

Convolution filters are a way to process images for certain features, where the feature filter is a matrix of values[1]. Convolution is a mathematical operation in which the result (a pixel value) is the weighted sum of the values of neighbouring pixels, where the weights are defined in the filter. The convolution process can be viewed as sliding the filter (kernel) over the whole image and multiplying the values under the sliding window by the weights in the kernel. The resulting image is the

[1]Convolution is central to image processing. This and many other fundamental methods in image processing/computer vision can be found in textbooks such as [115].

filtered image – the 'response' of the image to the filter.

Figure 2.1 shows some simple image features, and these overlaid on actual images to show where these filters would produce a high response. Much of object detection/recognition is based on the concept of simple features like these and how they might be combined in order to form a complete understanding of an object or scene.

#### 2.1.1.1 Scale-invariant Key Points

Sometimes the goal is just to match features across two images, and this necessitates finding image locations which are visually salient. A pixel inside a uniform region would be difficult to match with another image because there will be multiple potential matches, but a corner is likely to be matched because it is unique. The Scale-Invariant Feature Transform (SIFT) proposed in [81] is an algorithm to detect and describe local features in an image which are stable and repeatable under changes in scale, illumination etc. This algorithm shares many features with neuron responses in primate vision.

Figure 2.2 shows the process by which SIFT key points are found in an image. An image is repeatedly blurred and down-sampled by a factor of 2. Adjacent images at a particular scale (size) are subtracted to give difference-of-Gaussian images (DoG) which show edges in the image. Key points are found by comparing each pixel in a DoG image to its eight neighbours at the same scale and nine corresponding neighbouring pixels in each of the neighbouring scales. If the pixel value is the maximum or minimum among all compared pixels, it is selected as a candidate key point.

The key point can be made rotation-invariant by assigning it a dominant orientation based on local image properties. Its descriptor (a vector of values) is then represented relative to this dominant orientation and therefore made invariant to image rotation. An orientation histogram with 36 bins (or in general $N$ bins) is populated with the gradient orientations[2] of image points sampled in the neighbourhood

[2]Gradients are image derivatives e.g. differences between adjacent pixels. Given an image $I$, $I_x$ is the gradient in the $x$ direction. $I_y$ is the gradient in the $y$ direction. The gradient orientation at a particular pixel is $atan(\frac{I_y}{I_x})$ and the gradient magnitude is $\sqrt{I_x^2 + I_y^2}$.

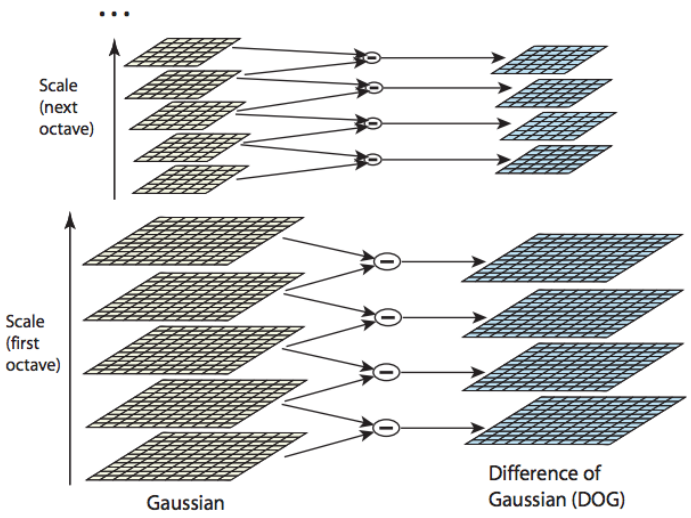



Figure 2.2: Scale-invariant Feature Transform. For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of blurred scale space images shown on the left. Adjacent blurred images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated. Image taken from [81].

of the key point. The bin corresponding to the peak of the histogram is taken to be the dominant orientation for this key point. The values in the descriptor are then rotated according to the dominant orientation.

A Gaussian weighting function with a standard deviation of half the width of the descriptor window is used to assign a weight to the gradient magnitude of each pixel in the descriptor window. The purpose of doing this is to avoid sudden changes in the descriptor with small changes in the position of the window, and to give less weight to values which are further away from the centre of the feature.

In the original work of [81], the key point descriptor contains the values of orientation histograms for $4 \times 4$ pixel patches. These histograms are computed from magnitude and orientation values of samples in a 16 x 16 region around the keypoint such that each histogram contains samples from a 4 x 4 subregion of the original neighbourhood region. The best results in [81] were obtained with an array of such histograms, each with 8 orientation bins, giving a 128 element feature vector for each key point. Finally, the vector is normalised to unit length to reduce the effect of changes in illumination.

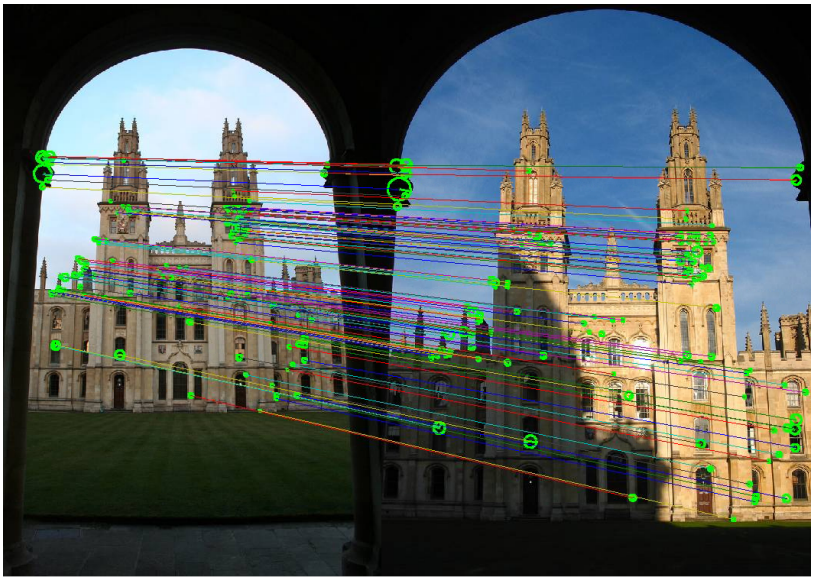


Figure 2.3: An example of SIFT features found in one image, and these matched to SIFT features found in another image of the same scene.

Once computed, the descriptor can be used to match a feature in one image to another image using some similarity measure such as Euclidean distance (see Figure 2.3), or it can be used as part of a more general object recognition algorithm to learn the appearance of an object. The original SIFT algorithm has high computational complexity because it involves computing a full image pyramid to achieve scale invariance. Each descriptor has a dimensionality of 128 to make it stable and repeatable, and since the number of extracted SIFT descriptors tends to be very high per image, the typically quadratic matching process is slow. There are several more computationally-efficient feature descriptors inspired by SIFT, the most popular of which is SURF (Speeded Up Robust Features), which boasts faster computation time due to its use of Haar wavelets and integral images [14], important concepts which are explained later in Section 2.1.3.1. Other variants of SIFT include ORB [108], BRISK [78] and FREAK [3].

One key application of key point descriptors and matchers such as the ones just described is in computing the mapping between two images. A homography is a projective mapping which maps one image plane to the other. The relationship between pixel location $(u\ v\ 1)^\intercal$ in the first image and pixel location $(x\ y\ 1)^\intercal$ in the

second image is given by

$$c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2.1}$$

where

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ x_7 & h_8 & h_9 \end{bmatrix} \tag{2.2}$$

and $c$ is any non-zero constant. There are various algorithms for efficiently estimating the homography matrix between two images; here we describe the Direct Linear Transform (DLT) algorithm [60]. Dividing the first row of equation 2.1 by the third row and the second row by the third row gives the following two equations:

$$-h_1x - h_2y - h_3 + (h_7x + h_8y + h_9)u = 0 \tag{2.3}$$

$$-h_4x - h_5y - h_6 + (h_7x + h_8y + h_9)v = 0 \tag{2.4}$$

Equations 2.3 and 2.4 can be written in matrix form as:

$$\mathbf{Ah} = 0 \tag{2.5}$$

where

$$\mathbf{A} = \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & ux & uy & u \\ 0 & 0 & 0 & -x & -y & -1 & vx & vy & v \end{bmatrix} \tag{2.6}$$

and

$$\mathbf{h} = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 & h_8 & h_9 \end{bmatrix}^T \tag{2.7}$$

Each point correspondence provides two equations; therefore four correspondences are sufficient to solve for the eight degrees of freedom of $H$, with the restriction that no three points can be collinear. The 1D null space of $\mathbf{A}$ is the solution space for $\mathbf{h}$.

In matching key points from one image to the other, there are often outliers, and these can result in incorrect homography computation. RANSAC (Random Sample Consensus) [44] is the most commonly used robust estimation method for homographies. For a number of iterations, a random sample of four point correspondences is

selected and a homography $H$ is computed from these correspondences. Every other correspondence is then classified as an inlier or outlier depending on its concurrence with $H$. After the final iteration, the iteration which contained the largest number of inliers is selected and $H$ can then be recomputed from all of the correspondences that were consider as inliers in that iteration.

### 2.1.2 Learning the Appearance of an Object

Object recognition relies on learning ways to find differences between different classes, and consequently, machine learning methods have exploded in the field of object recognition. Given an image, we would like to know whether it contains a particular class of object or not. This can be viewed as a comparison between two probabilities: (i) the probability that, given an image, it contains a particular object and (ii) the probability that, given an image, it does not contain the object. The question is, given an image, how probable is it that the object is in the image? Using Bayes' rule, the probability of an object being in an image can be expanded to:

$$\underbrace{P(\text{object}|\text{image})}_{\text{posterior ratio}} = \underbrace{P(\text{image}|\text{object})}_{\text{likelihood ratio}} \cdot \underbrace{P(\text{object})}_{\text{prior ratio}} \tag{2.8}$$

There are two main machine learning techniques that are commonly used to solve the problem of object categorisation – *discriminative* learning and *generative* learning. Discriminative learning tends to look at the posterior directly and tries to find boundaries between two (or $n$) classes. It learns a model that is able to distinguish between positive and negative training examples. (In the context of object recognition, positive training examples are images of the object; negative training examples do not contain instances of the object). Generative learning, on the other hand, is concerned with the likelihood and the prior. It tries to model the whole joint distribution of image features $P(x_1, x_2, \ldots, x_n)$. It asks the question, 'given an underlying model for the appearance of a person, how likely is it that this new image came from that underlying distribution?'. For this to work well, the joint distribution should accurately capture the relationship between the variables comprising an object. This kind of model typically learns by maximising the likelihood of positive training data.

From a computational point of view, generative models are considered to be more complex, since they produce a full probability density over all the image features. Discriminative models merely try to find a mapping from input variables to an output variable, with the goal of discriminating between classes, rather than modelling a full representation of a class. There is therefore no redundancy in the model representation. Many algorithms combine the two paradigms e.g. [6, 20, 43]. In order to capture the appearance of highly articulated people, a combination is often required; the range of possible appearances of a person is very wide, making it difficult to use one decision boundary to separate positive and negative examples.

In addition to deciding whether to opt for a discriminative or generative model, one also has to decide the level of *supervision* in learning. Supervision in this context refers to how much information is given to the learning algorithm about the nature of the training examples. For example, when building a person classifier, we can supply images of people on their own, or supply images of people in a natural scene and 'tell' the algorithm where to find the person by manually annotating them with a bounding box. For objects composed of a number of typical parts, we could supply the location of the most important features, such as the face, arms, legs etc. Some part-based algorithms, as will be seen in a later section, provide semi-supervised learning, as in the case of Felzenszwalb et al. [39], which supplies the bounding box of the person in the training image but leaves it up to the algorithm to learn where the individual parts are – by maximising the likelihood of certain part configurations.

Learning the appearance of objects in images is an inherently imbalanced problem: a single image tends to contain a large number of non-object patches and relatively few pixels belonging to the object, and this can lead to overfitting on the training set. This is because the classifier could end up learning the noise in images rather than the general nature of the object category. It is therefore important to have a training set containing many images of the objects, with as many sources of variation as possible. The next section gives a brief overview of some of the most successful machine learning algorithms used in the context of learning the appearance of an object in an image.

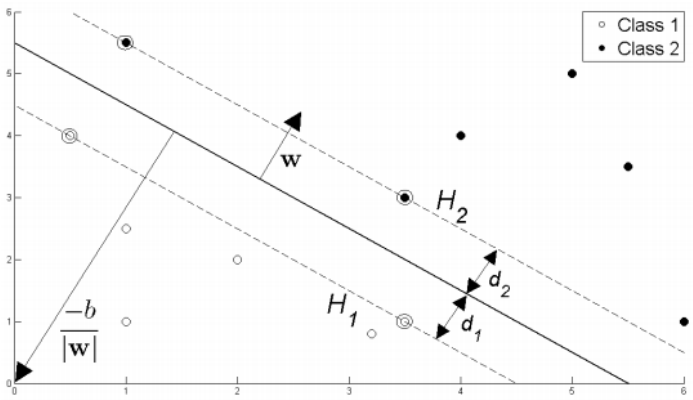



Figure 2.4: Hyperplane through two linearly separable classes.

#### 2.1.2.1 Support Vector Machines

Support vector machines (SVMs) [28] take a set of feature vectors as input and attempt to learn a decision boundary which best separates positive and negative training examples. Each feature vector $\mathbf{x}_i$ has dimensionality $D$ and is in one of two classes $y_i = -1$ (negative example) or $+1$ (positive example). For the basic algorithm the data must be linearly separable, meaning it is possible to separate the two classes with a line in the case of 2-dimensional vectors, and a hyperplane in the case of $n$-dimensional vectors. This hyperplane can be described by $\mathbf{w} \cdot \mathbf{x} + b = 0$ where $\mathbf{w}$ is the normal to the hyperplane and $\frac{b}{||\mathbf{w}||}$ is the perpendicular distance from the hyperplane to the origin. Figure 2.4 illustrates this idea. The support vectors are the training examples which are closest to the separating hyperplane and the goal of SVMs is to move this hyperplane in such a way as to maximise the distance between the closest members of both classes.

Implementing a SVM becomes a task of selecting the variables $\mathbf{w}$ and $b$ so that the training data can be described by:

$$x_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } y_i = +1 \tag{2.9}$$

$$x_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } y_i = -1 \tag{2.10}$$

These equations can be combined into:

$$y_i(x_i \cdot w + b) - 1 \geq 0 \quad \forall i \tag{2.11}$$

Vector geometry shows that the margin is equal to $\frac{1}{||\mathbf{w}||}$ and maximising it subject to the constraint in 2.11 is equivalent to finding:

$$\min ||\mathbf{w}|| \text{ such that } y_i(x_i \cdot \mathbf{w} + b) - 1 \geq 0 \quad \forall i \tag{2.12}$$

Minimising $||\mathbf{w}||$ is equivalent to minimising $\frac{1}{2}||\mathbf{w}||^2$ and the use of this term makes it possible to perform Quadratic Programming (QP) optimisation to solve it. To summarise the algorithm briefly, (and referring the reader to [28] for a full explanation), the use of Lagrange multipliers, differentiation and QP allows the variables $\mathbf{w}$ and $b$ to be computed.

In the case of object detection, assuming $\mathbf{w}$ and $b$ have been learned from a set of training images, detecting an instance of the object in a new image patch involves extracting the feature vector $\mathbf{x}$ from the patch, computing $\mathbf{w} \cdot \mathbf{x} + b$ and if this is greater than 1, reporting a detection.

#### 2.1.2.2 Ensemble Methods

The goal of ensemble methods is to combine the predictions of several base classifiers in order to improve robustness over a single estimator. Bootstrap aggregating, also called bagging [23], is a machine learning meta-algorithm designed to improve the accuracy of classifiers. Given a training set $D$ of size $n$, bagging generates $m$ new training sets $D_i$, by sampling from $D$ uniformly and with replacement. At training time, $m$ models are learned from the $m$ training sets and combined by voting to create a single output. The key idea of bagging is to decrease the variance in prediction. The algorithm was originally designed for classification and is usually applied to decision tree models, but it can be used with any type of model for classification or regression.

Random forests [24] use a similar idea to bagging, but in addition to selecting random subsets of the training set, they select random subsets of features from which a single feature is used as a split variable during construction of a decision tree. As a result, the individual decision trees which form the forest are more independent. Averaging over trees can substantially reduce instability that might otherwise result, and because the individual trees are independent, the gains in predictive performance

from averaging over a large number of trees can be dramatic.

Boosting [49] is a technique for combining multiple 'weak' classifiers to produce a committee whose performance can be significantly better than that of any of the individual classifiers. Boosting can give impressive results even if the weak classifiers have a performance that is only marginally better than random, hence the name 'weak learners'. Boosting defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \dots \tag{2.13}$$

where $F(x)$ is the final, strong classifier, $\alpha_i$ are the weights and $f_i(x)$ are the weak classifiers.

The weak classifiers are trained in sequence, and each classifier is trained using a weighted form of the data set, in which the weighting coefficient associated with each data point depends on the performance of the previous classifiers on that data point. AdaBoost [50], short for Adaptive Boosting, is a widely used form of boosting. It is adaptive in that subsequent classifiers are adapted in favour of instances misclassified by previous classifiers, in order to focus on examples which have previously been misclassified. This is done by assigning greater weights to the points that were misclassified by one of the earlier classifiers, when training the next classifier. Once all classifiers have been trained, their predictions are combined through a weighted majority vote. By iteratively combining weak classifiers in this way, the training error quickly converges to zero.

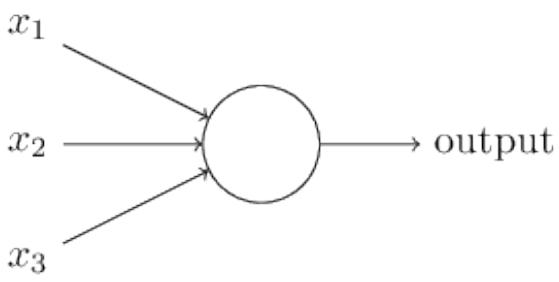
Used in object recognition, boosting provides an efficient algorithm for sparse feature selection. Often a goal of object recognition is that the final classifier depend on only a few features, since it will be quicker to evaluate. In addition, a classifier which depends on a few features will be more likely to generalise well. Tieu and Viola [122] used boosting to select a small set of discriminatory sparse features out of a possible set of over 45,000 in order to track a particular class of object. For each feature, the weak learner computes a Gaussian model for the positives and negatives and returns the feature for which the two class Gaussian model is most effective. No single feature can perform the classification task with perfect accuracy.

Subsequent weak learners are called upon to focus on the remaining errors in the way described above. In the context of a database of 3000 images, their algorithm ran for 20 iterations, yielding a final classifier which depended on only 20 features. Inspired by the work of Tieu and Viola, Viola and Jones [125] used boosting to build an extremely efficient face detector dependent on a small set of simple features which look at the difference in intensity between adjacent regions. In that algorithm, the weak learners are decision tree stumps which make a prediction based on the value of a single input feature. Decision tree stumps are a specialisation of general decision trees, where the goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. The major advantage of decision trees over other machine learning classifiers is that they are simple to understand and interpret.

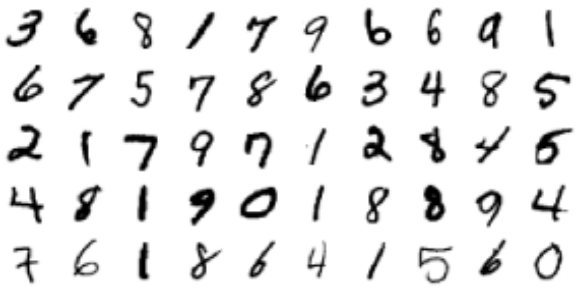
#### 2.1.2.3 Deep Learning / Convolutional Neural Networks

Perceptrons are a type of artificial neuron developed in the fifties and sixties by Rosenblatt [105] based on earlier work by McCulloch and Pitts [86]. Perceptrons work by taking several input values as shown in Figure 2.5a and producing a single output. Each of the inputs is weighted according to how much they are expected to affect the output, and the output is determined by whether the weighted sum of inputs is greater than a certain threshold. Whilst a simple network like this is limited in what it can represent, if more layers are added to the network it can represent any function, and this is what makes artificial neural networks so powerful. Given a set of labelled training examples, back-propagation is used to incrementally adjust the weights so as to minimise the difference between the desired output and the actual output of the network.

Neural networks were one of the first classifiers to achieve human-competitive performance on the famous MNIST handwritten digits problem [75] (Figure 2.5b). The networks used for this task were Convolutional Neural Networks (CNN), a special kind of multi-layer neural network but with a slightly different architecture. CNNs have special types of hidden layers: convolutional layers, which involve convolving




(a) The simple perceptron model of [86]



(b) A sample of handwritten digits from the MNIST database [75]

the input with a filter specified by the weights going into the layer, and subsampling layers, which involve averaging or maxing the input to the layer in order to gain some degree of translational invariance. Even with the convolutional and subsampling layers, there are still many parameters which can be specified. For example, each convolutional layer can have many filters, and this makes the network much more powerful; it enables the network to automatically learn features, as opposed to HOG or Haar features (described in Sections 2.1.3.3 and 2.1.3.1) which require manual hand-crafting. The result is a highly complex network which is susceptible to overfitting – and therefore many training examples are required.

Adding many such layers increases the classification power of the network, but in the late nineties when they were first applied to MNIST, computing power was such that only small networks could be used, and training of the network took weeks. Without efficient computing resources and methods to prevent overfitting with more and more layers, these CNNs could not be made deeper. With the increase in computing power, in particular GPU technology, it has been possible to train larger, deeper CNNs. In the ImageNet Large-Scale Visual Recognition Challenge of 2012[3], the deep neural network of [69] achieved error rates considerably lower than the previous state-of-the-art on a test set of 150,000 images. Figure 2.5 shows examples of the features which were automatically learned by the network. All major entries in the 2013 and 2014 ImageNet ILSVRC competition were based on deep learning approaches.

[3]ImageNet is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000 categories. Starting in 2010, as part of the Pascal Visual Object Challenge, an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been

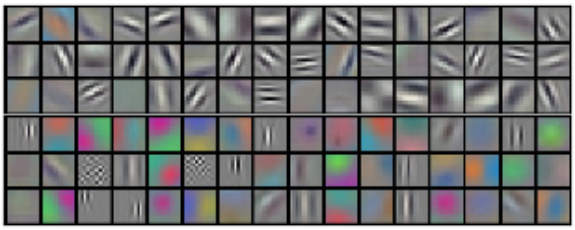


Figure 2.5: A selection of convolutional kernels automatically learned by the convolutional neural network of [69] for the 2012 ImageNet Large-Scale Visual Recognition Challenge. Image taken from [69].

### 2.1.3 Detecting an Instance of an Object in an Image

This section looks at low-level features used to represent the appearance of an object. A combination of such low-level features would form a feature vector which would be used as input to any of the machine learning algorithms described in Section 2.1.2 in order to learn the general appearance of an object.

#### 2.1.3.1 Haar features

Haar features are another low-level feature which gained popularity in the early 2000s following the first real-time face detection algorithm, that of Viola and Jones [125]. Haar-like features owe their name to their similarity with Haar wavelets, first proposed by Alfred Haar in 1909 as part of his doctoral thesis [54]. These consider adjacent rectangular regions at a specific location in a detection window, summing up the pixel intensities in each region and calculating the difference between these sums. The difference is then used to categorise subsections of an image.

Viola and Jones used three kinds of features: two-rectangle, three-rectangle and four-rectangle; these are shown in Figure 2.6a. A two-rectangle feature is computed as the difference between the sum of pixels within two rectangular regions. The value of a three-rectangle feature is the sum within two outside rectangles subtracted from the sum in a centre rectangle. A four-rectangle feature is computed as the difference between the sum of pixels in the rectangles that make up the main and off diagonals. Viola and Jones use the notion of an *integral image* as an intermediate representation

held. ILSVRC uses a subset of ImageNet with roughly 1000 images in each of 1000 categories.

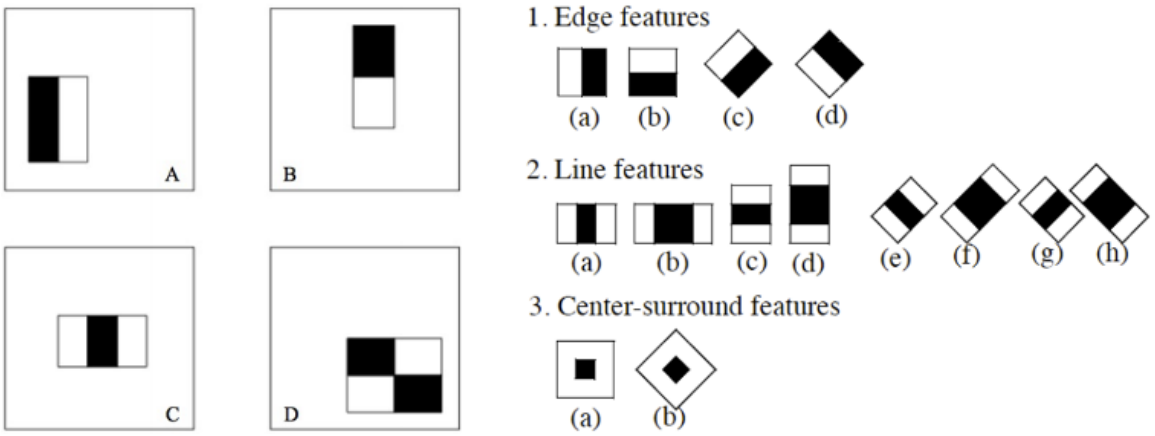



Figure 2.6: (a) Original Haar feature set proposed by [125]. (b) Extended feature set proposed by [79].

of the image in order to compute Haar features. They calculate the integral image at $(x, y)$ as:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y'), \tag{2.14}$$

where $ii(x, y)$ denotes the integral image and $i(x, y)$ is the original image – the integral image is simply the sum of pixels above and left of $x$, $y$, and including $x$, $y$. Using the recurrence equations:

$$s(x, y) = s(x, y - 1) + i(x, y) \tag{2.15}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \tag{2.16}$$

where $s(x, y)$ denotes the cumulative row sum, $s(x, -1) = 0$ and $ii(-1, y) = 0$, the integral image can be calculated in one pass over the whole image. A rectangular sum can be computed in four array accesses. Therefore the difference between two rectangular sums can be calculated in eight array accesses. The two-rectangular features shown in Figure 2.6a involve adjacent rectangular sums; hence, they can be computed in six array accesses, eight in the case of three-rectangle features, and nine for four-rectangle features.

A good object recognition algorithm incorporates methods to deal with rotation, translation and scale invariance, since objects can appear anywhere in the image and

at different sizes. The key advantage of the SIFT algorithm discussed in Section 2.1.1.1 is its rotational invariance – it computes the dominant orientation of a key point and then rotates the descriptor according to this orientation. This is in contrast to Haar features which are not rotationally invariant, since they are simply computing image differences. Scale invariance is generally dealt with using either of two approaches:

- Construct a 'pyramid' of images by repeatedly resizing an image into smaller dimensions. A detector of fixed size is then scanned across each of these images. While quite straightforward, computation of the pyramid takes a long time, so it is less than optimal for use in real-time applications.

- Leave the image at one size. Convolve with a series of different sized kernels to look for pattern matches. This is the approach used by Viola and Jones and is more computationally efficient than the pyramid scheme. It is now the de facto method used in feature extraction.

The computational advantage of the integral image technique is obvious when compared with the pyramid approach. Implemented on typical hardware when Viola and Jones first published the algorithm in 2001, it would have been very difficult to compute a pyramid at 15 frames per second. In contrast, the integral image approach enabled features to be computed at any scale and location in only a few operations, and made the face detector run much faster than contemporary face detectors. Since the publication of Viola and Jones' original face detection algorithm, there have been additions to the original feature set to increase the expressive power of the model, notably by [79], shown in Figure 2.6b.

#### 2.1.3.2 Local Binary Patterns

Local Binary Patterns [91] are a type of feature used for capturing local texture in an image. Let $T$ be a local neighbourhood of a grayscale image:

$$T = t(g_c, g_0, \ldots, g_{P-1}) \tag{2.17}$$

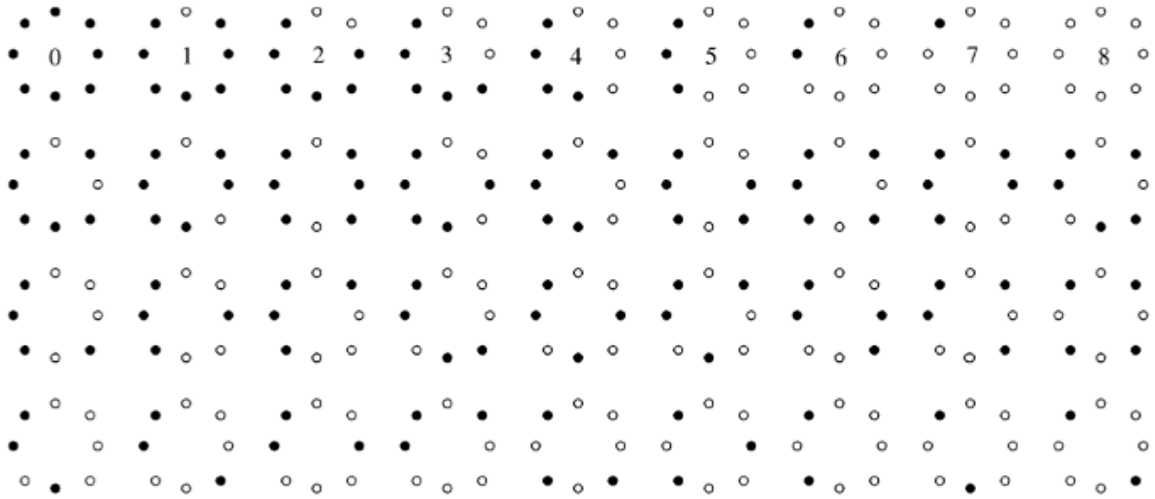



Figure 2.7: Local Binary Patterns. The 36 unique rotation invariant binary patterns that can occur in the circularly symmetric neighbourhood of a pixel. Black and white circles correspond to bit values of 0 and 1 in the 8-bit output of the operator. The first row contains the nine uniform patterns and the numbers inside them correspond to their unique label. Image taken from [91].

where $g_c$ corresponds to the grey value of the centre pixel of the neighbourhood, and the other $g_p(p = 0, 1, \ldots, P-1)$ correspond to the grey values of $P$ equally spaced pixels on a circle of radius $R$ that form a circularly symmetric neighbour set. The method involves subtracting the grey value of the centre pixel from each of the neighbouring pixels. For uniform regions, the differences are 0 in all directions; on an edge, the operator records the highest difference in the gradient direction and 0 values along the edge. Figure 2.7 shows the various possible patterns for a LBP operator with a $P = 8$. Each possible pattern can be represented in binary by recording a '1' where difference $s(g_p - g_c)$ is greater than 0, and a '0' otherwise. A local binary pattern is called uniform if the binary pattern contains at most two bitwise transitions from 0 to 1 or vice versa when the bit pattern is traversed circularly. It was found that uniform patterns account for nearly 90% of all patterns when using the LBP operator with $P = 8$ and $R = 1$. Such is the prevalence of uniform patterns in texture analysis that in the computation of the LBP labels, each uniform pattern has its own binary label, while all the non-uniform patterns are grouped together and labeled with a single label. The final texture feature for an image patch is the histogram of

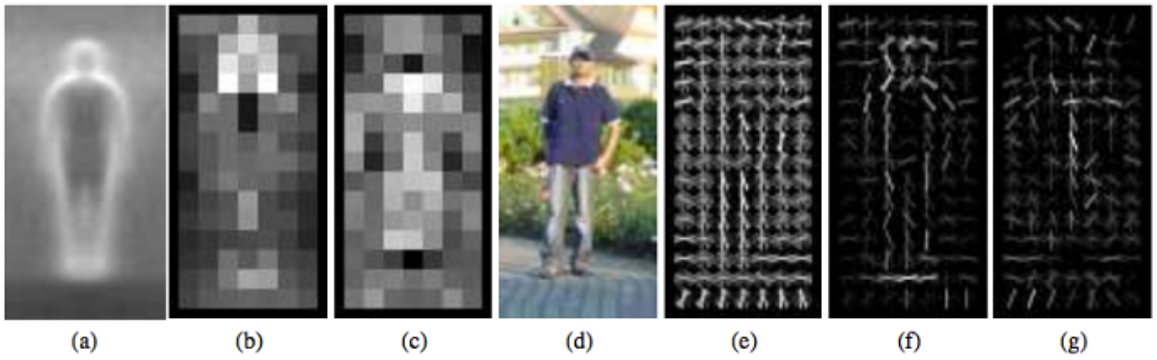



Figure 2.8: (a) Average gradient over training examples (whiter pixels correspond to stronger gradients). This shows that the HOG detector cues mainly on silhouette contours, especially the head, shoulders and feet. (b) Each 'pixel' shows the maximum positive SVM weight in the block centred on the pixel. (c) Likewise for the negative SVM weights. (d) A test image of a person. (e) The computed HOG descriptor of the image. (f,g) The descriptor weighted by its positives and negative SVM weights. Taken from [29].

pattern labels accumulated over a texture sample. Multi-resolution analysis can be done by changing $P$ and $R$ and then combining the information provided by multiple operators of varying $P$ and $R$. Once computed over an image patch, LBP patterns can be used as input to some discriminative machine learning algorithm such as support vector machines (Section 2.1.2.1) or AdaBoost (Section 2.1.2.2) in order to learn the appearance of an object.

The main advantage of LBPs is their computational simplicity, because the patterns can be computed with only a few comparisons in a small neighbourhood and a lookup table. Due to its fast computation time, and the fact that it is invariant to changes in illumination, LBP has been successfully used for many different image analysis tasks, such as face recognition [2], motion modelling [61] and medical image analysis [88].

#### 2.1.3.3 Histograms of Oriented Gradients

The Histograms of Oriented Gradients (HOG) algorithm [29] is an object detection algorithm inspired by SIFT which provided significant performance gains for object detection when it was first published in 2005. The main idea of HOG is that local object appearance and shape within an image can be described by the distribution

of edge directions. In that algorithm, histograms of image gradients are calculated for a group of cells (blocks of pixels) and normalised in a local and overlapping block scheme. These values are concatenated to form a single descriptor of an image patch. Given a set of training images, the HOG descriptors are fed into some supervised learning algorithm such as a SVM in order to learn the decision boundary between object and non-objects.

Figure 2.8 shows the HOG representation. Figure 2.8(b) shows the SVM weights learned on the positive training examples. Figures 2.8(a) and (f) show that the most important cells are the ones which typically contain major human contours such as head, shoulders and feet. The learned weights indicate that these parts have the most influence in the discrimination decision. This shows that the HOG detector cues mainly on the overall shape/silhouette of a person rather than their internal features.

HOG has a couple of advantages over other descriptor methods. Since it operates on localised cells the method is invariant to geometric and photometric transformations (except for rotation), since such changes would only appear in larger spatial regions. Also, coarse spatial sampling, fine orientation sampling, and photometric normalisation allows the individual body movement of pedestrians to be ignored as long as they maintain a roughly upright position.

## 2.2 People Detection Algorithms

Automatic people detection has received a huge amount of attention in the last decades in areas ranging from surveillance to self-driving cars to tracking players in a football game. The detection task is made especially difficult due to wide variation in pose along with differences in clothing, lighting, backgrounds etc. In the object recognition literature, algorithms for detecting people have been broadly categorised as either monolithic or part-based, the former concerned with representing the entire appearance of the body in a single descriptor, the latter dealing with individual parts and modelling the spatial relations between those parts.

### 2.2.1 Monolithic models

Monolithic models use a single descriptor to encode the appearance of an object. Detection is performed by scanning an image with an object classifier at multiple positions and scales and making a decision as to the presence of an object. These work well when the person is fully visible and in a similar pose to what the classifier was trained on. It does not work well when the person is highly articulated or partially occluded.

One of the first pedestrian detection systems was proposed by [96], who applied support vector machines to an over-complete dictionary of Haar wavelets. This idea was extended by Viola and Jones [126] in 2005, who used Haar-like wavelets with a boosted cascade of decision tree classifiers. The key contributions of that paper were the use of integral images for fast feature computation combined with a cascade of classifiers – each of increasing complexity – to enable real-time processing of images. These ideas continue to serve as the foundation for the best performing detectors today.

More recently, histograms of oriented gradients [29] provided significant performance gains for pedestrian detection. HOG is still considered the best performing of the various texture descriptors. Whilst no single feature has been shown to outperform HOG in pedestrian detection, it has been shown that combining features can lead to an improvement. The pedestrian detection survey of [131] showed how a combination of HOG, Haar-like features, shapelets and shape context outperformed any single feature. Similarly, Wu and Nevatia [132] combined HOG, edgelet and covariance features and came to the same conclusion. Wang et al. [129] combined a texture descriptor based on local binary patterns with HOG. [93] added colour information and implicit segmentation and found that segmentation cues outperformed HOG on its own. In an extension of Viola and Jones' original pedestrian detection algorithm, [126] integrated Haar features with difference images in order to accurately detect moving pedestrians.

### 2.2.2 Part-based models

Given the wide variation in human posture, a monolithic detector is too simplistic to work for anything other than upright pedestrians. Part-based models give the flexibility required to deal with highly varying body poses. Many of the state of the art part-based detectors are built on the pictorial structures framework [41, 45]. Here, an object is represented as a flexible configuration of parts, where one such configuration is denoted by $L = \{l_0, \ldots, l_N\}$, with $l_i$ denoting the location of part $i$. The posterior over part configurations $L$ given image evidence $E$ is calculated using Bayes' rule: $p(L|E) \propto p(L)p(E|L)$ and this allows the learning problem to be solved by maximum likelihood methods.

The Deformable Parts Model (DPM) of Felzenszwalb et al. [39] emerged in 2009 as a very powerful algorithm for detecting articulated people. At a high level, DPM can be characterised by (i) the combination of strong low-level features based on HOG to represent each part, (ii) efficient matching algorithms for matching a part-based model to an image, (iii) discriminative learning with latent (hidden) variables. The Felzenszwalb detector uses a star-shaped part-based model defined by a root filter (analogous to HOG) plus a set of part filters and associated deformation models. The score of the model at a particular position and scale is the score of the root filter plus the sum over parts of the maximum, over placements of that part, of the part filter score on its location minus a deformation cost measuring the deviation of the part from its ideal location relative to the root. The score for both root and parts is obtained by convolving a sub-window of a feature pyramid with a HOG filter learned specifically for the part. Figure 2.9 shows a star model for the person model learned by Felzenszwalb et al.

DPM is an example of semi-supervised learning whereby only the bounding box of the person is given in the training images; the locations of the parts must be learned during training. Felzenszwalb et al. use a formulation of MI-SVM [5] called Latent SVM. In a latent SVM, each example is scored by a function of the form

$$f_\beta(x) = \max_{z \in Z(x)} \beta \cdot \Phi(x, z). \tag{2.18}$$

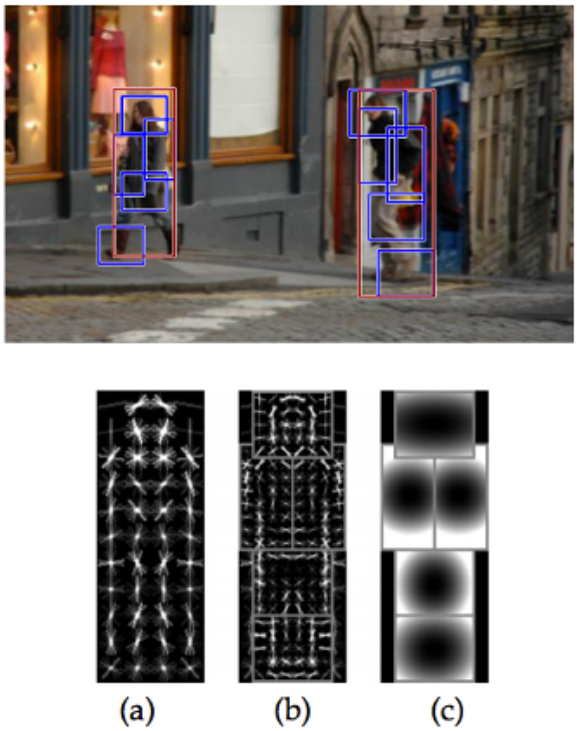



Figure 2.9: The Felzenszwalb model is defined by a coarse root filter (a), with several higher resolution part filters (b) and a spatial model for the location of each part relative to the root (c). The filters specify weights for HOG features. The visualisation of the spatial model shows the cost of placing the part at different locations relative to the root. Taken from [39].

$\beta$ is a vector of model parameters, $z$ are latent values (locations of the parts) and $\Phi(x, z)$ is a feature vector extracted from the image using the HOG feature representation. In the case of Felzenszwalb model, $\beta$ is is the concatenation of the root filter, the part filters and the deformation cost weights, $z$ is a specification of the object configuration, and $\Phi(x, z)$ is a concatenation of sub-windows from a feature pyramid and part deformation features.

The goal is to learn $\beta$, the vector of model parameters during training. This is trained from a set of training images annotated with a bounding box by minimising the objective function

$$L_D(\beta) = \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{n} \max(0, 1 - y_i f_\beta(x_i)), \tag{2.19}$$

A latent SVM leads to a non-convex optimisation problem, which means that there is a risk the learning algorithm may converge to a local minimum. Felzenszwalb et al.

found that the problem can be made convex once latent information is supplied for each of the positive training examples. Hence, the part configuration $Z$ is estimated for each positive training example as part of training. In practice this a two step procedure: (i) set an initial value for $\beta$, fix it, and optimise the objective function over fixed $\beta$ for different values of $Z$, (ii) optimise $\beta$ by solving the convex optimisation problem defined by using the $Z$ found in the previous step.

Figure 2.10 shows an overview of the detection stage, given a new image. The feature maps are extracted for root and part filters (at twice the resolution of the root filter for multi-resolution processing). This produces a set of response maps which are then transformed to take into account the deformation costs learned by the models. Matching parts to an image is typically the bottleneck of part-based detection algorithms. One of the key contributions of the Felzenszwalb algorithm is that the matching process can be computed in $O(nk)$ time (where $n$ is the number of parts and $k$ is the total number of locations in the feature pyramid) by using dynamic programming and the generalised distance transform of [40]. High scoring locations in the output response map yield detections.

The Felzenszwalb algorithm takes nearly 2.5 seconds to run over a $1280\times960$ image on standard hardware. Since the original publication [39], various methods have been proposed to speed up the algorithm without sacrificing detection accuracy. The method of [38] added cascade classifiers to quickly prune most of the hypotheses without losing detection accuracy. This allowed an increase of detection speed equal to 20 times that of the original detection system. The method of [98], which could be seen as complementary to the cascade detection algorithm, follows a coarse-to-fine approximation: detection is attempted first on a coarse, low-resolution model, and if a positive response is received, further detection tests are performed on gradually higher resolution models and part filters. In fact, the most expensive and time-consuming operations of the deformable parts model approach – HOG computation, convolution and distance transforms – are all parallelisable. The GPU implementation of [112] has succeeded in doing real-time detection of 20 different object classes by sharing part features across classes.

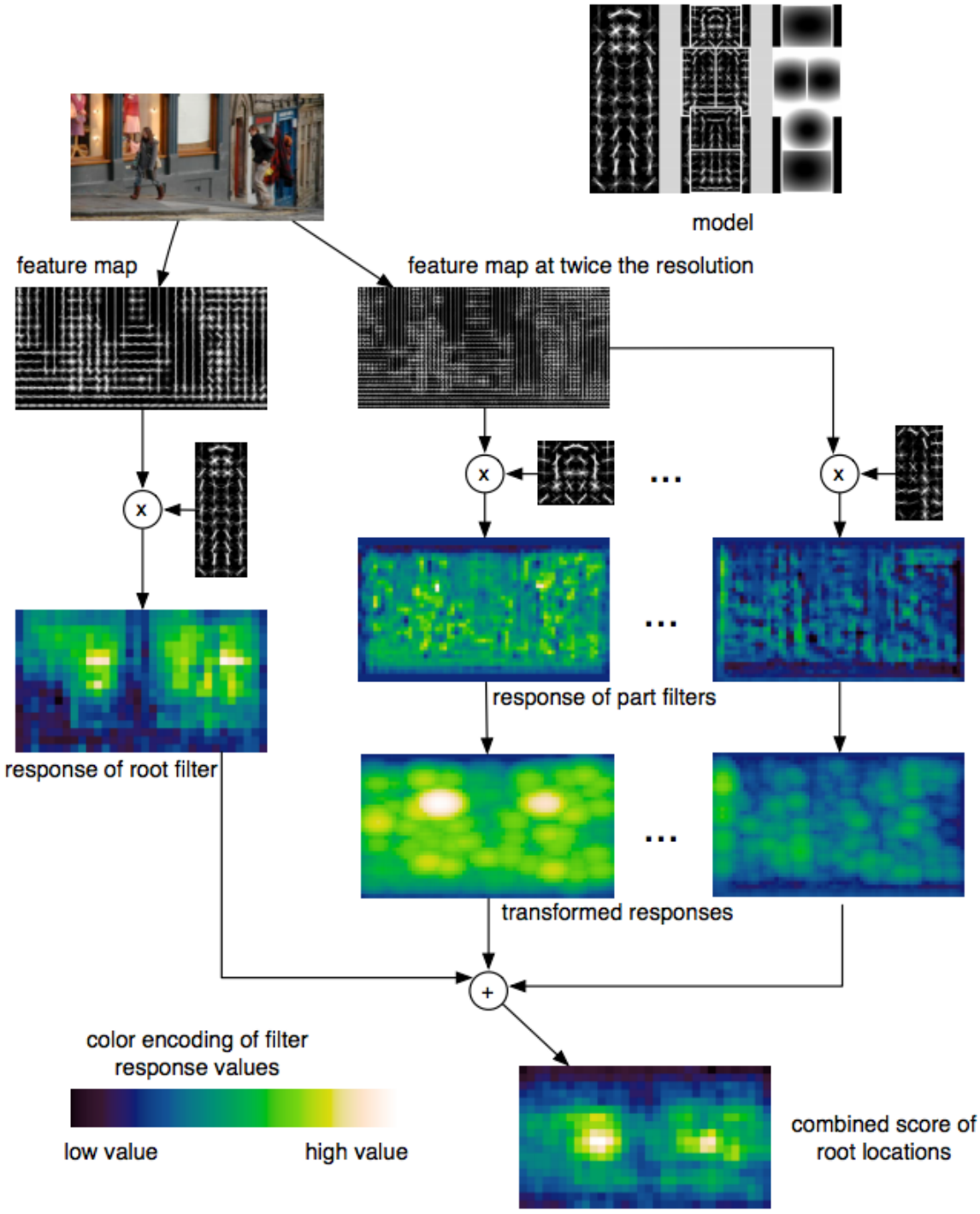



Figure 2.10: The Felzenszwalb detector in action. Taken from [39].

Several recent detectors are also based on the pictorial structures framework e.g. [6, 38, 42, 124, 136]. In general, the main differences between these algorithms lie in (i) the level of supervision during training (whether parts are explicitly labeled in

the training set, for example), (ii) the complexity of the graphical model – tree or fully connected – with fully connected graphical models resulting in more complex inference, and (iii) the underlying appearance model for the parts e.g. HOG, LBP, Haar features etc.

The Implicit Shape Model (ISM) [76, 77] is an alternative to the pictorial structures framework. This method represents a person as a collection of image patches that are prototypical of a person. These prototypical image patches (called codebook entries) are typically found by clustering training image key points in appearance space. Each cluster then forms a codebook entry and the descriptor represents the object in terms of whether or not it contains a these codebook entries. In addition, a probability distribution is defined which specifies where each codebook entry may be found relative to the centre. Each patch has its own associated detector e.g. HOG. Given a new image, the individual part detectors are run over the image, and the matched patches vote for the centroid of the object based on the statistics learned from a set of exemplar images. The advantage of this approach is that only a small number of training images are typically required.

Poselets [21] use an alternative definition of a part that does not necessarily correspond to a body part, recognising the fact that anatomical parts are not necessarily the most salient features for visual recognition. The training set is manually annotated[4] with a set number of key points such as left ear, right elbow etc. Image patches are selected randomly from the training set. For each patch, similar patches in all the other images are found, based on the configuration of key points in the patches. The patches are clustered, and each cluster forms a poselet. The appearance of each poselet is learned using HOG with support vector machines. Context is exploited by storing the location of each poselet relative to other poselets. At the detection stage, a HOG detector is run over the image for each poselet, and a person is reported if there is a set of poselet activations which is consistent with the spatial layout statistics of the training set.

[4]As with many training sets for object recognition, the authors make use of crowd-sourcing initiatives such as Amazon Mechanical Turk in order to annotate several millions of images.

## 2.3 Tracking Algorithms

Whereas detection is the process of determining whether an image contains an object of interest, usually done by searching the entire image, tracking is the process of estimating the location of the object in every frame of a video using knowledge of its past appearance. In general, the following things need to be considered when designing a tracking algorithm:

- **How the object is represented** - methods vary from using holistic templates i.e. raw pixel values [4, 55, 85], to using subspace-based (dimensionality-reduced) trackers which are better able to deal with appearance changes [18, 106]. One of the most basic tracking algorithms is MeanShift [25], which is an algorithm for finding the maxima of a density function given a histogram sampled from that function. Given a histogram of the object to be tracked, in each iteration of the algorithm, the mean of the data points under the current search window is found, and the window is then translated so that it is centred on that point. Mean shift refers to this translation and this is repeated until convergence. More recent methods have used sparse templates to handle appearance changes [13, 127, 139]. In addition to templates, other visual features are used in tracking algorithms, such as colour histograms [26], HOG [116] or Haar-like features [53]. A recent paradigm is tracking-by-detection, in which a classifier is learned online to discriminate the object from the background, using positive and negative training examples cropped from the image. Various learning methods have been used such as SVM [9], structured output SVM [58], boosting [10, 53], and multiple-instance boosting [12]. To make trackers more robust to pose variation and partial occlusion, an object can be represented by a set of parts where each part has its own representative feature vector [1, 73]. Other trackers integrate multiple representation schemes for robustness [74, 113].

- **How search is done in subsequent frames** - This can be deterministic or stochastic. Gradient descent can be used to find the maximal scoring image patch

according to some similarity function, but this is susceptible to finding local minima [26, 82]. To reduce the search space further, motion estimation is done using a linear model, which can be described using a Kalman filter, but a moving camera complicates this as there is the added motion of the camera which is separate from the motion of the object. Dense sampling can be used as an alternative to gradient descent at the expense of increased computational load [11, 53, 58]. Stochastic methods such as particle filters have also been proposed because they are insensitive to local minima [64, 106].

- **How the object model is updated, if at all** - In order to cope with appearance changes some trackers update the appearance model of an object as tracking goes on. For example, Matthews et al. [85] update the template model for Lucas-Kanade [82] by combining the template from the current image with the reference histogram extracted from the first frame when tracking begin. Some algorithms store a history of the last $n$ appearances of an object, comparing a new hypothesis with the history to see if it is similar. To avoid computational overload, older patches are forgotten as new ones come in. In tracking-by-detection algorithms, special care must be taken when choosing new positive and negative examples for the classifier, as if the wrong samples are chosen the tracker can start to drift beyond recovery.

### 2.3.1 Appearance-based Tracking

The current state of the art in general object tracking lies in tracking-by-detection, that is, learning a classifier online in order to detect the object in every frame and use this to update the classifier. This section describes three state of the art tracking-by-detection algorithms which we make use of in this thesis.

TLD (Tracking Learning Detection) [66] does as the title suggests; it decomposes the tracking problem into the separate tasks of tracking, learning and detection. *Tracking* estimates the object motion between consecutive frames under the assumption that the frame-to-frame motion is limited and the object is visible. *Detection* scans the entire frame attempting to localise all instances of past observed appear-

ances. *Learning* observes the performance of both tracker and detector, estimating detector errors and generating training examples to avoid these errors in the future.

MILTrack [12] uses multiple instance learning instead of traditional supervised learning in an attempt to deal with the problem of incorrectly labelling training examples which can cause the tracker to drift. The method recognises the difficulty in taking the current tracker region as the source for positive samples and the surrounding as the source for negative samples as the target may not completely fill the bounding box or cover some of the background.

Struck: Structured Output Tracking with Kernels [58] attempts to correct the deficiencies of other tracking-by-detection trackers by taking out the intermediate classification step which can result in incorrectly labelled training examples. Rather than learn a binary classifier as TLD and MILTrack do, Struck learns a prediction function to directly estimate the object transformation between frames. The output space is then the space of all transformations instead of a binary 0/1 answer. Training proceeds in an online manner as in the case of the other trackers.

In a 2013 extensive survey on the state of the art in object tracking [134], a conclusion reached was that background information is critical for successful tracking, because this helps differentiate the object from the background. Secondly, local models which divide the tracking window into patches are better than global ones which try to capture the whole appearance in one feature vector. Sparse local models which consider key points as opposed to denser descriptions perform better under partial occlusion. Finally, a motion model is necessary especially when the motion of the object is large or abrupt. Good location prediction based on the motion model can reduce the search space, thereby improving efficiency.

### 2.3.2 Motion Models for Tracking

Recursive state estimators can be used to model the motion of an object and predict its future location based on past measurements. State estimators deal with the problem of estimating quantities from sensor output which are not directly observable but which can be inferred. In object tracking, for example, the location of the object is

not directly measurable – instead, the location needs to be inferred from the result of searching for image patches which match some appearance model for the object. This section describes Kalman filters and particle filters, the two most commonly used recursive state estimation algorithms used in object tracking.

#### 2.3.2.1 Kalman Filter

The Kalman filter [67, 114] was invented as a technique for filtering and prediction in linear Gaussian systems – that is, the state transition function from one state to the next is linear. It is a probabilistic state estimation algorithm, computing belief distributions over all possible world states. The filter represents beliefs by the moments parameterisation; at time $t$, the belief is represented by the mean $\mu_t$ and the covariance $\Sigma_t$. The state transition probability $p(x_t|x_{t-1}, u_t)$ is a linear function of its arguments with added Gaussian noise. This is expressed as

$$x_t = Ax_{t-1} + B_t u_t + \epsilon_t, \tag{2.20}$$

where $x_t$ and $x_{t-1}$ are state vectors, $u_t$ is the control vector at time $t$ and $\epsilon_t$ is the process noise. The distribution of $\epsilon_t$ is a multivariate Gaussian with zero mean and covariance $R_t$. The measurement probability $p(z_t|x_t)$ must also be linear in its arguments with added Gaussian noise:

$$z_t = C_t x_t + \delta_t, \tag{2.21}$$

where $\delta_t$ is the measurement noise. The distribution of $\delta_t$ is a multivariate Gaussian with zero mean and covariance $Q_t$. The initial belief must be normally distributed, denoted by $\mu_0$ and $\Sigma_0$. These properties are sufficient to ensure that the posterior is always a Gaussian.

The Kalman filter algorithm is shown in Algorithm 3. The input to the filter is the belief at time $t-1$, represented by $\mu_{t-1}$ and $\Sigma_{t-1}$. The output is the belief at time $t$, represented by $\mu_t$ and $\Sigma_t$. The algorithm consists of two main steps, prediction and update. In the prediction step (lines 2 and 3), the predicted belief is calculated by incorporating the control $u_t$, but before incorporating the measurement

**Algorithm 1** Kalman filter

1: **procedure** KALMANFILTER($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)
2: $\quad \bar{\mu}_t \leftarrow A_t \mu_{t-1} + B_t u_t$
3: $\quad \bar{\Sigma}_t \leftarrow A_t \Sigma_{t-1} A_t^T + R_t$
4: $\quad K_t \leftarrow \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
5: $\quad \mu_t \leftarrow \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
6: $\quad \Sigma_t \leftarrow (I - K_t C_t) \bar{\Sigma}_t$
7: $\quad$ **return** $\mu_t, \Sigma_t$

$z_t$. In the update step (lines 5 and 6), the measurement $z_t$ is incorporated. The variable $K_t$ (line 4) is called the Kalman gain. This specifies the degree to which the measurement is incorporated into the new state estimate. In line 5, the mean is adjusted in proportion to the Kalman gain $K_t$ and the deviation of the actual measurement $z_t$ from the expected measurement $C_t \bar{\mu}_t$. Finally, the new covariance of the posterior is calculated taking into account the information gain resulting from the measurement.

The Kalman filter is computationally efficient. The complexity of a matrix inversion is approximately $O(d^{2.4})$ for a matrix of size $d \times d$. Each iteration of the Kalman filter is lower bounded by $O(k^{2.4})$, where $k$ is the dimension of the measurement vector $z_t$. It is also at least $O(n^2)$, where $n$ is the dimension of the state vector, due to the multiplication in line 6. In many applications, the measurement space has a much lower dimension than the state space, and the update is dominated by the $O(n^2)$ operations.

#### 2.3.2.2 Extended Kalman Filter

The assumption that the state is a linear function of the previous state and that observations are linear functions of state are crucial for the correctness of the Kalman filter. Unfortunately, the state and measurement transitions are often not linear functions. The extended Kalman filter (EKF) [130] relaxes this assumption. In the EKF the assumption is that the state transition probability and the measurement

probabilities are non-linear functions $g$ and $h$ respectively:

$$x_t = g(u_t, x_{t-1}) + \epsilon_t \tag{2.22}$$

$$z_t = h(x_t) + \delta_t \tag{2.23}$$

The function $g$ replaces the matrices $A_t$ and $B_t$ in Equation 2.20 and $h$ replaces the matrix $C_t$ in Equation 2.21. However, with arbitrary functions $g$ and $h$, the belief is no longer a Gaussian. In fact, updating the belief is usually impossible for non-linear functions and there is no closed-form solution.

The EKF therefore calculates a Gaussian approximation to the true belief. The goal of the EKF is to efficiently estimate the mean and covariance rather than compute an exact posterior. The key idea of the EKF is linearisation. Linearisation approximates the non-linear function $g$ by a linear function that is tangent to $g$ at the mean of the Gaussian. Similarly, the EKF approximates $h$ by a linear function which is tangent to $h$, thereby retaining the Gaussian nature of the posterior.

EKFs use a method called Taylor expansion to linearise non-linear functions. Taylor expansion calculates a linear approximation to a function $g$ from the value of $g$ and its slope. $g$ is approximated by its value at $\mu_{t-1}$ and the linear extrapolation is achieved by a term which is proportional to the gradient of $g$ at $\mu_{t-1}$ and $u_t$.

As is the case with the linear Kalman filter, each update requires time $O(k^{2.4}+n^2)$, where $k$ is the dimension of the measurement vector $z_t$ and $n$ is the dimension of the state vector $x_t$. A major limitation of the EKF is the approximation of the state and measurements using Taylor expansions. In many estimation problems, state transitions and measurements are non-linear. If the non-linear functions are approximately linear at the time of the estimate, then the EKF approximation may be a good one. The less certain the state estimate is, the wider is its Gaussian belief, and the more the filter is affected by non-linearities in the state and measurement functions.

**Algorithm 2** Particle filter

1: **procedure** PARTICLEFILTER($\chi_{t-1}, u_t, z_t$)
2: $\bar{\chi}_t = \chi_t = 0$
3: **for** $m = 1$ to $M$ **do**
4: `sample` $x_t^{[m]} \sim p(x_t|u_t, x_{t-1}^{[m]})$
5: $w_t^{[m]} = p(z_t|x_t^{[m]})$
6: $\bar{\chi}_t = \bar{\chi}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7: **for** $m = 1$ to $M$ **do**
8: `draw` $i$ `with probability` $\propto w_t^{[i]}$
9: `add` $x_t^{[i]}$ `to` $\chi_t$
10: **return** $\chi_t$

#### 2.3.2.3 Particle Filter

Kalman filters represent the belief about a state with a uni-modal distribution. Sometimes one may wish to represent the state estimate by a multi-modal distribution, and this is where particle filters [52] are useful. The particle filter is a non-parametric implementation of the Bayes filter [121]. The main idea of the particle filter is to represent the posterior by a set of random state samples drawn from this posterior. This representation is approximate but it is non-parametric and can therefore represent a much broader space of distributions than a Gaussian.

In particle filters, the samples of a posterior distribution are called particles, denoted as:

$$\chi_t := x_t^{[1]}, x_t^{[2]}, \ldots, x_t^{[n]} \tag{2.24}$$

Each particle is an instantiation of the state at time $t$. $M$ denotes the number of particles in the particle set $\chi_t$. Ideally, the likelihood for a state hypothesis $x_t$ to be included in the particle set would be proportional to its posterior:

$$x_t^{[m]} \sim p(x_t|z_{1:t}, u_{1:t}) \tag{2.25}$$

Like the Kalman filter, the particle filter algorithm constructs its current belief recursively from the previous belief.

The particle filter algorithm is shown in Algorithm 2. Line 4 generates a hypothetical state $x_t^{[m]}$ for time $t$ based on the previous particle and control $u_t$. This

step involves sampling from the state transition distribution $p(x_t|u_t, x_{t-1})$. The set of particles obtained after $M$ iterations is the representation of the belief before incorporating the measurement. In line 5, an importance factor is calculated, denoted $w_t^{[m]}$. This importance is the probability of the measurement $z_t$ under the particle $x_t^{[m]}$. Interpreting $w_t^{[m]}$ as a weight, the set of weighted particles represents the updated filter posterior.

The most important step of the particle filter is the resampling trick in lines 7 to 9. The algorithm draws with replacement $M$ particles from the set $\bar{\chi}_t$, with the probability of a particle being drawn given by its importance weights. Resampling converts a particle set into another particle set of the same size, but by incorporating the importance weights into the resampling, the distribution of the particles changes. After resampling, the particles should be distributed according to the posterior. The particles not included in the new set tend to be the ones with low posterior probability.

A source of error in the particle filter algorithm relates to the variation inherent in random sampling. For example, if samples are drawn from a Gaussian random variable, the mean and variance of the sample will differ from the true mean and variance. To deal with this variance, a high number of samples must be used to give better approximations with less variability. Another limitation of particle filters is that repeated sampling in the absence of sensor observations can lead to a loss of diversity. This could be alleviated by re-sampling only when the variance of the weights of the particles is high.

In summary, particle filters are easy to implement and can model any probability distribution – discrete or continuous – unlike Kalman filters. There are various ways to reduce the error in particle filters, with methods ranging from reducing the variance of the estimate that arises from the randomness of the algorithm, to techniques for adapting the number of particles in accordance with the complexity of the posterior. In general, the particle filter algorithm has higher complexity than a Kalman filter, but the complexity is difficult to estimate given the wide variation in numbers of particles used and the many different techniques used for efficient resampling.

## 2.4 Multi-modal Methods in Computer Vision

Although most of the literature on people detection has focused on the visible light modality, there has been increased interest in using infrared due to the fact that the infrared signature of a person tends to be invariant to clothing and lighting changes. Infrared cameras record electromagnetic radiation emitted by objects in a scene as a thermal image whose pixel value is indicative of the heat being emitted by an object. Given the lack of detailed colour information in infrared images, approaches to people detection in infrared are based on looking at contrast and gradients rather than colour.

### 2.4.1 Detecting People in Thermal Imagery

HOG features in conjunction with support vector machines have been used to detect pedestrians in infrared with considerable success [17, 56, 87, 133]. Others have used shape context descriptors with AdaBoost [128] or used SURF features to learn an implicit shape model [65] or visual codebook [72]. More recently, [101] have experimented with the use of sparse representation-based classification to classify humans in infrared images in combination with a dense representation like HOG.

A common misconception about infrared is that the person always shows up as a hot spot in the image. Often this is not the case, especially on a hot day when the temperature of a person is similar to their background. This makes gradient-based descriptors less robust and has has motivated the use of phase congruency maps to detect human silhouettes [92]. Phase congruency reflects the behaviour of the image in the frequency domain. Features extracted using phase information from the Fourier transform of an image are more robust to changes in illumination and contrast than are gradient-based features.

The lack of detailed colour and texture information can result in a high rate of false positives if the classifier is evaluated over the entire image; this has motivated the use of background subtraction [30] or finding Maximally Stable Extremal Regions [119] to first identify candidate regions possibly containing people, and then applying

a classifier to those regions. As can be seen, a lot of the techniques originally applied to the visible light modality have also been used with success in infrared.

### 2.4.2 Cross-modal Image Registration

Given the complementary nature of the visible light and infrared modalities, it makes sense to try to fuse images from both modalities. Visible light imagery provides rich colour and texture information while the infrared provides thermal information which is much less susceptible to lighting changes and can therefore be used to disambiguate what is in the visible light image. The main difficulty with combining modalities is determining correspondences between the two modalities. Features which appear in one modality may not appear in the other, or there may be contrast reversal. A small amount of research has been done into finding ways of matching points in one image to the other.

Mutual information (MI) measures the statistical co-occurence of pixel-wise information such as textures and patterns inside two images. [70] finds correspondences by maximising the mutual information between infrared and visible light images. [37] showed that MI outperforms traditional matching methods such as normalised cross-correlation-based methods. The foreground regions are extracted in both images; in the visible light image this is done using background subtraction, in the infrared, by intensity thresholding. Matching proceeds by fixing a window in one image, and sliding a correspondence window in the other to find the best match according to mutual information in the form of image grayscale histograms. The drawback of this approach is that it assumes that an accurate foreground segmentation is available in both images, which is not always the case.

Local Self-Similarity (LSS) is another metric proposed for measuring the similarity between two images. LSS was originally proposed for image template matching. While most image descriptors represent colours or gradients, LSS represents the layout/shape of objects inside an image region. It has an advantage over MI in that it can deal better with situations in which there is uniformity in the infrared but texture in the visible light, as long as they have a similar spatial layout. [19, 123] use LSS as

a similarity metric between points in the infrared and the visible light.

HOG descriptors have been shown to be useful in matching gradients between images of different modalities. [99] developed a similarity measure based on dense HOG descriptors using unsigned image gradients to deal with the contrast reversal. The HOG descriptor is based on histograms of oriented gradient responses in a local region around each point of interest. Using such features in combination with a strong match optimisation approach, they were able to compute largely valid but coarse depth maps for a multi-modal pair, with results comparable to visible light only stereo setups.

Although exact feature matching using the methods listed above is required for scene depth recovery, if the scene is far enough away from the camera as to make the varying depths of objects in the scene negligible compared to the observation distance, then a planar homography can capture the mapping between the two images reasonably accurately. This is the approach taken by [31, 51, 89, 90] who use a homography manually computed in advance in order to register the two images.

Most of the research into multi-modal people detection using infrared and visible light cameras focuses on fixed surveillance cameras, and this allows for the use of background subtraction to help identify moving objects in the scene. One such example is [31], who do Gaussian background subtraction in the infrared and then use the regions found in this way to fine-tune the background subtraction in the visible image. In that method the images are already registered. A similar method of background subtraction is proposed in [57], but in that case they use the result to automatically register the two images by matching the foreground silhouettes in a genetic algorithm-based search scheme.

In [89], information from the LUV[5] channels of the visible image is used alongside the corresponding infrared pixel value as part of a non-parametric model which is aimed at detecting moving people. In order to initialise the background model in the

[5]Unlike RGB, the LUV colour space is perceptually uniform – meaning that two colours equally distant in the colour space according to the Euclidean metric are equally distant perceptually. The L, u and v channels correspond roughly to luminance, green-red and blue-yellow.

presence of foreground objects such as people, the infrared image is thresholded at a point far from the mean of the intensity histogram in order to do a rough segmentation of hot objects.

In [71] they manually find the homography between the two images, then extract regions of interest (ROI) through background subtraction, and then look at the intersection of the ROIs found in both images. A set of hard-coded fuzzy rules is used to determine the accuracy of the infrared and visible measurements, and this information is incorporated into a Kalman filter which tracks the detected objects.

In addition to differences in *how* the modalities are fused, a key factor in the success of detection/tracking is *when* the information is fused. [32] does an evaluation of when is the best time to fuse information from the two images. In that system, they perform motion detection using optical flow, with the assumption that moving objects are people and the goal is to track people using a particle filter. One way to combine the information is to interlace the two images and then do motion detection on the combined image. Another way is to do motion detection separately in each image and then combine the motion masks. Alternatively, both images can be processed independently of one another and, in the event of there being a detection in one image but not the other, this information is used to re-initialise a tracker where the detection is lacking. The study concludes that performance is maximised by fusing later on in the tracking process i.e. tracking independently in both modalities and then fusing the results. Fusion too early can result in errors from one modality being propagated through the system. Fusing later in the pipeline gives more control over what information can be used or ignored.

## 2.5 Application to Aerial Imagery

This thesis is concerned with the automatic detection and tracking of people in aerial imagery. One potential application area is on board Unmanned Aerial Vehicles (UAVs). UAVs can be broadly divided into two categories: fixed-wing and rotor-craft (these are shown in Figure 2.11). Fixed-wing aircraft generate lift efficiently by

(a) (b)

Figure 2.11: (a) APM:Plane Fixed-wing UAV [8]. (b) Ascending Technologies Pelican Quadrotor [117].

continuously moving through the air at speed, and this enables extended operating time. However, these constraints require routes to be planned in advance, and the limited manoeuvrability means they are not suitable for some applications, such as hovering in a fixed location. In rotor-craft vehicles, lift is generated by all of the rotors; changes in orientation and hence movement are controlled via the associated motor currents. Advantages of rotor-craft UAVs over fixed-wing aircraft include the ability to hover, operate near ground level and even fly inside buildings. In addition to fixed-wing and rotor-craft, there is a hybrid class of UAV which operates likes a powered glider, requiring forward movement to maintain altitude, but with a slow stall speed. Although not as manoeuvrable as rotor-craft, a long flight time and large payload makes these UAVs good for aerial photography and photogrammetry.

Detecting people from aerial imagery is an important capability for UAVs in search and rescue systems. The problem of detecting people varies considerably depending on how far away the camera is from the person, and this is dependent on the type of UAV used. Fixed-wing aircraft fly at much higher altitudes than rotor-craft ones, allowing them to take in a much wider field of view, but a person occupies only a small part of the image. Such imagery precludes the use of complex appearance-based person detectors. Rotor-craft UAVs tend to fly much closer to the ground (10 - 20 metres) and they can hover; this allows them to get much more detailed video footage, but limited to a small area. There is also the added issue that in aerial views people

tend not be viewed head on but at an oblique angle.

In the system of [118], a fixed-wing UAV equipped with a single visible light camera flies at 400m above ground looking for people, with the assumption that people are moving. This allows them to compute a global transformation between consecutive images and factor this motion estimate out, leaving the independent motion in the scene. These regions are then analysed with a classifier based on texture, gradient and local binary patterns. In [103, 104] they use the metadata provided by a UAV e.g. latitude, longitude, altitude, pitch, roll and yaw, to produce geometric constraints on the shadows cast by humans in order to identify regions of interest in an image, and then classifies these regions using Haar wavelet features and a SVM.

A small amount of work has focused on combining infrared and visible light footage taken from UAVs. In [109] a robotic helicopter flying approximately 50m above ground is used to detect people lying on the ground; they use the infrared to narrow down the search to warm regions and then analyse the corresponding regions in the visible light image with Haar classifiers trained to detect lower, upper and full bodies lying down. Another system which combines two modalities is that of [22, 51], who have a fixed-wing UAV flying at 200m above ground. People detection is done using cascaded Haar classifiers on both thermal and visible images, with secondary confirmation provided by a multivariate Gaussian shape matching technique or an additional classifier. In that method the two images are related by a planar homography, the planar assumption being valid due to the distance of the UAV from the ground.

The above-mentioned systems deal with the case of imagery taken from altitudes so high as to make detailed appearance-based classifiers inapplicable, but only a small amount of work has been done on close-range imagery. One example is [7] who have a small quadrotor UAV flying inside an office environment and they apply a part-based people detection algorithm to detect people lying on the floor. Another work, similar in spirit to the topic of this thesis (though using only infrared images), is that of [100]. They created a dataset of videos taken from aerial views, and trained a rotationally-invariant part-based detector to detect people who are then tracked using a particle

filter. Background subtraction is used to reduce the search space to those regions more likely to contain people.

The work that this thesis presents differs from these previous approaches in two key aspects. Firstly, previous work such as [22, 51] focused on fixed-wing UAVs flying at high altitudes where the cameras have a much wider view over the terrain, and hence, the person occupies only a tiny part of the image. This precludes the use of complex human classifiers to use on the visible light imagery. In contrast, our work focuses on the type of aerial imagery expected to be captured from small robotic helicopters which are more manoeuvrable and can fly closer to the ground. In contrast to [51], which processes both modalities independently and then combines the result into a single confidence map, in this DPhil the aim is to reduce the search space from the very beginning, similar in spirit to [109] – by first processing the lower resolution infrared image and using that to reduce the search space in the visible light.

---

We now conclude the literature review with a summary of the state of the art in each of the areas dealt with in this thesis. The current state of the art in pedestrian detection can broadly be classified into approaches based on DPM (section 2.2.2), deep networks (section 2.1.2.3) and decision forests (section 2.1.2.2). To recap, DPM treats the body as a collection of parts with springs between them representing spatial connections. Methods in this category differ in the amount of supervision during training (whether the parts are explicitly labelled, for example), and how the parts are represented and learned. Examples include [43, 97, 135]. Deep architectures are very large neural networks which have become popular in recent years due to large amounts of training data and increased computing power. Some deep architectures model the entire body as a single template [110]; others use deep architectures to jointly model parts and occlusions [94, 95]. Finally, decision forests use an ensemble of decision tree classifiers to classify an image patch giving better performance than a single classifier alone. Examples include [15, 33, 34, 35, 138]. While these three are based on different learning approaches, their results on benchmark pedestrian

datasets such as the Caltech Pedestrian Detection dataset and ImageNet are on a par with each other. In the pedestrian detection survey of [16], the authors conclude that most of the improvement in pedestrian detection over the last decade can be attributed to the improvement in features alone, and how they are combined. Whilst much progress has been made in dealing with the issues of articulated people and occlusion through the use of part-based detection algorithms, these algorithms fail when the person occupies only a small part of the image. In other words, there is a need to use contextual information in addition to object-specific cues, similar to how the human brain processes visual input.

Most state of the art tracking algorithms such as [12, 58, 62, 66] make use of the tracking-by-detection paradigm, that is, training a classifier online to discriminate the object from the background, using positive and negative training examples cropped from the image. The main difficulty associated with tracking-by-detection algorithms is in choosing correct positive and negative training examples for the classifier – if there are slight inaccuracies in the training examples the tracker will start to drift and ultimately fail. Several different approaches have been proposed to deal with this issue, including the use of Multiple Instance Learning [12] or structured output SVMs [58]. Another approach has been to decompose the tracking problem into tracking, learning and detection [66]. The learner observes the performance of both tracker and detector, estimating detector errors and generating training examples to avoid these errors in the future. A recent survey on the state of the art in tracking found that background information is critical for successful tracking, because this helps differentiate the object from the background. Secondly, local appearance models are better than global ones which try to capture the whole appearance in one feature vector. Sparse local models which consider key points as opposed to denser descriptions perform better under partial occlusion. Finally, a motion model is necessary especially when the motion of the object is large or abrupt. Good location prediction based on the motion model can reduce the search space, thereby improving efficiency.

The problem of detecting people from aerial video footage has received a small amount of attention in the last decade. Most of it has focused on footage taken from

UAVs at high altitudes where the person occupies only a small part of the image. Detecting people in such imagery is prone to many false positives; one approach to deal with this [103, 104], who use the metadata provided by a UAV e.g. latitude, longitude, altitude, pitch, roll and yaw, to produce geometric constraints on the shadows cast by humans in order to identify regions of interest in an image, and then classifies these regions using Haar wavelet features and a SVM. Another work attempts to find people moving in the scene by estimating a global transformation between frames and then factoring this out, leaving the independent movement in the scene. Given the high cost of infrared technology, there has been relatively little work published on multi-modal people detection/tracking in aerial imagery. The main works in this regard are [109], who use the infrared to narrow down the search to the warm parts of the image, and [22, 51], who process both modalities independently with Haar classifiers and then do secondary confirmation with Gaussian shape matching.

As already mentioned, the work that this thesis presents differs from these previous approaches in two key aspects. Firstly, previous work such as [22, 51] focused on fixed-wing UAVs flying at high altitudes where the cameras have a much wider view over the terrain, and hence, the person occupies only a tiny part of the image. This precludes the use of complex human classifiers to use on the visible light imagery. In contrast, our work focuses on the type of aerial imagery expected to be captured from small robotic helicopters which are more manoeuvrable and can fly closer to the ground. In contrast to [51], which processes both modalities independently and then combines the result into a single confidence map, in this DPhil the aim is to reduce the search space from the very beginning, similar in spirit to [109] – by first processing the lower resolution infrared image and using that to reduce the search space in the visible light. In the next two chapters we describe the main contributions of the thesis.

# Chapter 3

# Multi-Modal People Detection

This chapter presents one of the main contributions of the thesis: a method of detecting people in aerial video footage which is an order of magnitude faster than searching the entire image – by using the infrared modality to guide the search in the visible modality. A description of the basic approach is followed by a number of improvements to the algorithm which resulted in more accurate detection of people.

This thesis examines approaches to finding and localising a person in multi-modal imagery recorded from aerial locations. There are two tasks under consideration: the *initial detection* of a person in the scene, and the *tracking* of that person from frame to frame. Detection is the process of determining where a person is in an image, without any prior knowledge of where the person is or whether there is even a person in the scene. This is usually done using a sliding window approach – running a detection algorithm over the image at multiple scales and considering those windows which 'score' above a certain threshold. Without knowledge of where the person is, the whole image must necessarily be searched, and this can be computationally expensive, particularly with more complex part-based detectors[1]. Monolithic detectors such as HOG or Haar wavelets (discussed in Sections 2.1.3.3 and 2.1.3.1) have faster computation times, but are unsuitable for deformable objects.

Given an initial detection of a person, tracking aims to localise that person in subsequent frames – ideally without having to run the original detection algorithm over each new image. Some trackers search only within a small vicinity of where the object was last seen, such as MeanShift [25]; others attempt to gauge the velocity of the object and use this to predict where the object will next appear, as in the case of Kalman or particle filter-based trackers (discussed in Sections 2.3.2.1 and 2.3.2.3). In either approach, the goal is to narrow down the search to a smaller part of the image so as to reduce computation time.

Figure 3.1 shows an overview of the detecting and tracking system developed during this DPhil. This chapter focuses on the initial detection of a person in a scene – looking at how the infrared modality can be used to narrow down the search to more promising parts of the image, and how the resulting regions are then processed in order to detect a person more accurately than the traditional approach of processing the entire visible light image. The next chapter will deal with how the person is then tracked from frame to frame.

[1]The deformable parts model of Felzenszwalb et al. [39], for example, takes nearly 2.5 seconds to run over a 1280×960 image on standard hardware, which is not feasible if the video is to be processed at standard frame rates.

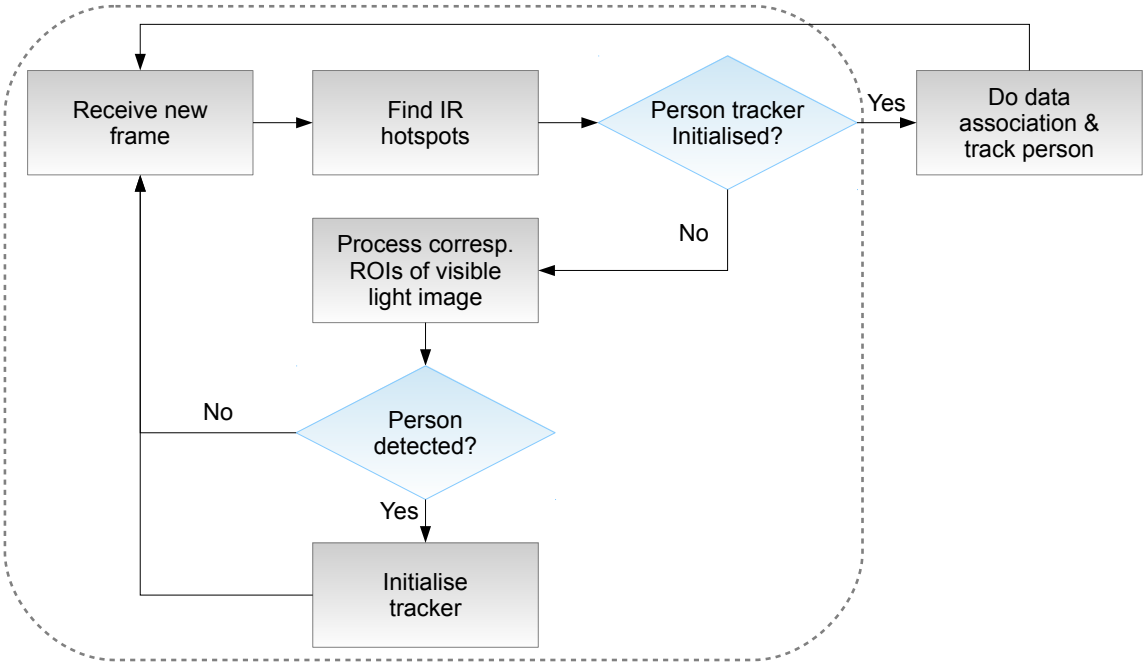



Figure 3.1: Overview of detection and tracking. This chapter is concerned with the area inside the dashed grey line.

## 3.1 The Basic Approach

In contrast to traditional approaches to people detection, which process the entire image, in this approach the infrared modality is used as a way to focus the image search on those regions more likely to contain people. This basic approach makes the assumption that humans are hotter than their surroundings, though as discussed in Section 2.4.1, this assumption can be violated (Sections 3.3.2 discusses an improvement to this approach which does not require this assumption to be made). Given a set of candidate regions in the infrared image, the corresponding regions of interest (ROI) in the visible light image are processed with a part-based person detection algorithm. This idea is depicted in Figure 3.2. The primary motivation behind using the infrared in this way is to reduce the computation time required to search the entire visible light image, and to increase the precision of detection by reducing the amount of false positives.

Although there has been previous research into detecting people from aerial images using a multi-modal setup, the approach described here differs in two aspects. Firstly,


Figure 3.2: Overview of the detection method. (a) The original infrared image. (b) The corresponding visible light image. (c) The warmer parts of the infrared image segmented out, using image thresholding or a gradient-based algorithm. (d) The corresponding regions in the visible light image, found by projecting the bounding boxes in the infrared image into the visible light image using a planar homography or some other method to match pixels in one image to another.

previous work focused on fixed-wing UAVs flying at high altitudes where the cameras have a much wider view over the terrain, and hence, the person occupies only a tiny part of the image. This precludes the use of complex human classifiers to use on the visible light imagery. In contrast, this work focuses on the type of aerial imagery expected to be captured from small robotic helicopters which are more manoeuvrable and can fly closer to the ground. In contrast to [51], which processes both modalities independently and then combines the result into a single confidence map, in this DPhil the aim is to reduce the search space from the very beginning, similar in spirit to [109]

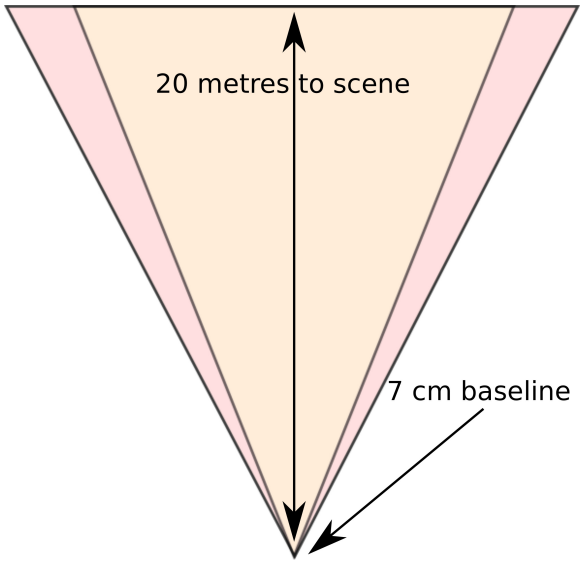



Figure 3.3: Diagram showing the expected horizontal field of view overlap in the two cameras looking at a scene 20 metres away. The camera baseline is 7 cm and the horizontal fields of view of the visible and infrared cameras are 55° and 46°, respectively. The diagram is shown to scale; therefore the 7 cm baseline is negligible compared to the expected distance to the scene.

– by first processing the lower resolution infrared image and using that to reduce the search space in the visible light. Although this method has a clear advantage in terms of computation time, its success is predicated on the person showing up clearly in infrared. If not, then the entire visible light image should be searched.

### 3.1.1 Camera setup & Synchronisation

The camera rig, shown in Figure 3.4, is a handheld camera rig comprising a visible light camera (Point Grey Chameleon 1280×960 with a 8.5mm lens and 2/3" sensor, giving field-of-views of 55°×42°) and an infrared camera (Thermoteknix Miricle Microcam 640×480 with a 18.8mm lens and 1/3" sensor, giving field-of-views of 46°×35°). With the intention of capturing videos approximately 15 - 20 metres away from a scene, the lenses were chosen so that a person would occupy a large enough region in the visible light image as to be detectable by standard detection algorithms. The camera baseline is 7 cm. Hence, there is a significant overlap in the fields of view as shown in Figure 3.3. The spectral response of the infrared camera is 8$\mu$m - 12$\mu$m.


Figure 3.4: Multi-modal camera rig

Spectral response describes the sensitivity of the photosensor to optical radiation of different wavelengths. The infrared camera has a thermal sensitivity of <= 50 milloKelvins (mK). Thermal sensitivity is a measurement of the smallest temperature difference that an infrared camera can detect. With higher sensitivity, a hot object can dominate the image and make the rest of the scene have less contrast. The lower the sensitivity, the more the camera can discern small temperature differences resulting in a more detailed image. The effect of thermal sensitivity will be discussed later in the chapter in the context of detecting humans where there are hotter objects in the scene.

The system was intended to fly onboard a UAV with limited payload, so we chose to use an Atom Fit-PC2 processor [27], shown in Figure 3.5, which is light enough to fly and powerful enough to capture from two image streams simultaneously. At the time of purchase, $640 \times 480$ was the highest available image resolution for the infrared camera, and this was with an analog interface. Because the camera has an analog interface, an analog-to-USB adapter is required in order to connect it to the Atom processor. During the conversion process, a black border is put around the infrared image, so that although the specified image resolution of the camera is $640 \times 480$,

Figure 3.5: Atom Fit-PC2 [27]

after the conversion process this is reduced to 584×474.

Out of the box, the visible light and infrared cameras have frame rates of 15 fps and 25 fps respectively, and synchronising the two cameras temporally is non-trivial. However, temporal synchronisation is very important especially if both camera and person are moving. From the experiments done here it was found that even a couple of frames difference between the two cameras can mean the imaged scene looks very different in both images. The standard approach to deal with this is to have an external trigger in both cameras which would allow an external source to trigger an image to be captured from both cameras on demand. The Point Grey has an external triggering function but the infrared camera has no such capability. In this work an attempt is made to align the images as close in time as possible by having two parallel threads of execution for capturing from both cameras. As the infrared is lower resolution it takes much less time to capture a single image than for the visible light camera. The infrared capture thread therefore waits until the current visible light image has been grabbed from the camera buffer, before both threads proceed to capture the next pair. Although this does not guarantee that the two images are perfectly synchronised given that cameras have different frame rates, for the purposes of this work it was found to be adequate.

### 3.1.2 The Geometry of Multiple Views

If the infrared is to be used to focus the search in the visible light, then the mapping between the two images must be determined. Given that the two cameras have different resolutions and fields of view, a direct pixel to pixel mapping is not applica-

ble. The process of finding the transformation involves selecting corresponding points between the two images and then solving a system of geometric equations which minimise the total reprojection error when points from one image are projected into the other using the estimated transformation.

For the general case of a scene with objects at different depths, the geometry between two views is encapsulated by the Fundamental matrix $\mathbf{F}$ [60]. $\mathbf{F}$ is a $3 \times 3$ matrix of rank 2. If a scene point $X$ is imaged at $x$ in the first image and $x'$ in the second, then the image points satisfy the relation $x'^T \mathbf{F} x = 0$. The matrix $\mathbf{F}$ is independent of scene structure, but can be computed from correspondences of image points alone, without requiring knowledge of the camera's internal parameters. Given the matrix $\mathbf{F}$ and a point $x$ in image 1, the search for its correspondence $x'$ in image 2 is reduced to searching along the epipolar line which satisfies $l' = \mathbf{F}x$.

A special case of the Fundamental matrix is a planar homography (already discussed in Section 2.1.1.1), which applies to scenes which are planar or far enough away that the effects of parallax are negligible. A planar homography $\mathbf{H}$ satisfies the constraint $x' = \mathbf{H}x$. $\mathbf{H}$ is a $3 \times 3$ matrix of rank 3 in which there is a one-to-one point correspondence between the first and second images. In the videos captured for this thesis, the scene is sufficiently far away as to allow the use of a planar homography which is computed manually (once per video) in advance of image processing. The use of a homography to map between two views of an aerial scene has precedence in the earlier works of [51, 109]. A homography is an approximation, but avoids the need to estimate the epipolar geometry which is made especially difficult by the need to match features in different modalities.

### 3.1.3 Registering Images of different modalities

For two images from the same modality, the process of matching features is a well-studied problem. Using a feature extraction algorithm such as SIFT, SURF, FAST or ORB, the salient features are extracted in both images and a descriptor vector is computed around each key point. For each descriptor in image 1, an exhaustive search is done in image 2 to find the nearest neighbour, where the nearest neighbour is defined

as the key point with minimum Euclidean distance from the descriptor vector. Given a set of 2D to 2D point correspondences, the affine transformation between them is computed using the DLT (Direct Linear Transform) [60] and a robust estimator such as RANSAC [44] (both discussed in Section 2.1.1.1).

If the images are from different modalities, determining which points correspond to which can be especially difficult because features may appear differently in different modalities. As the infrared is showing a heat map, it may show up things which are not visible at all in the visible light image, and vice versa. Even if a feature does show up in both images, there can be contrast reversal. Consequently, trying to do a global image registration using gradient-based features such as SIFT or SURF may fail. On the other hand, previous work using unsigned HOG descriptors has made it possible to to a dense registration between infrared / visible light pair, but this comes at the expense of an expensive global optimisation step [99].

Given the difficulty of automatically matching features across modalities, for this work a manual approach was taken to determining a global mapping or homography between the two images. From Figure 3.4 it can be seen how closely attached the two cameras are to each other; this means there is a wide overlap in the field of view, which results in many point correspondences between the two images. For each of the videos recorded, corresponding points between infrared and visible light are selected manually from a random infrared/visible light pair in the sequence, and a 2D homography is computed from these point pairs using the DLT and RANSAC.

The assumption taken here – that a single homography will suffice per video – is reasonably valid, as will be shown in Chapter 4. This is because the videos were recorded from fixed locations – looking at largely the same scene throughout the video – albeit with considerable camera vibration induced to simulate the type of footage one might get from a UAV. If the footage were taken from a UAV moving over terrain with significant height changes, the homography would most likely need to be recomputed at various intervals. Automatic computation of a homography between different modalities, although outside the scope of this thesis, may be possible in light of the recent work of [99]. In that paper the focus is on producing dense depth maps

from cross-spectral stereo image pairs. Computing depth maps necessitates matching features across images, and this is done by using unsigned HOG descriptors in combination with a strong (although computationally expensive) match optimisation step. That approach produces largely valid, yet coarse, dense depth maps.

### 3.1.4 Infrared Segmentation

The histogram of the pixel intensities in an infrared image is typically Gaussian-shaped (see Figure 3.6 (b), with the mode of the distribution representing the ambient temperature in the scene. Objects which are hotter than the background typically show up as bumps to the right of the main distribution. (The Gaussian assumption is not always valid, however. If there is a large hot region in the image this will dominate the histogram and consequently there will be a greater proportion of pixels near the right end of the distribution. See for example Figure 3.11 (middle). Section 3.3.2 discusses an improved method which does not require this assumption to be made). If one is to assume that the histogram is Gaussian-shaped, this initial approach attempts to find a point at which to threshold the image so that the hotter objects are isolated from the rest of the scene. This is done by first smoothing the histogram with a Gaussian kernel, then looping through the histogram values from right to left and finding a local minimum near the end. The infrared image is thresholded at this value, meaning that all pixels with a value less than this are set to 0 (black) and all values greater than or equal to this value are set to 255 (white). This results in a binary image of the sort shown in Figure 3.6 (c), which shows the people clearly segmented out.

The infrared signature of a human tends to produce a set of disjoint hotspots, representing the exposed skin regions, since the body is not a uniform temperature if they are wearing heavy clothing. This can be seen in Figure 3.6 (c). If the contours are extracted from this image, and their bounding box projected into the visible light image, it can happen that two or more bounding boxes end up being processed for the same person, but none of them cover the whole body. Because of this, the white re-

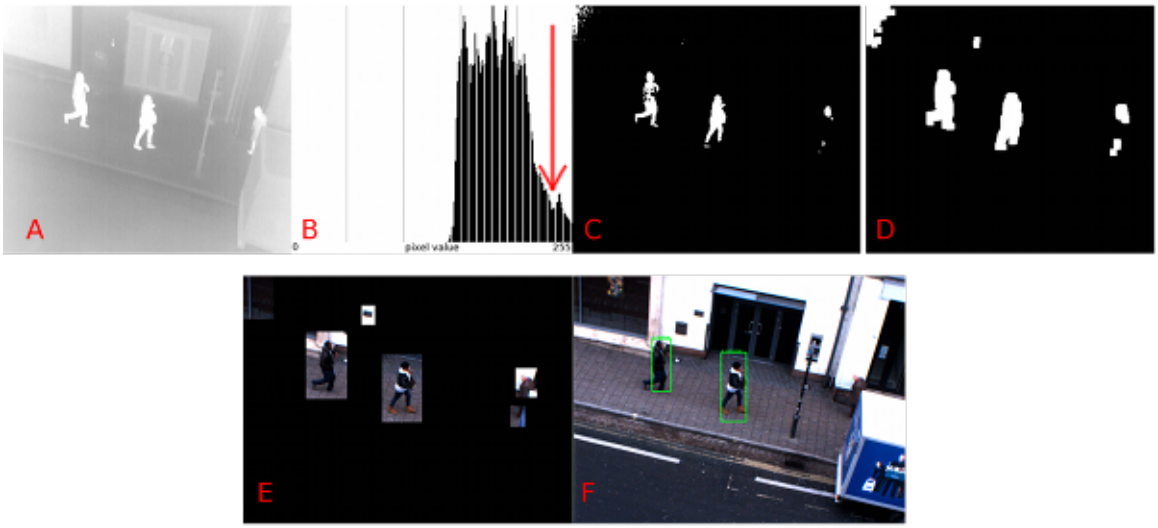



Figure 3.6: Overview of the detection method. (a) and (f) show the original infrared and visual light images. (b) shows the histogram of intensity values in the infrared image, with an appropriate threshold value. (c) is the result of thresholding the infrared image at this value and (d) dilating the resulting hot regions found. The bounding box of each of these regions is computed and projected into the visible light image, as shown in (e). (f) shows the result of running a person detection algorithm over these smaller regions of interest.

gions in the binary image are morphologically dilated[2], a mathematical operator used on binary images which has the effect of expanding boundaries and filling small holes. This tends to join the regions into one big blob. The contours of this binary image are then extracted, and their bounding box is projected into the visible light image using the homography. Once projected into the visible light image, the bounding box is inflated by 20% in order to cater for the possibility of error in the homography. This value was chosen empirically based on the approximate distance of the camera from the scenes recorded for this thesis, but it could be parametrised based on the computed reprojection error of the homography. This is the error arising from projecting points from one image into to the other according to the estimated homography. This value, which is dependent on depth to the scene, could approximate how much the

[2]Dilation and erosion are the two basic operators in mathematical morphology, typically applied to binary images. The basic effect of dilation is to gradually enlarge the boundaries of regions of foreground pixels i.e. white pixels. Thus areas of foreground pixels grow in size while holes within those regions become smaller. The erosion operator erodes away the boundaries of regions of foreground pixels, while holes within those regions become larger.

bounding box needs to be inflated in order to cater for the homography error.

### 3.1.5 Visible Light Processing

The process of detecting people in the visible image follows the standard method of running a sliding window detector over the image at different scales, but instead of searching the entire image, only the regions found using the infrared pre-processing step are processed, i.e. the regions shown in Figure 3.6 (e). In this work the Felzenszwalb part-based detector [43] (as described in Section 2.2.2) was used, owing to its success in detecting articulated people, but any (part-based) detector could be used in its place or in combination with it. The key point is that in contrast to traditional methods of detecting people, the infrared processing eliminates the need to process the entire visible light image. Furthermore, in this work the height of the camera above the ground is known, and this enables us to restrict the search to more likely scales a person may appear. This step, in addition to filtering out large parts of the image with the help of the infrared processing, helps to reduce computation times further.

Although there is a certain amount of rotation invariance implicit in the Felzenszwalb detector, in that it can detect people with varying degrees of articulation, it is not rotationally invariant in terms of the overall orientation of the person. For this work the assumption is that the person is roughly right-side up. The process could however be made rotationally invariant by scanning the image at various orientations and taking the maximally scoring one.

In order for a window scanned by the detector to be deemed a detection, its score must be above a certain threshold specified in advance. Determining the best threshold to use was determined empirically by examining the Receiving Operator Characteristic (ROC) curve obtained by varying the threshold (see Figure 3.7). The ROC curve is a plot of the true positive rate against the false positive rate at various discrimination thresholds. If the threshold is set too high, there is a chance that people may not be detected. If it is too low, then false positives are more likely to be reported. The ROC curve helps to choose a threshold which gives acceptable trade-off

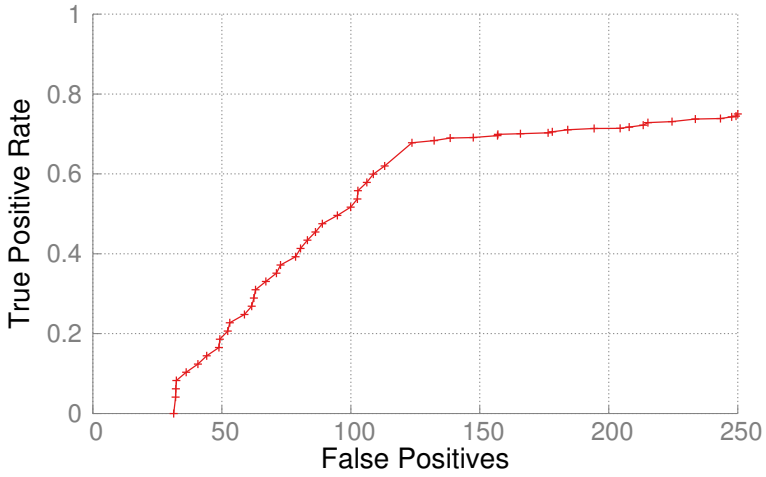



Figure 3.7: ROC curve obtained by varying the threshold score on the Felzenszwalb detector over a data set of 250 visible band images, each containing one person. The horizontal axis shows the total number of false positive detections; the actual false positive rate can be calculated by dividing this by the number of scanned detection windows.

between true and false positives for a given domain.

## 3.2 Results

The results presented in this chapter compare the standard method of detecting people in visible light images – searching the whole image – with using the infrared modality to narrow down the search to more likely parts of the scene. For the evaluation we independently sampled 250 infrared/visible light image pairs from the dataset of 47 videos recorded specifically for the DPhil. Each of these frames contains one or two people and has an accompanying ground truth file which contains the coordinates of the top left corner of the bounding box and its dimensions. The videos were taken at five locations at different times of day and time of year. Figure 3.8 shows some snapshots from the dataset.

For evaluating the success of the method, each detection in an image is compared with the ground truth file for that image. If the detection overlaps with the ground truth bounding box by more than 50% it is deemed to be a true positive; otherwise

Figure 3.8: Snapshots from the video data set showing visible light images and corresponding infrared. These show the wide range of infrared signatures a person can have, varying from sharp and distinct on a cold day to barely perceptible against a hot background.

it is taken to be a false positive. The overlap criteria used is *intersection over union* (IOU) of two bounding boxes; this is in line with how object recognition challenges such as ImageNet[3] determine correct detections. A false negative is defined as a person who was not detected at all.

The overall detection performance is evaluated with precision recall graphs. Precision and recall are defined as follows:

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

In other words, recall is the proportion of people in the entire set of images who were detected. Precision is a measure of how accurate the set of all detections is. Recall rate can be varied by varying the discrimination threshold for the detector. The lower the threshold, the more detections there will be, but with an increased false positive rate; the higher the score, the fewer detections there will be, but with a higher precision. With precision recall graphs, the goal is to have the curve in the top right corner, i.e. having high levels of recall accompanied by high levels of precision. The results of running the Felzenszwalb detector over the entire image and the segmented part of the image are shown in Figure 3.9, which shows the precision recall curve for the two methods. Intuitively, the new method should result in fewer false positives than if the entire visible image were to be searched, with no change in the true positive rate. If the processing of the infrared is able to segment out a person correctly, this turns out to be the case. If it is not, then regions which should have been searched end up not being searched, and consequently people are missed.

### 3.2.1 Overall Detection Performance

The first thing to notice is how low the recall rate is overall – simply by running the Felzenszwalb over the entire image. At a precision even as low as 50%, just over 50%

[3]http://image-net.org/challenges/LSVRC/2014/index

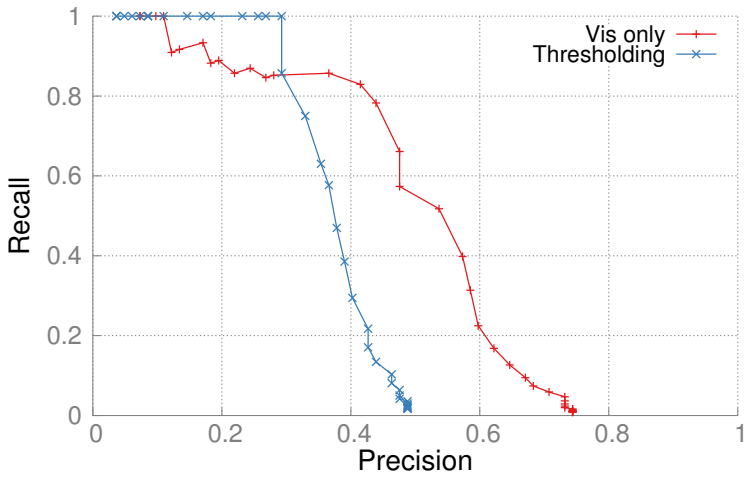



Figure 3.9: Precision recall curve of detection on a set of 250 infrared/visible light images sampled independently from the entire set of videos. The red line is the result of running the Felzenszwalb detector over the entire visible light image (making no use of infrared). The blue line is the curve resulting from using infrared thresholding to first narrow down the search to the warm regions, and then running the Felzenszwalb detector over the resulting set of regions.

of the people in all the videos are detected. This is indicative of the usual problems which beset people detection algorithms: low illumination, highly articulated people and the small size of people in some of the images. Figure 3.10 shows some tricky examples where the Felzenszwalb detector failed to detect a person. Another issue with object detectors is that even minor changes between one frame and the next can mean that a person may be detected in one frame and not in the other.

### 3.2.2 How using Infrared Affects Detection

The second thing to notice from the precision recall curve of Figure 3.9 is how much lower the recall is when the infrared is used to narrow down the search. At a precision even as low as 40%, the infrared method only manages to recall around 30% of true positives, compared to over 80% if the entire visible image is searched. The result is somewhat surprising, and suggests that thresholding the infrared image fails to segment out the people accurately. As a result, the corresponding regions never get

Figure 3.10: Some tricky examples where the Felzenszwalb detector fails to detect a person. Left: the person near the top of the steps is barely visible to the human eye. Middle: a person walking along the footpath in poor illumination. Right: people who appear too small in the image.

analysed by the Felzenszwalb detector. To see why this is happening, it helps to look at some examples of where the people are being missed. In Figure 3.11 (left), the person is standing in a grassy area which is under direct sunlight. They blend into the background. In Figure 3.11 (middle), the roof in the foreground is hotter than the person and the automatically chosen threshold ends up being too high. If it had been lower, then both the roof region and the region containing the person would have been searched. In Figure 3.11 (right), the person does not have a strong infrared signature against the warm wall behind them. The effects seen here are partly related to the thermal sensitivity of the camera. As discussed in Section 3.1.1, thermal sensitivity affects how much the infrared camera can discern small temperature differences in the scene. If the camera had a lower sensitivity then it is likely that the people would show up in greater contrast to the background in the images shown in Figure 3.11.

### 3.2.3 Where it Fails

An analysis of the results on each individual image makes it clear that the infrared thresholding process fails to segment the people in the scene when either of two conditions occur: (i) the background of the person is also hot, meaning the person blends into the background, or (ii) there is an object in the scene which is hotter than the person, and the computed threshold ends up being too high. Both conditions usually happen on a hot day. To quantify the effect which temperature has on the success of detection, we divided the dataset into two sets: those images taken on hot

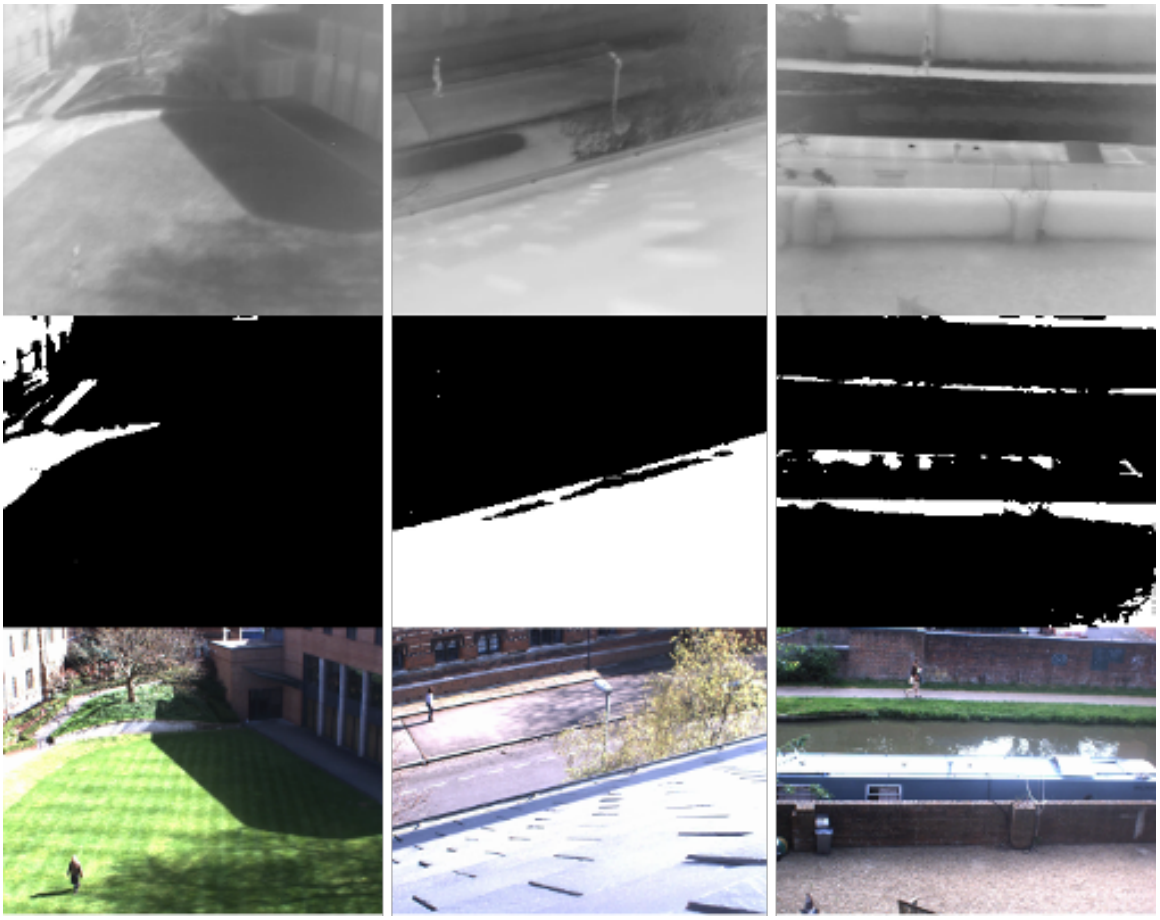
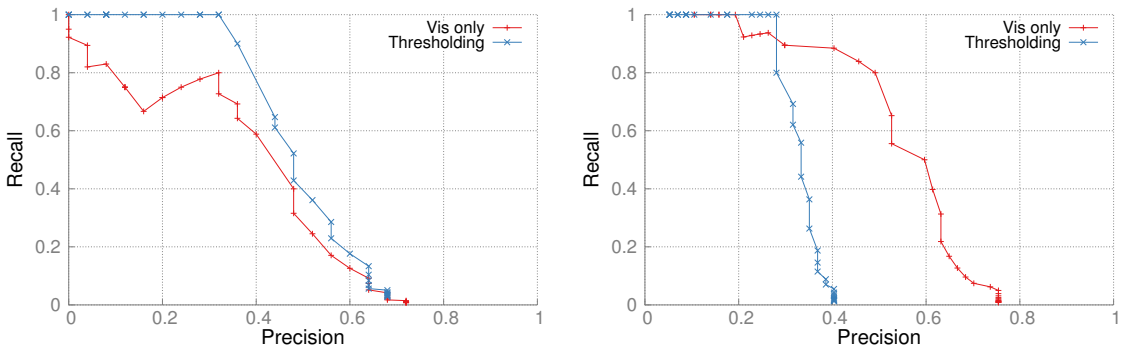


Figure 3.11: Some examples where the infrared fails to segment the person well. Left: the person blends into the hot background. Middle: the roof in the foreground results in the wrong threshold being chosen. In this case the histogram of intensity values is not Gaussian-shaped, so it is difficult to choose an appropriate threshold. Right: the person is barely distinguishable against a warm background.

days and another set containing images taken on colder days. Figure 3.12a shows the precision recall curve for each set. This shows that on colder days, using the infrared gives better precision (fewer false positives), than the visible only method. The recall rate is the same. It is clear from this that the infrared provides a clear advantage in terms of higher precision (fewer false positives) on colder days when the thermal signature of a person is more distinct against their background.

By looking at the middle image in Figure 3.11, one might be surprised to find that the person is not segmented during the infrared processing; their silhouette is clearly distinguishable from the background. There is an obvious white blob against




(a) Cold weather

(b) Hot weather

Figure 3.12: Precision recall curves showing the effect of outdoor temperature on detection with the aid of the infrared modality.

a dark background. This highlights the deficiencies of using simple thresholding to segment out the people, especially if there is something hotter in the scene. If a lower threshold were used, this would pick out the person but would result in more of the image being processed unnecessarily, even though most of the hot region corresponds to the roof in the foreground. This result suggests it would be better to look at image gradients rather than absolute pixel values, and to consider regions of light against dark and vice versa.

## 3.3 Improving Detection

This section discusses the most common problems encountered in detecting people reliably in a multi-modal image pair, and details improvements made to the basic approach which cope better with these problems.

### 3.3.1 Common Problems

The detection method described above worked very well in cases where the person was clearly distinguishable in the infrared. Usually this happens on a cold day when the body temperature of a person is much warmer than the background, or where the person is close to the camera. However, in videos where there was something

Figure 3.13: Some examples of positive (top) and negative (bottom) training examples for a machine learning algorithm algorithm to learn the appearance of a person in infrared images.

hotter in the scene, such as a roof with the sun beaming down on it, this resulted in the wrong threshold being chosen for the infrared segmentation. This meant that the person could be missed altogether, while large regions are searched unnecessarily. This suggested that looking at absolute pixel values was not always a good idea; rather, the contrast between neighbouring pixels is what should be taken into account. This is an approach adopted by others previously, namely by [51] and [109], who used cascade Haar classifiers to detect people in aerial infrared images.

Another problem noticed in some cases is that when the locations of bounding boxes in the infrared image were projected into the visible light image, the result can be inaccurate, sometimes as much as 50 pixels away from where it should be. This is because a 2D homography is a poor approximation of the mapping between two views if the scene is not entirely planar. This problem was foreseen, and to cater for the possibility of error in the homography, each bounding box, when projected into the visible light image, is inflated by 20%. While this is sufficient for providing a rough estimate of where the person is, it is not a good method if the person is being tracked. A better method of finding corresponding points between the two images is needed.

### 3.3.2 Improved Infrared Segmentation

This section presents the results of using of a histograms of oriented gradients (HOG) detector [29] and a local binary patterns (LBP) detector [91] in order to detect people in infrared images. These are proposed as better alternatives to intensity thresholding, as they are not dependent on absolute pixel values. In the case of HOG (discussed

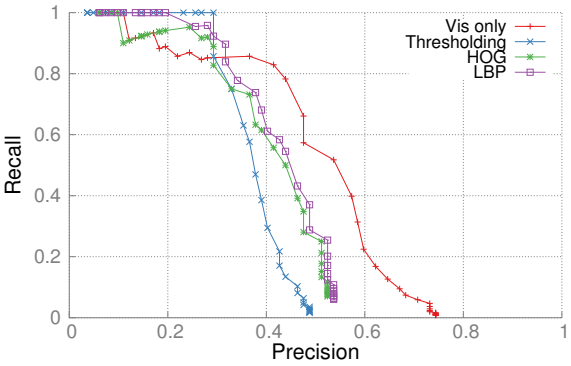
Table 3.1: Training, validation and test set errors for HOG and LBP classifiers.

| | HOG | LBP |
|---|---|---|
| Training error | 4.73% | 10.53% |
| Validation error | 8.40% | 28.52% |
| Testing error | 18.64% | 57.66% |

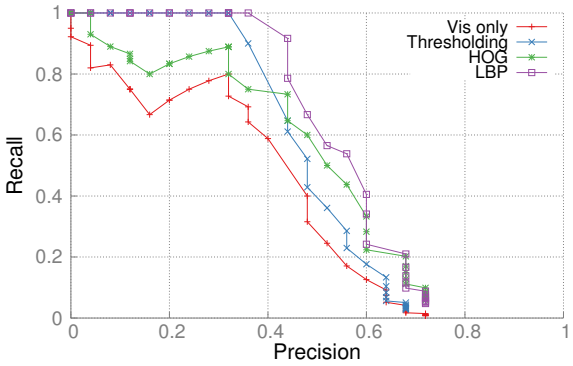
in Section 2.1.3.3), a histogram of oriented gradients is built up for each image in the training set, and a support vector machine is used to learn the decision boundary which best separates positive feature vectors from negative feature vectors. In the case of local binary patterns (discussed in Section 2.1.3.2), each pixel is compared with the 8 pixels in its neighbourhood, and the difference between it and its neighbours is encoded as a 0 or a 1 in a feature vector, and this is done in each cell and concatenated into a single feature vector for a training example. The appearance is learned using decision tree stumps and AdaBoost (both are explained in Section 2.1.2.2).

HOG and LBP classifiers were trained to detect bright blobs in the infrared. 50% of the data set was used for actual training while 25% was used for validation and 25% for testing. Figure 3.13 shows some examples from the positive and negative training set chosen specifically for this task. These were extracted both from the data set collected for this DPhil and from the freely available OTCBVS data set[4] and the data set of Portman et al. [100]. Initially, a HOG detector was trained for the task of detecting upright people in infrared images using 500 positive and 1000 negative training examples. However, running a HOG detector over the infrared image is more computationally demanding than thresholding – it takes 482 milliseconds to do HOG compared to 2 ms for thresholding. For speed reasons it was decided to try the less computationally demanding LBP detector, which takes 63 ms to run over an image. For the LBP detector, a much bigger negative training set (20000 images) was required. The final training set was generated by randomly warping and mirroring examples from an original training set. Figure 3.1 shows the training, validation and test set error for both classifiers. As can be seen, the error is quite high on the test set, especially for the LBP classifier, but we were not aiming for precision at the

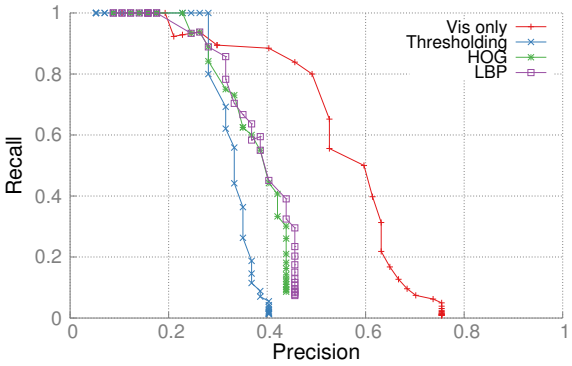
[4]OTCBVS Benchmark Dataset: http://www.vcipl.okstate.edu/otcbvs/bench/




(a) All videos




(b) Cold weather videos




(c) Hot weather videos

Figure 3.14: Effect of different infrared segmentation methods (thresholding, HOG and LBP) on tracking precision. In hot weather, no infrared segmentation method can improve precision levels over searching the entire visible light image. In cold weather, all three infrared processing methods help to narrow down the search so that there are fewer false positives than the visible only method. This is reflected in the higher levels of precision for each level of recall in Figure 3.14b.

infrared processing stage. These bounding boxes are merely used to guide the search in the visible band, so it is expected that false positives here will be filtered out at a later stage.

The results of using these two classifiers to segment out people in the infrared image are included in the precision recall curves of Figure 3.14. These are the same curves as shown in Figure 3.12, but showing the results of using HOG and LBP to compare to simple thresholding. There is a clear improvement over thresholding in

terms of precision and recall – both HOG and LBP give a higher overall recall and accompanying precision than using thresholding. They are more computationally demanding than thresholding, but still take less time than if the whole visible light image is processed.

However, it is clear that the infrared modality does not always provide the information required to narrow down the search for people, and this is the disadvantage of using the infrared to guide the search in the visible light image. Perhaps it is better to search to the entire visible light image, and to use the presence or absence of the person in infrared as an additional confidence measure. However, if reduced computation time is required, it would be useful to know when the infrared should not be trusted; if this was known, then the whole visible light image should be searched. This issue is as important for tracking as it is for detection and is something we return to in Chapter 5.

### 3.3.3 Improved Matching Across Modalities

From analysing the results of those videos taken at close range it is clear that a planar homography can be a poor approximation to the mapping between the infrared image and the visible light image; this is due to effects of parallax caused by objects at different depths having different disparities between the two images. This section describes an attempt made to fine tune the matching between a thermal signature of a person and their corresponding appearance in the visible light image.

Previous attempts at matching features across a multi-modal image pair have been fraught with difficulty because feature descriptors of the same object can be very different in different modalities. One issue is contrast reversal, i.e. where an object may appear light against a dark background in the infrared image, but appear dark against a lighter background in the visible light image. In that situation the gradient directions can be opposite in the two images.

Previous work in cross-modal stereo matching [99] has used unsigned HOG descriptors to match features in an infrared/visible light image pair, in that case for the purposes of depth estimation. In that approach they use a dense set of unsigned HOG

features and match these across images using standard stereo matching algorithms normally used for a pair of optical images. Using unsigned as opposed to signed gradient directions makes sense because it means the feature descriptor is more invariant to contrast reversal.

Inspired by that work, we use HOG descriptors to match a region in an infrared image to the visible light image. For a given bounding box in the infrared, the unsigned HOG descriptor is extracted and then a search is done in the visible light image to find the closest match. The most obvious way to do the matching would be to slide a window over the whole visible light image and store the best match found. However, it was found in practice that this is quite prone to error, especially if the infrared signature is not that distinct to begin with. Instead, we make use of the approximate location provided by the homography and use this as a starting point to fine tune the localisation.

Given a bounding box in the infrared, we wish to find the closest matching bounding box in the visible band image. The infrared bounding box is projected into the visible image using the pre-computed homography. Searching within a rectangular region around the corresponding point found by the homography, and finding the location which gives the minimum Euclidean distance between the two HOG descriptors, was found to give better results than using the homography estimate alone in the vast majority of cases. As the original infrared bounding box and its projection in the visible light image are different sizes, in order to compare the two HOG descriptors we downscale the visible patch to match the infrared patch. The decision to downscale the visible patch rather than upscale the infrared patch was made because we found that the extra detail in the visible patch tends to confuse the HOG matcher. Figures 3.15 and 3.16 shows some examples of the match found when this method is used. The downside of the approach is the extra computation time required to find the minimum match, something on the order of 30 ms, but still capable of running in real-time on the hardware available today.

This method was tested on 300 image pairs randomly selected from the set of 47 videos. To compare the homography estimate with the HOG matching estimate, we

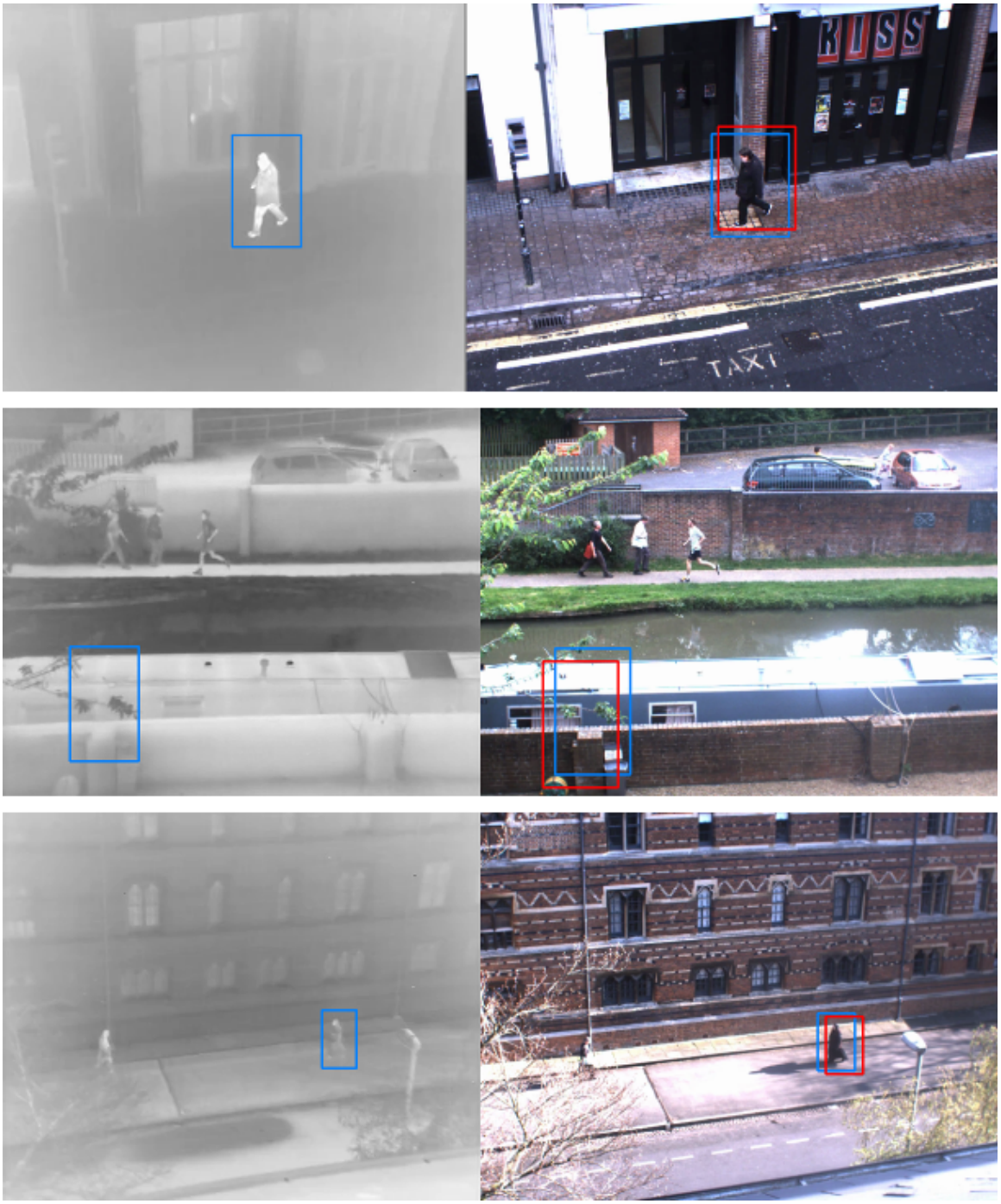



Figure 3.15: Improved matching between infrared and visible light images using HOG descriptors. Left: The HOG descriptor is extracted from the blue box in the infrared image. Right: Red boxes are the estimated location according to the homography mapping computed manually in advance. Blue boxes are the regions having closest matching HOG descriptor to the original infrared descriptor.

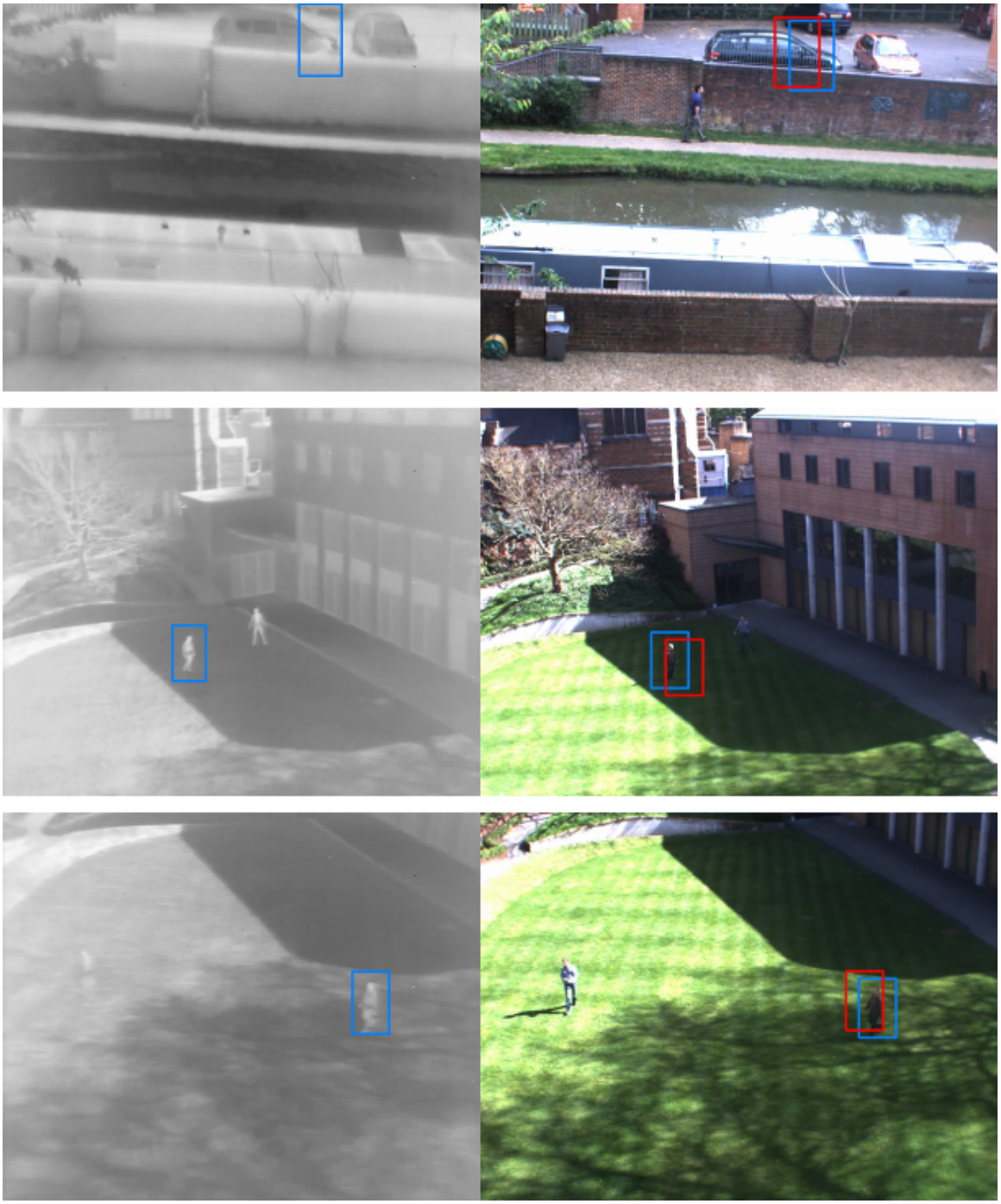


Figure 3.16: Improved matching between infrared and visible light images using HOG descriptors. Left: The HOG descriptor is extracted from the blue box in the infrared image. Right: Red boxes are the estimated location according to the homography mapping computed manually in advance. Blue boxes are the regions having closest matching HOG descriptor to the original infrared descriptor.

compute the Euclidean distance between the bounding box centres of both estimates and the centre of the ground truth box. Out of 300 image pairs, in only 24 images (8%) was the match provided by the homography alone closer to ground truth than the match found by looking for the closest HOG descriptor, when matching from infrared to visible light (that is, the original descriptor is extracted from the infrared and the closest match is found in the visible light). However, if the method is carried out the other way around, that is, given a salient region in the visible light image, try to find its correspondence in the infrared, in only 32% of cases does the HOG matching provide a closer estimate than the homography.

## 3.4 Computation Times

Table 3.2 shows computation times for the overall detection method described in this chapter. The code was run using a single core on a Macbook Pro 2.5 GHz Intel Core i5 with 4 GB of RAM and no GPU. The upper table shows the computation times for the three alternative methods of finding hotspots in the infrared image, thresholding, HOG detection and LBP detection. These can be used interchangeably, with varying results as already discussed. Thresholding is the quickest method, but is the least effective in terms of segmenting out the person. Of the two gradient based detectors, LBP is by far the most efficient, taking 63 ms per frame, compared to HOG which takes more than seven times that time.

The lower part of Table 3.2 shows the time to process the visible light image with the Felzenszwalb detector, depending on whether the whole image is being searched (which would be the case if there were no infrared), and a reduced region of interest found by the infrared segmentation method. On average, it takes more than 2 seconds to process the full $1280 \times 960$ image with the Felzenszwalb detector. Although the regions found by the infrared segmentation vary depending on the scene and segmentation method chosen, on average it takes 238 ms to process the more likely visible light regions. The code was run on a standard CPU, but as discussed in Section 2.2.2, speeded up results could be obtained on a GPU and/or using a cascaded version of

Table 3.2: Computation times for detection

IR Segmentation (584×474 px)

| **Segmentation method** | Computation time (ms) | ms per 1000 px |
|---|---|---|
| Thresholding & connected components | 3 | 0.01 |
| HOG detector | 482 | 1.74 |
| LBP detector | 63 | 0.23 |

Visible light detection (1280×960 px)

| **Felzenszwalb detector** | Computation time (ms) | ms per 1000 px |
|---|---|---|
| Full image | 2456 | 1.99 |
| Reduced ROI after IR segmentation | 238 | 0.19 |

the algorithm.

## 3.5 Summary of Results

This section summarises the results that were observed in the experiments presented in this chapter.

**The difficulty of detecting people in aerial imagery**

Detecting people in the aerial imagery recorded for this thesis is a difficult problem even using state of the art people detection algorithms. The benchmarks for such detection algorithms are image classification challenges such as ImageNet (formerly PASCAL VOC Challenge), where the scenarios, though difficult, are still not as difficult as detecting people in the aerial imagery recorded for this thesis. The problem is made especially difficult by the small size of the person in the image and poor lighting conditions. Although great improvement has been seen on benchmark datasets with current state of the art algorithms [33, 34, 35], it is clear that, with the lower level of detail (on the person) present in many of the images captured for this thesis, some contextual knowledge is required by a computer in order to be able to detect the people recorded in these videos.

**The benefits of having an extra modality**

There are cases in which using infrared imagery improves the detection performance and there are cases in which it hinders it. The results depend largely on how good the infrared is at segmenting the person from the background, and this is dependent on weather conditions (temperature and humidity), whether or not the person is in the shade and whether there are objects which are hotter than the person in the scene. The segmentation method used also has a significant effect on detection: using detectors specifically trained to recognise white silhouettes against a darker background tends to do a better job of identifying person candidates than thresholding does. This is because thresholding focuses on finding the hottest object in the scene, which is not guaranteed to be a person, and it is difficult to determine what the appropriate threshold value should be.

**Using infrared reduces computation times**

Using the infrared modality to narrow down the search for people reduces the computation times to detect a person over processing the entire visible light image, even after taking into account the additional infrared processing required to find regions of interest. The bottleneck of the entire method is running the parts-based people detection algorithm on the visible light regions. If it is possible to segment the person accurately in the infrared, then only a small region must be searched in the visible light image. The time taken to do the infrared processing is negligible compared to the visible light processing. This is because of the lower resolution of the infrared, and that the methods for identifying potential candidate can be done in time linear in the number of pixels.

---

This concludes the explanation of the basic method used to detect people using a combination of infrared and visual light imagery. While previous work such as [51, 104, 109] has focused on footage taken from UAVs flying at high altitudes (where the person occupies only a small part of the image), our work focuses on close-range video footage which would be taken from small robotic helicopters. Such footage

allows for the use of more complex part-based detection algorithms. In contrast to [51], which processes both modalities independently and then combines the result into a single confidence map, our method reduces the search space from the very beginning, similar in spirit to [109] – by first processing the lower resolution infrared image and using that to reduce the search space in the visible light. While [51, 109] use a homography to map one modality to the other, this chapter presented a method to match two images using HOG descriptors and showed that this results in more accurate matching than using a homography alone. The next chapter looks at tracking a person once they have been detected, that is, predicting where a person will next appear, so that it is not necessary to keep running the full detection pipeline presented in this chapter.

# Chapter 4

# Multi-Modal People Tracking

This chapter presents one of the main contributions of the thesis – HeatTrack – an algorithm for tracking people using a multi-modal camera setup. A description of the basic approach is followed by a number of improvements to the algorithm which resulted in more accurate tracking compared to three state of the art tracking-by-detection algorithms [11], [66] and [58].

**Algorithm 3** Kalman filter

1: **procedure** KALMANFILTER($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)
2: $\bar{\mu_t} \leftarrow A_t\mu_{t-1} + B_tu_t$
3: $\bar{\Sigma_t} \leftarrow A_t\Sigma_{t-1}A_t^T + R_t$
4: $K_t \leftarrow \bar{\Sigma_t}C_t^T(C_t\bar{\Sigma_t}C_t^T + Q_t)^{-1}$
5: $\mu_t \leftarrow \bar{\mu_t} + K_t(z_t - C_t\bar{\mu_t})$
6: $\Sigma_t \leftarrow (I - K_tC_t)\bar{\Sigma_t}$
7: **return** $\mu_t, \Sigma_t$

The previous chapter describes an approach to initially detecting a person in a scene without any prior knowledge of the location of the person. Although the proposed method is quicker than searching the entire visible light image, ideally this would not have to be carried out in every single frame – once the person has been detected it should be possible to predict roughly where they will appear in the next frame. This chapter describes an approach we developed to track people from frame to frame i.e. localise them in every frame without having to run the full computationally expensive detection pipeline. Rather than search the whole image every time, the method attempts to gauge the velocity of a person and use this in order to predict where to look in the current frame.

## 4.1 The Basic Approach

Our approach, denoted HeatTrack hereafter, uses a Kalman filter to track a person in 2D image space. The filter takes measurements from both image modalities and uses these to update a model of the $(x, y)$ position of a person and their velocity in pixels. In the absence of measurements i.e. if the person becomes occluded, a motion model is used to predict where the person is. This section describes the approach to tracking with a stationary camera. In that case, a constant velocity model is a reasonably valid assumption: in general we would expect a person walking along a footpath or hillwalking to be walking at an approximate speed of 4 mph. With a moving camera, however, the constant velocity model is obviously violated. In Section 4.2 we describe how we adapted our method to work in the case of a moving camera, which is the case for all of the videos captured for this DPhil.

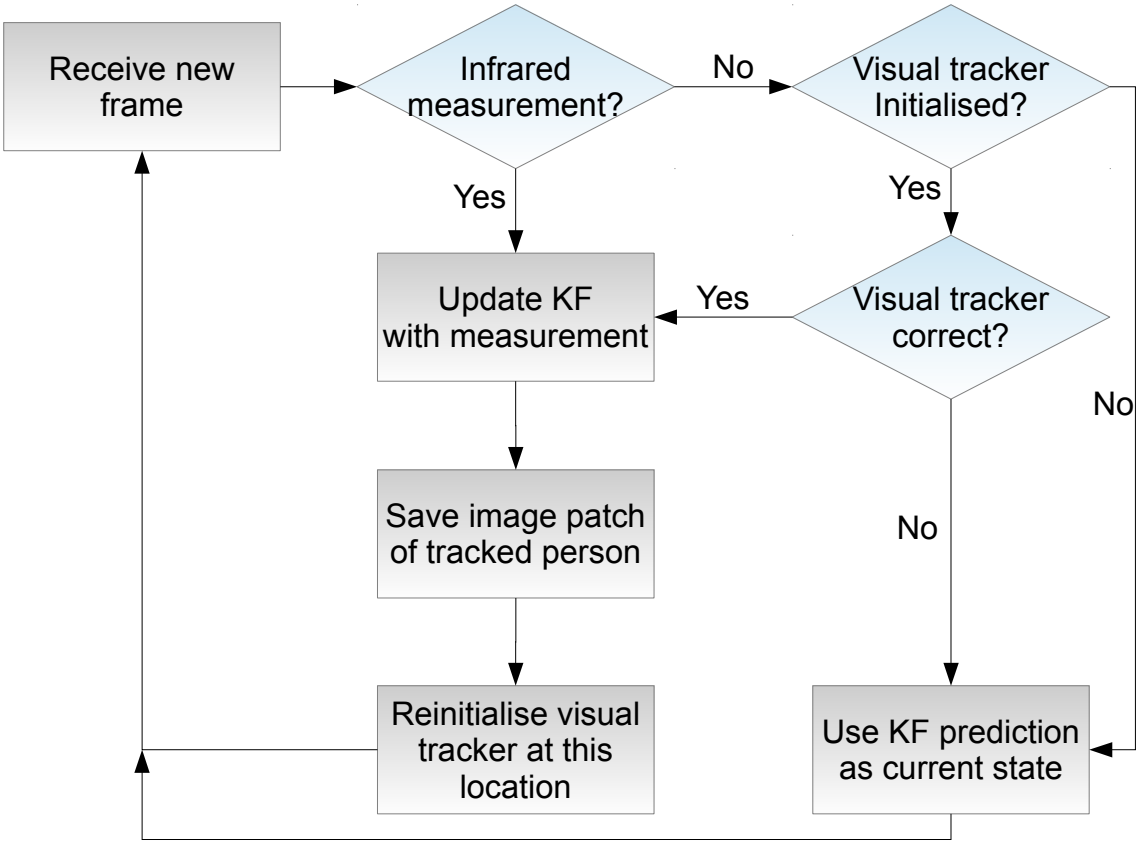



Figure 4.1: Overview of the tracking method. This process is initiated once a person has been detected with confidence.

Figure 4.1 shows a high level overview of how the HeatTrack algorithm works. We chose a Kalman filter for this implementation primarily because of its low computational cost. A particle filter could also be used for this purpose, but it may need a large number of samples in order to converge to the true posterior, which increases the computational complexity. The general Kalman filter algorithm was described in Chapter 2 and is shown again below for ease of explanation. When a new person is detected, using the method described in the previous chapter, a Kalman filter is initialised with their location in pixels and a velocity of 0. Given measurement models for both modalities, the Kalman filter then proceeds to incorporate measurements from both modalities in every subsequent frame. The specific measurement models used are described in sections 4.1.1 and 4.1.2.

In this implementation of the tracker, preference is given to the infrared over the

visible light, and the Kalman filter coordinate system is the coordinate system of the infrared image. If a measurement is received from infrared, this is used to update the Kalman filter; if there is no infrared measurement but there is one from the visual light, then that is used instead. The reason we give preference to the infrared modality is because the infrared signature of a person is less susceptible to lighting changes and body articulation and hence is easier to track.

Given that tracking occurs in the infrared images, measurements received from the visible light image must necessarily be converted to the coordinate system of the infrared image before being input to the Kalman filter. If there are no measurements at all from either modality, then the prediction of the Kalman filter is used as the estimate for where the person is.

### 4.1.1 Infrared Measurement Model

Each infrared frame is processed to segment out potential person candidates using any of the methods described in the previous chapter – either by thresholding to find the hottest parts of the image, or by using a method which looks at pixel differences, such as histograms of oriented gradients or local binary patterns. Whichever method is used, the result is a set of bounding boxes, one of which may be associated with the person being tracked (the approach to data association is detailed in Section 4.1.3). The coordinates of the centre of this bounding box are used as the measurement vector in the Kalman filter.

### 4.1.2 Visible Light Measurement Model

Every time a measurement is received from the infrared image, an independent tracker is initialised at the corresponding location in the visible light image. The purpose of this tracker is to continue tracking in the absence of any infrared measurements, but if an infrared measurement is received, this tracker is re-initialised at the corresponding location. As a result, the visible light tracker is relied upon only when there are no infrared measurements. One example of where this is useful is when the person is under direct sunlight. In that case, the person tends to blend into the background in

the infrared, but shows up very clearly in the visible light, making it easier to track in that modality. The estimate of this independent tracker is used as a measurement in the Kalman filter.

We tested a number of different state of the art tracking algorithms on our own dataset. These trackers are ones which have outperformed most other tracking algorithms on a wide range of tracking tasks as detailed in the extensive surveys of [134] and [111], namely TLD [66], MILTrack [12] and Struck [58] (detailed in Section 2.3.1). Each of these trackers provides an estimate of the location of the person in the form of a bounding box in the visible light image. The coordinates of the centre of this bounding box after being projected into the infrared coordinate system are used as the measurement vector in the Kalman filter. It will be seen later in the chapter that Struck outperformed the other two trackers especially in the case of videos exhibiting camera vibration. The tracking results to be presented in this chapter were obtained using Struck as the visible light measurement model.

### 4.1.3 Data Association

Given that there can be several candidates in the infrared which may correspond to the person being tracked, there is a data association problem that must be solved before updating the Kalman filter with a measurement. The issue is especially important if multiple people are being tracked, as one must decide which measurement to assign to which person. Classic approaches to data association include Multiple Hypothesis Tracking [102] and the Joint Probabilistic Data Association Filter [48]. MHT maintains multiple possible associations over several time steps, but the complexity of the algorithm limits its application to just a few time steps. JPDAF instead tries to make the best possible assignment in each time step by considering all possible assignments between detections and targets, this having exponential complexity. The method developed for this DPhil uses a greedy mechanism whereby the bounding box having the best score over various different criteria is chosen as the measurement to be incorporated into the Kalman filter. This method is less computationally intensive than JPDAF or MHT and is more suited to real-time short-term tracking (tracklets)

for our short video sequences. The criteria are:

1. **Size and aspect ratio** - The dimensions of a candidate bounding box are compared with the saved image patch of the tracker. If the dimensions are too different the measurement is ignored. There is no exact science to this, as depending on the way a hot spot was identified in the infrared, only the upper body may have been detected; ideally this detection would not be discarded.

2. **Proximity to previous tracker estimate** - If a bounding box is too far from where the person was last tracked, the box should not be considered. The proximity measure is based on the covariance of the Kalman filter state estimate. If the box does not lie within the covariance ellipse it is ignored.

3. **Appearance match** - the visible light image patch corresponding to the candidate measurement is compared with the saved image patch of the tracker. The appearance score is the average of intersection and correlation scores between the HSV histograms of both patches. Intersection and correlation are two ways of comparing the similarity of two histograms. In both cases, a score of one indicates a perfect match and a score of zero indicates a perfect mismatch. Given that these scores are variable, we consider the average of the two scores for extra robustness. If this is less than a certain threshold the measurement is ignored.

If a bounding box satisfies the size requirement, is within the covariance ellipse of the Kalman filter and has a histogram score of at least 0.4, it is taken as a measurement. The covariance ellipse is a spatial ellipse on (x,y) position rather than a 4D ellipsoid in the full Kalman space. The ellipse is computed from the eigenvalues and eigenvectors of the state covariance matrix, with the eigenvectors representing the major and minor axes and the eigenvalues indicating the extent of the ellipse. The required histogram score is deliberately set to a low value because histogram matching is often not a reliable measure of similarity. Two consecutive image patches of the same person may have a low histogram match if the background has changed slightly or if the patch covers a slightly different area of the body. This method for testing the validity

of candidate image patches is used in both modalities. In the case of infrared, it is used to choose from among a number of potential candidate bounding boxes returned by the segmentation; in the case of visible light, it is used to determine whether the independent visible light tracker is correct or not.

### 4.1.4 Handling Occlusion

If there is no measurement in either modality satisfying the criteria of Section 4.1.3, then the prediction of the Kalman filter is used instead. This is the prediction based on the estimated motion model before a measurement is taken into account. The prediction is only reliable if the Kalman filter has already received observations over a number of frames, because otherwise the velocity will not have converged to an accurate estimate, and therefore the location estimate will be wrong. The length of time it takes to converge to an accurate velocity is dependent on the process and measurement noise covariance values chosen for the Kalman filter, but in the worst case it was found to take roughly 30 frames, which corresponds to 2 second's worth of video. In that case the Kalman filter is tweaked in favour of trusting the process model more than the measurement model – so it does not follow the measurements closely – and as a result it takes longer to converge to the true velocity.

In each frame where no measurement was received, the state covariance of the Kalman filter increases - an indication of the increasing uncertainty surrounding the location of a tracked person. The search vicinity used during data association, whereby measurements which are too far away are ignored, increases. When a new measurement is subsequently received, the covariance goes down.

### 4.1.5 Tracking Multiple People

Although the capability to track multiple people is not an explicit goal of the DPhil, the current implementation is able to deal with the case of two people overlapping. As long as a Kalman filter has been initialised on both, when their bounding boxes overlap the system detects an overlap and the Kalman filters stop taking measurements. As long as the bounding boxes are overlapping, both trackers rely solely on

**Algorithm 4** Bayes filter

1: **function** BAYES FILTER($bel(x_{t-1}), u_t, z_t$)
2: **for all** $x_t$ **do**
3: $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$
4: $bel(x_t) = \eta\, p(z_t|x_t) \overline{bel}(x_t)$
5: **return** $bel(x_t)$

the Kalman filter prediction before the update step (the situation is handled the same as when a person becomes occluded). When the bounding boxes of the two people are no longer overlapping, as long as the appearance of both people has not changed dramatically since the last measurement, each tracker is able to associate itself with the correct measurement and tracking resumes using measurements to update the filter.

The success of tracking under occlusion and tracking multiple people depends largely on how good the data association is when the person reappears, and comes down to determining good values for the data association criteria of Section 4.1.3. A greedy approach is currently used, but it is possible to put the data association problem into a Bayesian framework. As a simple example, imagine the person being tracked becomes occluded and there are multiple hypotheses for where they could be, which can be seen as paths the person might have taken. The general Bayes framework (upon which the Kalman filter is based) is shown in Algorithm 4 (taken from [121]). Here $bel(x_t)$ is a probability distribution over possible states of the tracked person, $x_t$ is the state vector (containing position $y_t$ and velocity $v_t$), $u_t$ is a control input (often unknown to the system), $z_t$ is a measurement and $\eta$ is a normalisation operator to ensure the updated probability density function sums to one. Lines 3 and 4 correspond to the predict and update steps of the Kalman filter. The state vector could be modified to include the path of the person i.e. $x_t = \langle y_t, v_t, \rho_t \rangle$, where $\rho$ is the set of possible paths. Then the prediction step in Line 3 becomes:

$$\overline{bel}(\langle y_t, v_t, \rho_t \rangle) \;=\; \int\int \sum_{c \in \rho} p(y_t, v_t, c|u_t, x_{t-1}) bel(x_{t-1}) dv_{t-1} dy_{t-1}$$

i.e. doing an additional summation over the possible paths the person could have taken. This step is no more difficult than before, but is slightly more computation-

ally expensive. The difficulty here is in computing the state transition probabilities $p(y_t, v_t, \rho_t | u_t, x_{t-1})$, but in the discrete case it would be possible to make reasonable guesses as to the position: a person who was moving from left to right is more likely to be in that direction than the opposite.

## 4.2 Tracking with a Moving Camera

The method described up to this point is for the case of a stationary camera looking at a person moving. In that case, the Kalman filter can be in a single coordinate system, that of the image in which the person is moving, and a constant velocity model is reasonably valid. If the camera is moving, however, a constant velocity model is no longer valid, and the question arises as to how to determine the velocity of a person if there is no fixed coordinate system. Tracking the person in 3D scene space is desirable, but this would require accurate estimation of the distance to the person being tracked. This is only possible with a wide baseline stereo pair, or if there is sufficient distance between pairs of consecutive frames so that the distance to a person can be triangulated. Alternatively, if we can assume an approximate height of a person, and the focal length is known, the depth can be computed using the standard pinhole projection model as is done in [72]. We instead focus on tracking within the 2D image space, and attempt to cancel out the effect of camera motion so that a constant velocity model may be used as in the previous section.

The approach we adopted is inspired by the video stabilisation algorithms of [63, 80, 84], which use 2D affine transformations (homographies) to estimate the image motion between frames which are then warped back to the original frame using the inverse of the estimated motion. The result, if the homographies are correct, is that the sequence looks like it was taken from a stationary camera. Figure 4.2 illustrates this idea. On the top are four consecutive video frames, with an estimated affine transformation $\mathbf{H}_{ij}$ between each consecutive pair. An object imaged at location $\mathbf{x}_1$ in image 1 will appear in image 2 at location $\mathbf{x}_2 = \mathbf{H}_{12}\mathbf{x}_1$ where $\mathbf{H}_{12}$ is the affine transformation going from image 1 to image 2. Conversely, the same object imaged

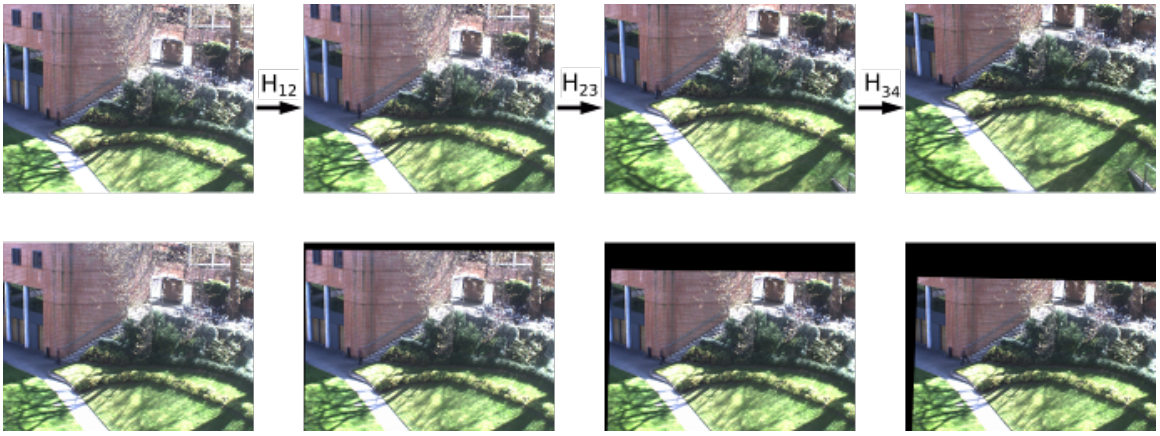



Figure 4.2: Top row: A sequence of four consecutive visible light frames. Between each consecutive pair there is a 2D affine transformation. Bottom row: Each frame warped by the inverse of the camera rotation/translation.

at $\mathbf{x}_2$ in image 2 will appear at $\mathbf{x}_1 = \mathbf{H}_{12}^{-1}\mathbf{x}_2$ in image 1. By matrix composition and inversion, one can convert the imaged location $\mathbf{x}_n$ of an object in image $n$ to its imaged location in image 1: $\mathbf{x}_1 = (\mathbf{H}_n\mathbf{H}_{n-1}\ldots\mathbf{H}_1)^{-1}\mathbf{x}_n$. The bottom row of Figure 4.2 shows what happens when image frames are warped back to an original frame using these equations. The effect, if the estimated 2D transformations are correct, is that the warped images, if placed on top of one another, should match up pixel by pixel – except for those objects in the scene which are moving independently of the camera.

The homography model is not valid if there are significant depth variations in the scene, because objects at different distances from the camera will have different disparities between two images. However, if the scene is far away enough i.e. nearly planar, the homography model will suffice. This is a key assumption for the tracker developed for this DPhil: given that the camera is of aerial scenes at least 20 metres away, it is assumed that the motion between two images can be reasonably approximated by a homography.

We use homographies to estimate the 2D motion in an image, but instead of using these to warp the images back to the original frame as the stabilisation algorithms do, we leave the images untouched and use the homographies to convert pixel coordinates from the current frame back to a reference frame. It is in this reference frame that

the Kalman filter has its state. This makes it possible to express all measurements in a common coordinate system which is necessary for the Kalman filter to estimate velocities.

The question remains as to whether the Kalman filter state should be the location of the person in the infrared image or the visible light image. Regardless of which is chosen, measurements must be converted from visible light to a reference infrared frame or vice versa. In this DPhil, the initial infrared frame in which the person is detected is chosen as the reference coordinate system in which the Kalman filter operates. In other words, the measurements input to the Kalman filter must be projected into the original infrared frame, and the output of the Kalman filter will also be a location in the original infrared frame.

As an example, assume there is a new measurement in visible light frame $n$ which must be input to the Kalman filter. The pixel location of the measurement must first be projected into the reference visible light frame using the intermediate 2D affine transformations between each visible light pair, and then converted to the corresponding location in the reference infrared frame. Assume the measurement is at $\mathbf{x}_n$ in the current visible light frame. If $\mathbf{H}_{ir\_vis}$ is the pre-computed homography between the infrared and visible modalities, then the corresponding location in the reference visible light frame is is $\mathbf{H}_{ir\_vis}^{-1}(\mathbf{H}_n\mathbf{H}_{n-1}\dots\mathbf{H}_1)^{-1}\mathbf{x}_n$, where the $\mathbf{H}_n$ are the intermediate motion estimates between each consecutive pair of visible light frames.

An important step in removing the effect of camera motion is determining the most accurate inter-frame homographies possible. The basic algorithm in the literature works as follows: (i) extract SIFT or SURF key points from both images (as described in Section 2.1.1.1), (ii) find corresponding points in both images using a brute force matcher, and (iii) estimate the 2D affine transformation from this set of points (as described in Section 2.1.1.1). For more robustness, RANSAC is used to try out many different combinations of corresponding point pairs, and the best estimate – which has the most amount of inliers after reprojecting points from one image into the other according to the estimated transformation – is used as the final homography. As an extra measure to ensure the homography is accurate, the extracted SURF key points

from each image are sorted by their response – a measure of how robust the key point is to changes in illumination and viewpoint – and the top $n$ key points are used in the homography computation. In practice, setting $n$ to 100 works well while keeping computation time less than a second per frame.

In the current implementation, features are extracted from the full $1280 \times 960$ visible light image and matched to the previous frame using this method. Motion blur, caused by abrupt movements of the camera, can give very inaccurate results. In practice, due to the fast frame rate, the homography between each pair of frames is close to the identity matrix, and it is easy to determine if the homography computation is wrong. If this happens, the estimate is ignored. The tracking algorithm waits until it is possible to compute an accurate homography between a new frame and the last frame in which the person was successfully tracked. In the experiments of this thesis, motion blur caused by abrupt movements of the camera tends to be rare, and lasts only a frame or two. Given a frame rate of 15 frames per second, the system can afford to go without a couple of frames and still be able to compute an accurate homography from the next clear frame and the previous frame in which the person was successfully tracked.

## 4.3 Results

This set of experiments examines how using infrared in combination with visible light video footage helps to track a person in the scene. In these experiments it is assumed that the person has already been detected with a high degree of confidence using the method of Chapter 3, and that their initial bounding box is known. The task now is to localise the person in every subsequent frame without having to run the original detection method each time.

Evaluating the performance of a tracker can be difficult. Which is better: a tracker which tracks an object very closely for most of the video but then loses the object completely near the end of the video, or a tracker which follows the object not as closely but for the entire video? Qualitative analysis on individual video clips

is most common, and this is dealt with in Sections 4.3.2 and 4.3.3. Quantitative comparison involves plotting centre location error against frame number. For this thesis the tracker is quantitatively evaluated using three different criteria:

1. Average tracking error, which is the sum of the individual per-frame tracking errors divided by the total number of frames in the data set.

2. Tracking error plots (for each video), which plot the tracker error as an $x$-$y$ plot with frame number on the horizontal axis and tracking error in pixels on the vertical axis.

3. Precision plots which show, for a given threshold in pixels, the proportion of frames for which the tracking error was less than the threshold.

Since tracking error plots can be difficult to interpret, it is common to take the average tracking error over all video frames. However, this can fail to accurately capture tracking performance. If an object is successfully tracked for most of the video but is then lost towards the end of the video, the average tracking error may be higher than if the object was tracked less closely but for the whole video. For this reason, and similar to [12] and [137], precision plots are used as part of the quantitative analysis. An important point to consider when analysing the tracking error is the ground truth labelling error for each of the videos annotated. It was found empirically that the ground truth labelling error for our videos is 6 pixels on average, that is, if the same image is annotated twice we can expect the bounding box centres to differ by approximately 6 pixels.

### 4.3.1 Overall Tracking Performance

This section examines the overall tracking performance of HeatTrack over a data set of 47 videos recorded from aerial vantage points around Oxford (Appendix A shows a comprehensive set of snapshots from the set of videos). Given that there is no prior work done on the type of aerial video footage being considered here i.e. close-range aerial footage with a multi-modal rig, the new method is compared with state of

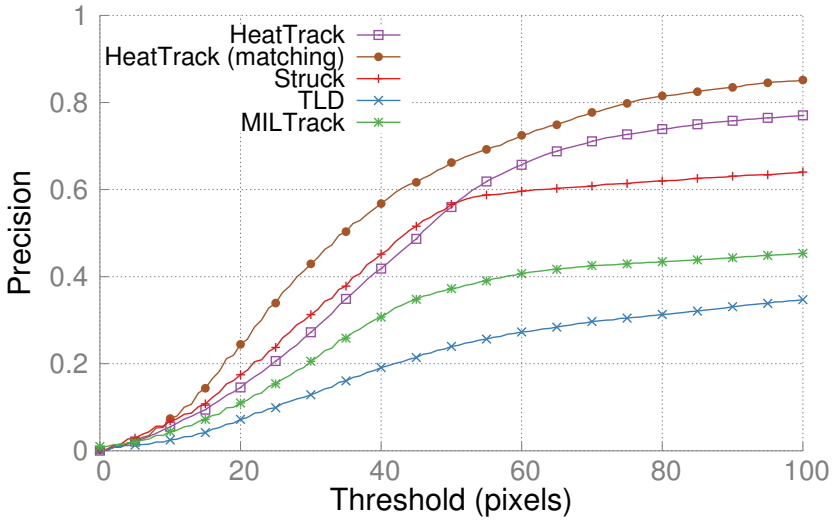



Figure 4.3: Tracker comparison. This shows, for a given threshold in pixels (horizontal axis), what proportion of frames had tracking error which was less than that threshold. Tracking error is the Euclidean distance between the centres of the estimated and ground truth bounding boxes. The average ground truth labelling error in labelling is 6 pixels.

the art trackers operating in the visible light domain only. These are the trackers discussed in Section 4.1.2: Struck [58], MILTrack [12] and TLD [66].

Tracking error is the distance in pixels between the ground truth location of an object and the tracker estimate of where it is. For each video a ground truth file was created manually which contains the centre of the bounding box in every frame. During tracking, the tracker estimate is compared with the ground truth location for that frame and the Euclidean distance is saved to a file.

Results are shown for two implementations of HeatTrack: the basic version, which uses a homography computed in advance in order to match between infrared and visible light, and an enhanced version which uses the improved matching method detailed in Section 3.3.3 to match an image patch in the infrared with the visible light.

Figure 4.3 shows a tracking precision plot calculated over the entire set of videos. The mediocre performance of all trackers reflects the difficulty of tracking articulated

Table 4.1: Tracker evaluation in terms of average tracking error

| **Tracking algorithm** | Avg. tracking error (pixels) |
|---|---|
| Struck [58] | 154.25 |
| MILTrack [12] | 213.07 |
| TLD [66] | 266.14 |
| HeatTrack | 100.84 |

people in unsteady video footage. Of the three state of the art algorithms, Struck performs the best, having the highest percentage of frames for which the tracking error was relatively low. HeatTrack performs fairly similarly to Struck when it comes to tracking the person closely, but where Struck might lose the person completely and not recover, HeatTrack continues to track in the vicinity of the person. This is evidenced by the fact that HeatTrack has a 20% higher proportion of frames which have a tracking error of less than 100 pixels. As qualitative analysis will show, this improvement is down to the use of a Kalman filter and the infrared providing crucial information which enables tracking to continue where visible-band trackers fail.

Table 4.1 shows the average tracking error for all four trackers. The results follow the same trend as the graph in Figure 4.3 – HeatTrack has the lowest average tracking error, followed by Struck, with the other two performing particularly badly. It should be noted however that the average figures are less meaningful than what the plot shows because they are influenced by large outliers when the tracker loses the person.

The performance of Struck, TLD and MILTrack is highly dependent on the initial bounding box on which the tracker was initialised. Each of these tracking algorithms is trying to learn a model of the appearance of the person in order to distinguish it from the background; if this covers the person precisely and does not include part of the background then tracking is most likely to succeed. All trackers are given the exact same bounding box – the projection of the initial infrared bounding box into the visible light image – but Struck is much better able to deal with inaccuracies in the initial bounding box than is TLD or MILTrack. The different ways of segmenting the person in the infrared, such as LBP or HOG, tend to produce a bounding box which includes some of the background. These detectors do not tend to give a precise

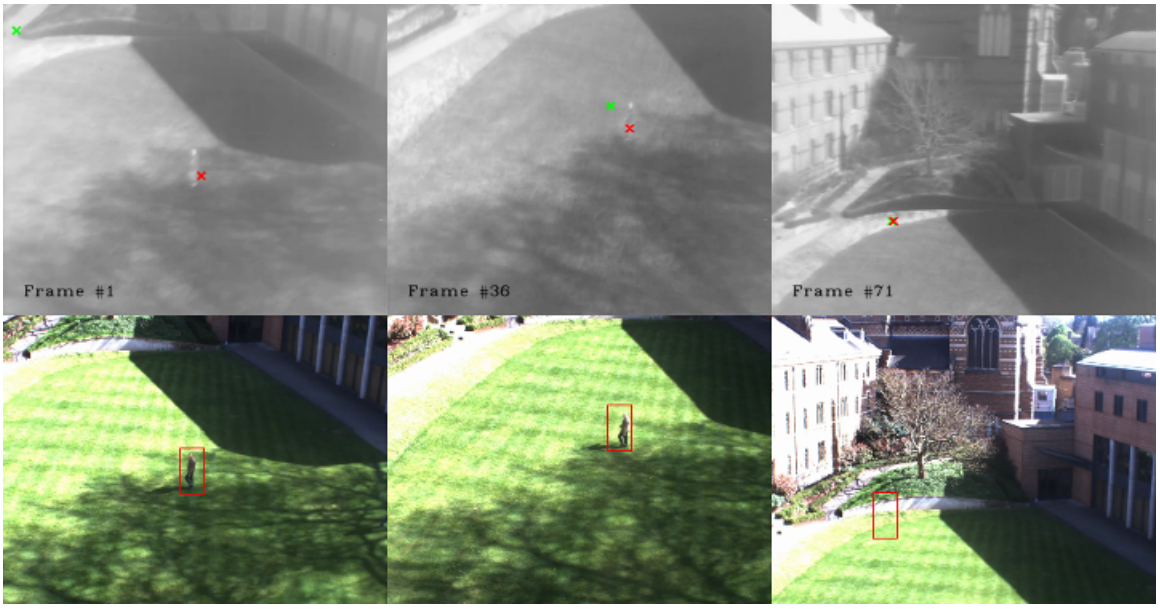



Figure 4.4: Frame grabs from tracking example A

segmentation of the person. In the case of thresholding, not all of the person may be segmented out. Regardless of which method is chosen to identify the person in infrared, it is rarely the case that the initial bounding box given to the tracker precisely covers the person and none of the background.

### 4.3.2 Why HeatTrack Can Fail

An analysis of individual tracking sequences yields insights into when HeatTrack can fail. This section looks in detail at some of the videos where this happens.

#### Video A (Figure 4.4)

Early on in the video, the camera field of view shifts significantly and the person goes out of view for approximately 17 frames. When the person reappears, they have moved into the hotter part of the scene and hence are not detectable in infrared. The visible light tracker fails to pick up the person when they reappear, so HeatTrack now relies on the prediction of the Kalman filter, but because the person moved out of the field of view early on, the Kalman filter did not have enough time to converge on an accurate velocity estimate. The full video is shown in Appendix A.10.

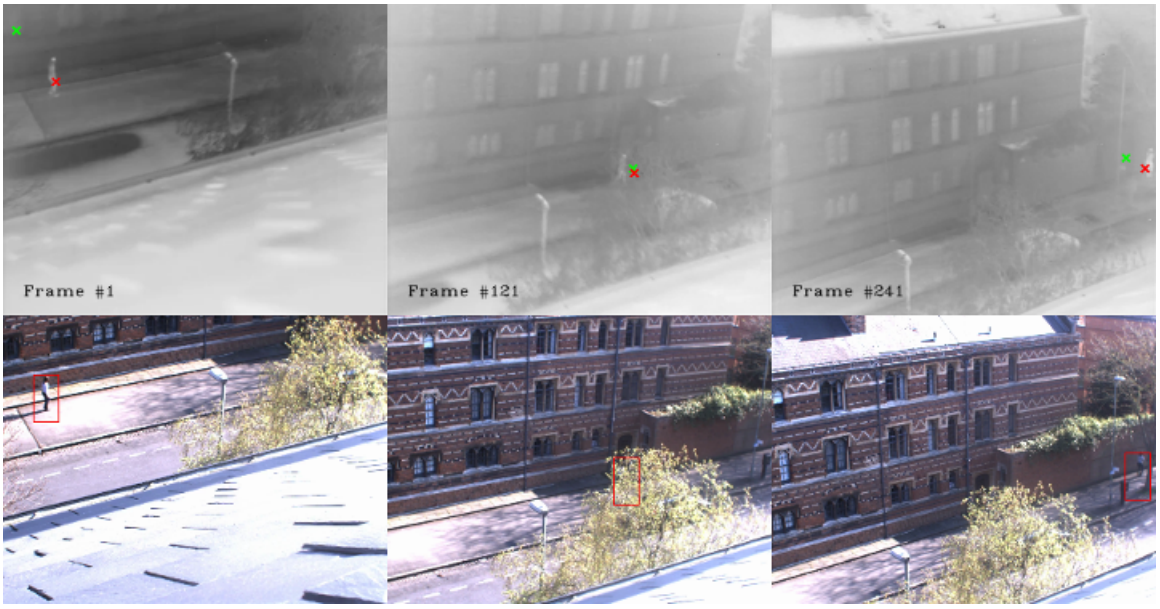



Figure 4.5: Frame grabs from tracking example B

### Video B (Figure 4.5)

The person disappears behind the leaves of a tree for approximately 4 seconds. The person is still picked up in infrared briefly, as the foliage is sparse. After a couple of frames, however, there are no longer any infrared measurements. HeatTrack must now rely on the Kalman filter prediction. Depending on the value which has been set for the measurement and process noise covariances $Q$ and $R$ of the Kalman filter, it may have converged on a good velocity estimate, in which case it follows the person closely while they are not visible. In this particular case, the values chosen meant that tracking succeeded. However, the $Q$ and $R$ that work for this video may not work for other videos, and vice versa. With a lower value for the measurement noise covariance, the filter will follow the measurements very closely while the person is visible, but when they are no longer visible, the velocity estimate is not good enough to continue following them behind the tree. If a higher value for measurement noise covariance is chosen, the filter can continue tracking the person behind the tree on the basis of its velocity estimate, but the downside to this is that the tracking error for the first $n$ frames of the video is quite high, because the filter does not trust the measurements as much. The full video is shown in Appendix A.14.

This illustrates the trade off between good overall tracking performance and the behaviour of the tracker when the person becomes occluded. To achieve a low overall tracking error requires that the Kalman filter follow the measurements very closely, but this comes at the expense of poor tracking when the person becomes occluded. A different set of values for $Q$ and $R$ is required if the tracker is expected to continue tracking under occlusion.

**Video C (Figure 4.6)**

A person is walking along a footpath. The camera moves abruptly and the infrared image becomes blurred. The person is detected in infrared and the system attempts to find the corresponding match in the visible light image, by searching for the image patch which matches the HOG descriptor the best. Unfortunately, the estimate of the corresponding location in the visible light image is wrong, presumably because the blur makes it difficult to match shapes accurately. This patch is then compared with the saved image patch of the person, produces a poor match, and consequently, the infrared measurement is not used. The system checks the output of the visible light tracker, and this is also wrong, also because of the blur. Unfortunately, the system cannot tell that this is wrong, because it passes the histogram match test. The full video is shown in Appendix A.16.

This illustrates two key issues which are common to many of the videos where tracking fails: (i) the shortcomings of matching the infrared to the visible light image using HOG descriptors: if the object has a sharp outline in the infrared, then HOG matching tends to work well, otherwise there can be multiple high scoring matches in the visible light image, and (ii) the difficulty in determining if tracking has failed in the visible light image.

**Video D (Figure 4.7)**

This video is an example of where an incorrect mapping between infrared and visible light images will result in tracking failure. The tracker is initialised with a bounding box in the infrared and this is then projected into the visible light. This video shows

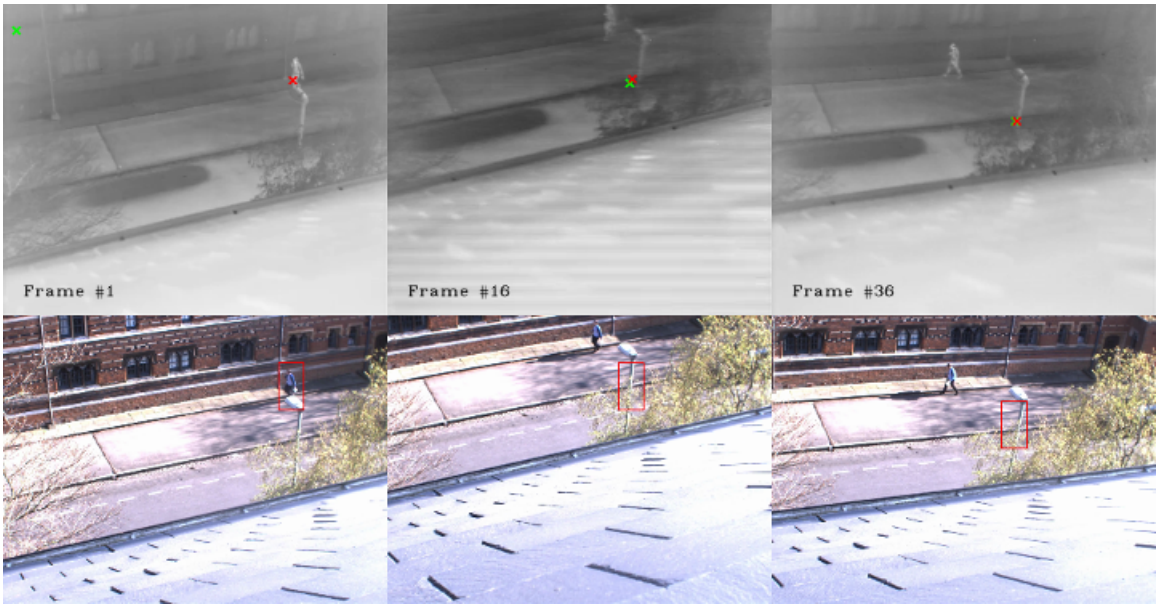



Figure 4.6: Frame grabs from tracking example C

that the mapping is incorrect, and the visible light tracker gets initialised on a section of the wall behind the man. The weather conditions on this day were such that there were very few infrared measurements, so HeatTrack was relying solely on the visible light tracker. The full video is shown in Appendix A.38.

### Summary

In summary, the most common reasons a tracker is likely to fail are:

- **Incorrect data association**. This can happen if the tracker associates the wrong infrared bounding box with the person, if it has the best histogram score for example, or if it takes the closest bounding box to the previous tracker estimate, which can happen when two people overlap. Incorrect data association can also happen if there are no infrared measurements and the system trusts the visible light tracker when it is wrong – again due to a good histogram score. In that case the tracker should simply use the Kalman filter prediction instead of incorporating any measurements.

- **Incorrect mapping between infrared and visible light**. Either of the two methods for mapping from infrared to visible light – homography or HOG descriptor


Figure 4.7: Frame grabs from tracking example D

matching – can give erroneous results. The method of matching HOG descriptors is particularly susceptible to failure if the infrared signature of the person does not have much contrast against the background, or if the person is in very poor illumination in the visible light. In both cases the distribution of image gradients around the person may not be similar in both images, and this confuses the HOG matching method.

- **Lack of measurements early on**. If the Kalman filter ceases receiving measurements from both modalities early on during tracking (within the first 20 frames, say), then relying on its prediction only will lead to errors, because the filter has not had enough time to converge on the true velocity.

The next section examines in detail some of the situations in which HeatTrack outperforms the best of the visible light trackers.

### 4.3.3 Where Infrared Improves Tracking

Figure 4.3 shows that HeatTrack outperforms Struck in terms of proportion of frames having a low tracking error. The difference is accounted for by the situations in which infrared provides better information than the visible light. Figure 4.8 shows tracking

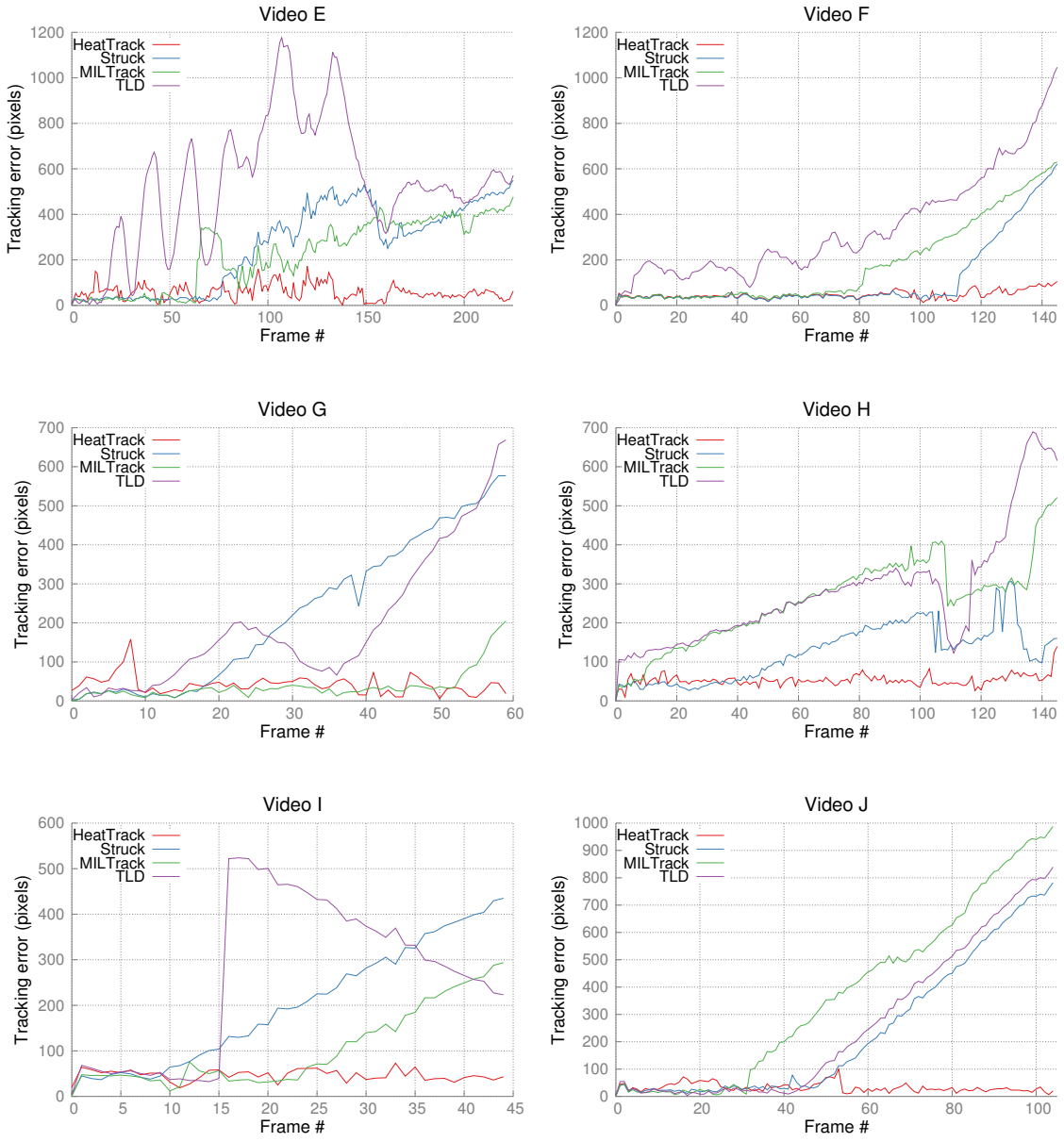



Figure 4.8: Tracker error plots for each of the videos described in the text. Tracker error is the distance in pixels (in the visible image) between the centre of the tracker bounding box and the ground truth location of the person.

error plots for six videos in which HeatTrack outperformed Struck by a large margin. Given the image resolution and the approximate size of a person in these images, if we want the tracker bounding box to overlap with ground truth by at least 50%, then a good tracker would be within approximately 80 pixels of ground truth (doing rough calculations). These calculations are done in the visible image. Each of the six videos

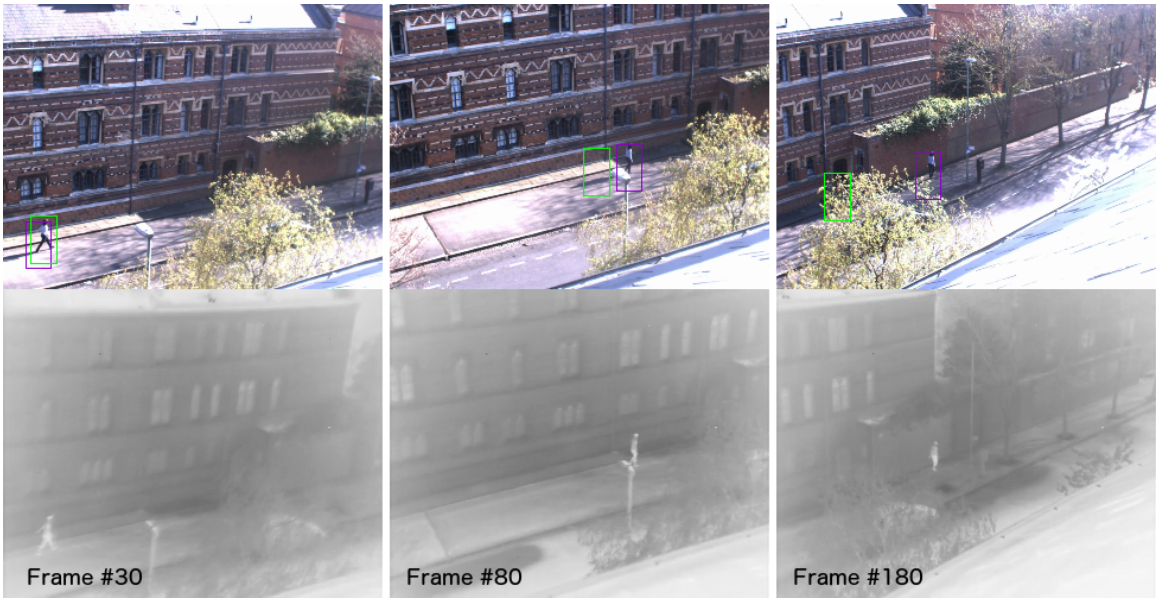



Figure 4.9: Tracking sequences. Purple bounding box is the estimate of HeatTrack. Green bounding box is the estimate of the Struck tracker.

is now examined in detail.

**Video E (Figure 4.9)**

In this clip a person is walking along a footpath and then goes behind a lamppost briefly, then re-emerges, and then goes behind a tree for a couple of seconds before re-emerging again. As soon as the person goes behind the lamppost the Struck tracker loses it and fails to pick it up again. The person has quite a distinctive infrared signature, and this is detected by the local binary pattern detector. HeatTrack continues to track the woman with the aid of infrared measurements. By this stage (80 frames in) the Kalman filter has converged on a reasonably accurate velocity estimate so that, when the women goes behind the tree and there are no longer any infrared measurements, the filter is able to predict the location of the woman as she is occluded. When she re-emerges, there is an infrared measurement which is within the covariance ellipse of the Kalman filter, and tracking continues. The full video is shown in Appendix A.14.


Figure 4.10: Tracking sequences. Purple bounding box is the estimate of HeatTrack. The green bounding box is the estimate of the Struck tracker.

### Video F (Figure 4.10)

In this clip there are two people walking in a grassy area who cross paths after about 110 frames. Both are being tracked with separate Kalman filters. Notice the difference in the infrared between the area of grass in the shade and the area which is under strong sunlight. The background is so hot as to make the person on the left appear indistinct. As a consequence, there are no infrared measurements for the person on the left, at least initially. HeatTrack is therefore relying solely on the visible light tracker which performs very well in this situation. As soon as the person on the left moves into the shade, they are picked up in the infrared image and this is used as a measurement in the Kalman filter. When the two people cross paths the Struck tracker gets confused and switches to tracking the other person. HeatTrack senses that there is an overlap and momentarily ceases to incorporate any visual measurements, relying instead on the filter prediction. When the two people become distinct again the Kalman filter starts incorporating the infrared measurements. The full video is shown in Appendix A.9.

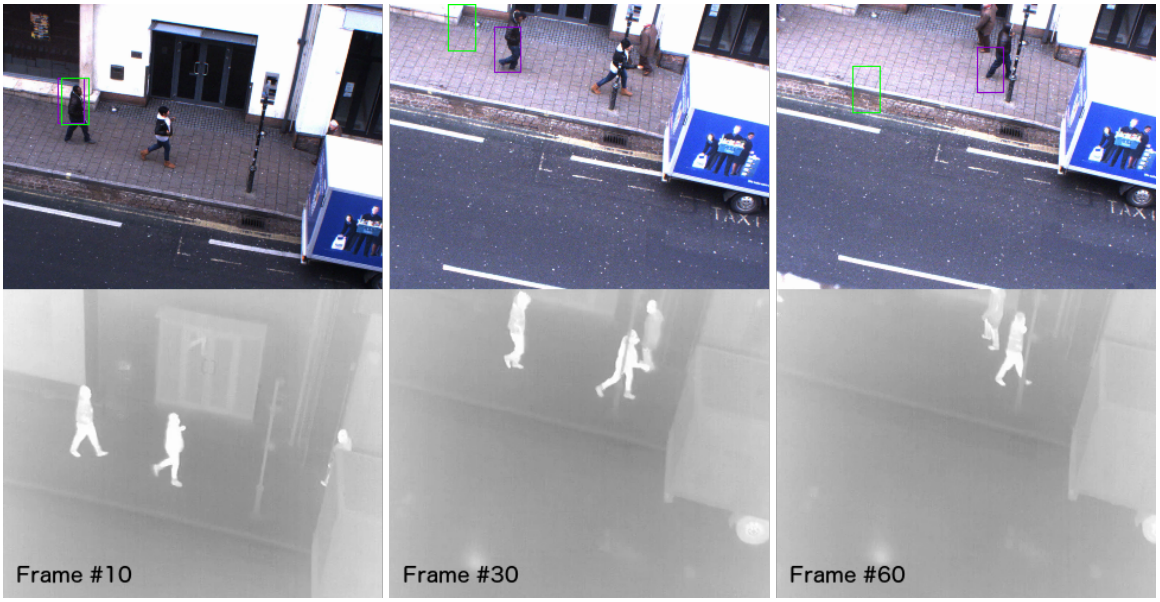



Figure 4.11: Tracking sequences. Purple bounding box is the estimate of HeatTrack. The green bounding box is the estimate of the Struck tracker.

### Video G (Figure 4.11)

In this clip the camera was quite a bit closer to the people than in the other video sequences, and therefore the person occupies a larger part of the image. This is a common situation in which the other trackers fail. A person is walking along the footpath. The initial bounding box supplied to the tracker included part of the dark background on the left. As soon as the person passes the pillar on the left (and the background changes) the Struck tracker fails, presumably because it has not had enough time to learn an accurate representation of the appearance. This highlights another issue with the tracking-by-detection approach – the tracker is more susceptible to failure early on than later, as it has not been able to learn all the sources of variation in the appearance of the person. This scene is an example of where HeatTrack performs particularly well. In this scene the person's infrared signature is highly distinctive given how close the camera is, and the video was taken on a cold day. As a consequence HeatTrack is able to track solely with the infrared measurements. The full video is shown in Appendix **??**.

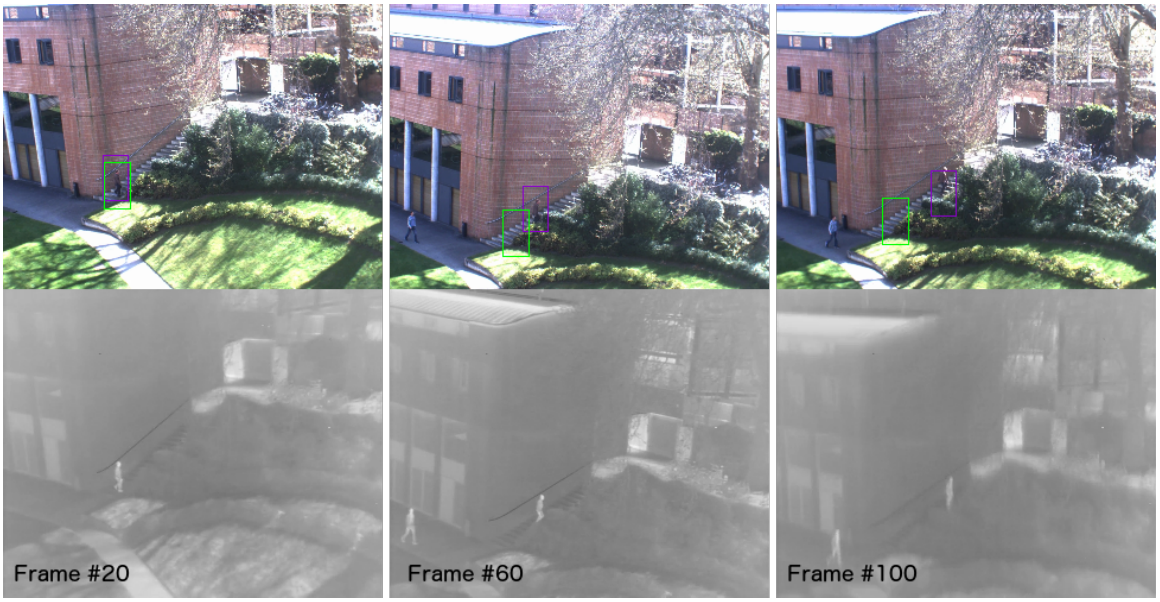



Figure 4.12: Tracking sequences. Purple bounding box is the estimate of HeatTrack. The green bounding box is the estimate of the Struck tracker.

### Video H (Figure 4.12)

In this clip there is a person walking up a set of steps wearing clothing of a similar colour to the wall behind them. Given the small size of the man in the image and the similar background, the Struck tracker fails almost straight away. In the infrared, however, the person shows up very clearly. The person has a very distinct silhouette, and this is used as the sole input to the Kalman filter. By the time the person reaches the top of the steps and goes behind some foliage, the Kalman filter has converged to a velocity estimate so that, in the absence of measurements from either infrared or visible light, the prediction of the Kalman filter keeps within a reasonably small distance of ground truth. The full video is shown in Appendix A.12.

### Video I (Figure 4.13)

In this clip a person is walking along a footpath. Both trackers continue to track him as he approaches the tree on the left. As soon as he becomes partially occluded by the branches of the tree, the Struck tracker loses him. The sparsity of the branches on the tree mean that the person is still detectable in infrared, and this continues to provide measurements periodically. By this stage, 100 frames in, the Kalman filter

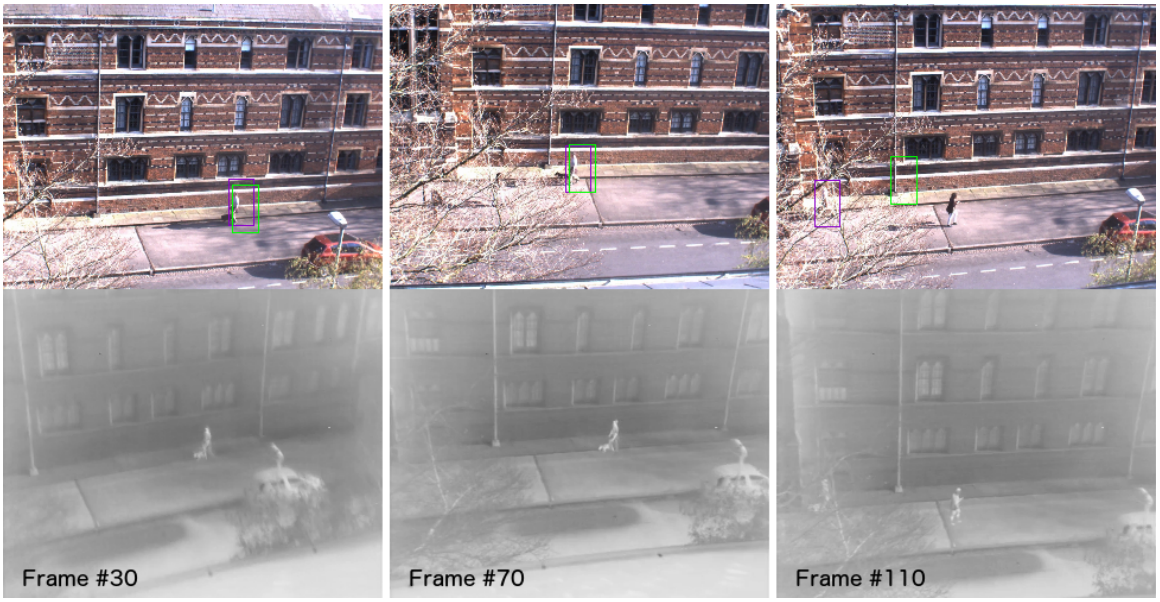



Figure 4.13: Tracking sequences. Purple bounding box is the estimate of HeatTrack. The green bounding box is the estimate of the Struck tracker.

has converged to an accurate estimate of the velocity of the person, and this is used when there are no longer any measurements from infrared or visible light. This clip also highlights the advantage of using a filter such as a Kalman filter – when the measurements stop, the filter can continue tracking reasonably accurately. The full video is shown in Appendix A.19.

### Video J (Figure 4.14)

In this clip a person starts being tracked as he is walking along the footpath. He then picks up pace and runs into the grass. As he does so, the Struck tracker loses him because his background changes early on. Here again the person has a sharp infrared signature, especially so because he is in the shade, and this provides the Kalman filter with consistent measurements which enable it to continue tracking for the remainder of the video. The full video is shown in Appendix A.7.

### Summary

It does not come as a surprise that HeatTrack outperforms the other trackers when the person has a sharp infrared signature, as in that case it can rely solely on infrared

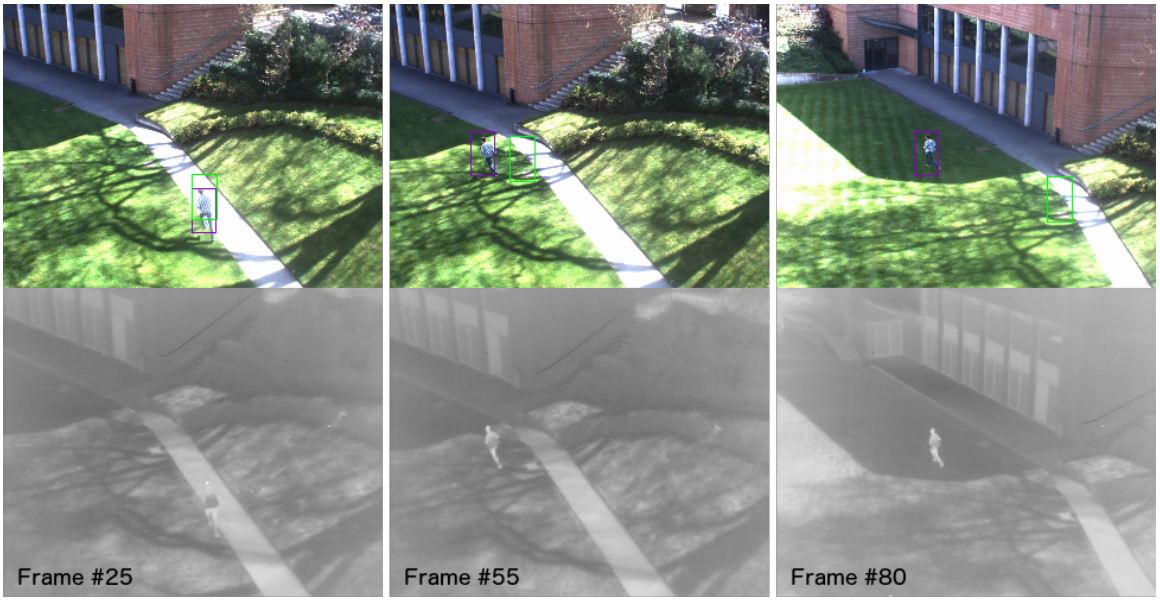



Figure 4.14: Tracking sequences. Purple bounding box is estimate of HeatTrack. The green bounding box is the estimate of the Struck tracker.

measurements and there is less room for confusion than in the visible light. However, what is surprising is that the conditions under which a person will blend into the background are not necessary high outdoor temperatures. High humidity levels will tend to make the person appear indistinct, especially if they are far away. The camera essentially has to 'see' through water in the air.

### 4.3.4 Tuning the Kalman Filter

Recall from Section 2.3.2.1 the two parameters $Q$ and $R$ which are the measurement and process noise covariance of the Kalman filter, respectively. The measurement noise covariance $Q$ reflects how much we trust the measurement process. The process noise covariance $R$ reflects how much we trust the process model, in this case, how much we think the person follows a constant velocity model. The values chosen for these have a significant effect on the success of tracking, and can be tweaked depending on how well one wants the tracker to perform under occlusion versus how closely one wants the tracker to follow the object when there are measurements. The main sources of noise are expected to result from the approximation error inherent in converting image coordinates between infrared and visible light, and the error in

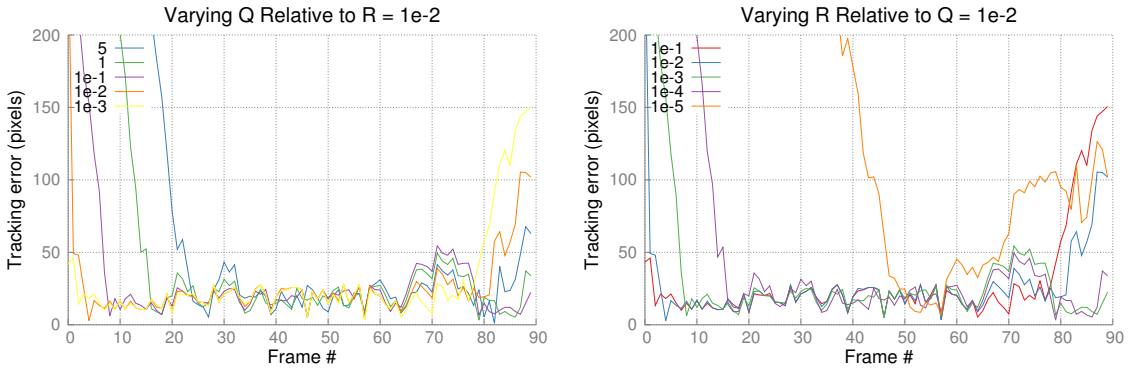



Figure 4.15: Effect of varying $Q$ and $R$ for a typical video with occlusion (from frame 60 onwards). In general, the longer it takes for the Kalman filter to converge to the true posterior, the better it is at tracking when there is occlusion.

the homographies estimated between each pair of visible light frames. It is difficult to estimate exactly what the values should be for a particular system. There is surprisingly little information available in the literature on appropriate values of $Q$ and $R$ to track people in images. This set of experiments was done in an attempt to estimate $Q$ and $R$ empirically.

Figure 4.15 shows a tracking error plot for a typical video where occlusion happened at around frame 60. On the left is a plot of different tracking errors got by keeping $R$ fixed and varying $Q$. Similarly, on the right is a plot of different tracking errors produced by keeping $R$ fixed (left), and varying $Q$. If $Q$ is low, $K$ (the Kalman gain) is large and therefore the measurements are trusted more than the process model. The filter is able to deal with large fluctuations in the movement of the person because it relies more on measurements. With a higher $Q$, the Kalman gain is lower which means that the state does not change much from frame to frame and it can take a while for the estimated state to converge to the true state. This is because with larger Q, less credence is given to the measurements.

The importance of choosing the right $Q$ and $R$ is highlighted when considering what happens when the person becomes occluded. In this case there are no measurements, so the filter is relying solely on its process model. As Figure 4.15 shows, if process noise covariance $R$ is high, then the measurements are trusted more than

the process model, but when there are no longer any measurements, the estimated state moves further and further away from the true state. Looking at the plot for measurement noise covariance $Q$, a higher $Q$ means the filter takes longer to converge to the true state, because measurements are trusted less, but when there are no longer any measurements, the process model can continue tracking the person because it has converged on an accurate velocity estimate.

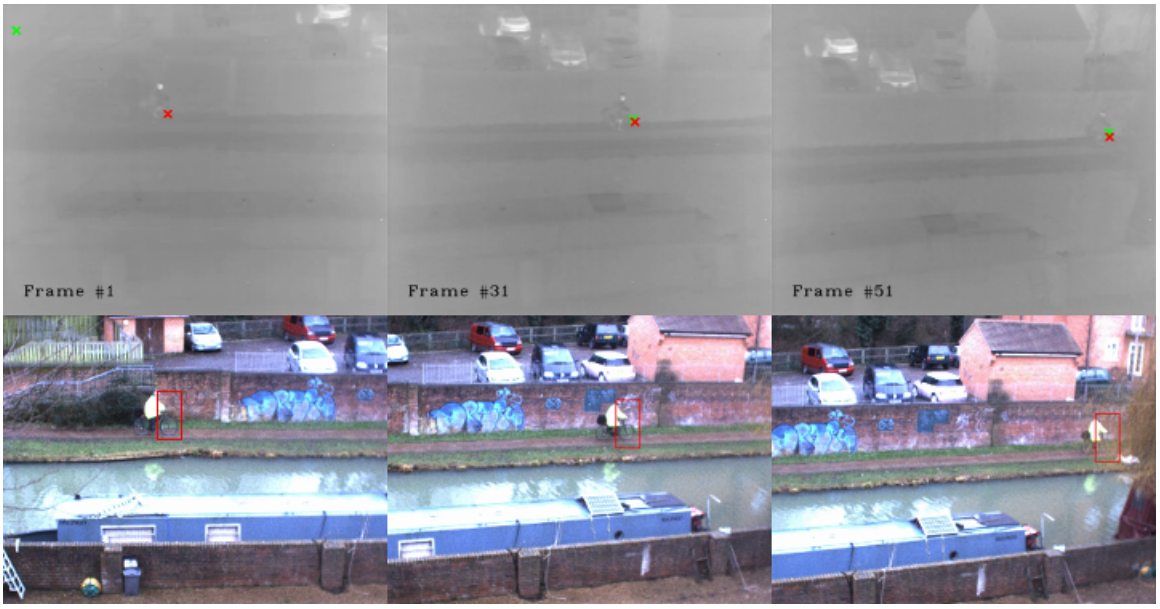
From the experiments it appears that the constant velocity model is valid, and that therefore small values of $R$ ($\sim 1.0 \times 10^{-2}$) are suitable. The measurement process is more noisy – the main sources of error arising from homography computation and conversion between visible light and infrared image coordinates. Therefore higher values of $Q$ (5 to 10 pixels) give better results. Note also that values of $Q$ and $R$ which are set too high means that the covariance around the state will be large and therefore not of much use in the data association step when deciding which measurements are more plausible than others.

### 4.3.5 How Lighting Affects Tracking

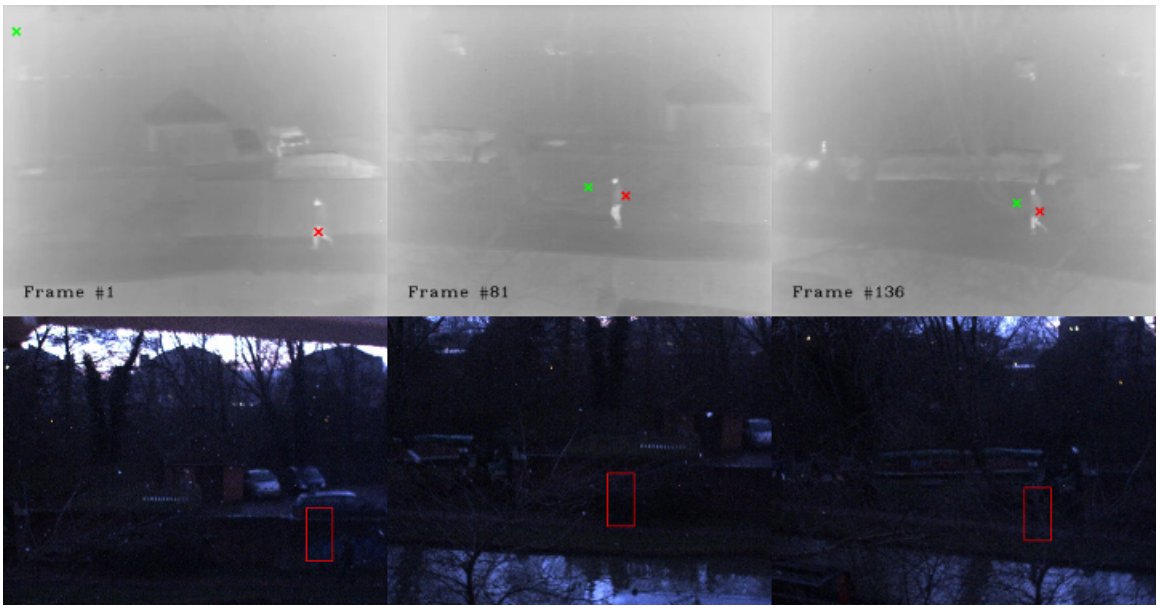
This set of experiments looks at the effect which outdoor lighting levels have on tracking. Intuitively, low lighting levels should have no effect on tracking if the person shows up clearly in the infrared, and this is indeed what happens with HeatTrack. Figures 4.16 and 4.17 show three videos taken at the exact same location but at different times of day: morning, dusk and night. These times were chosen specifically because there is a very noticeable difference in the lighting levels between these times (full video summaries are shown in Appendix A).

In the first video it was 2°C outside but the person is seemingly wearing very heavy clothing because they are barely perceptible in the infrared. HeatTrack is entirely dependent on the visible light tracker which works very well in this case due to the lighting. (The fact that it has latched onto the wheel of the bicycle and not the person is due to incorrect matching resulting from a lack of clarity in the initial infrared box provided). The full video is shown in Appendix A.20.

In the second video, taken at dusk, it is very dark outside and the person is




(a) Morning video




(b) Dusk video

Figure 4.16: An example of tracking in the morning and at dusk

hardly visible in the visible light image, even to a human observer. However, it is -4°C, and therefore not a surprise that the infrared signature is very clear and tracking is successful. Note that even though it is dark, there is still enough detail in the visible light images to be able to compute accurate motion estimates between consecutive frames. The full video is shown in Appendix A.39.

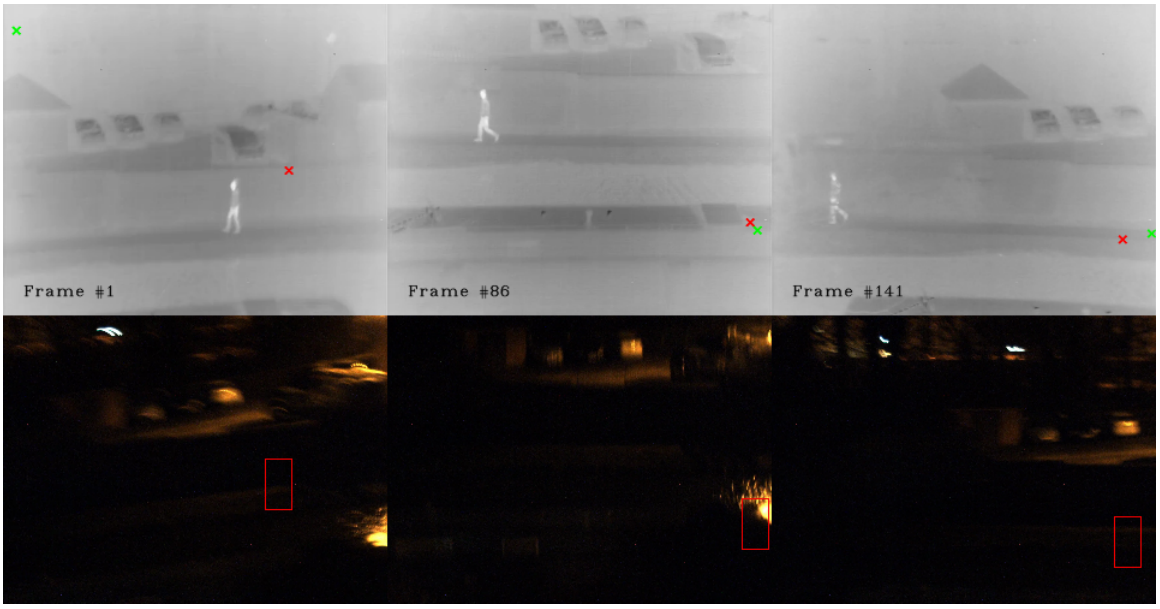



Figure 4.17: An example of tracking at night

In the last video, taken at night, the visible light camera shows very little apart from black. Although the infrared image shows the person very clearly, HeatTrack fails. This is a surprising result but the explanation is straightforward: without any detail in the visible light image it is impossible to compute the homographies which are necessary to cancel out the camera motion. By default, the identity matrix is used as the homography for every frame – which would apply if the camera were stationary – and since this is used to convert coordinates from the current frame back to the reference frame of the Kalman filter, the results are inaccurate. The full video is shown in Appendix A.45.

We can deduce that HeatTrack will work in the dark if either (i) the camera is stationary, in which case there is no need to estimate camera motion from the visible band, (ii) there is sufficient artificial lighting to light up the scene e.g. street lamps as in the case of Figure A.48, or (iii) there is some other way of estimating the camera motion between image frames. One might ask why the infrared image can not be used to estimate the inter-frame homographies. Computing homographies between pairs of infrared images tends to give bad results due to excessive blur and the ambiguity of matching when there are too many similar points. While there may

be accurate results every couple of frames, in general the system is not robust enough to use as a consistent estimate of camera motion. An alternative to looking at image displacements, if the images were coming from a UAV, is to use the pitch/roll/yaw information coming from the onboard Inertial Measurement Unit (IMU) in order to gauge how much the camera has moved between frames. This would require synchronising the IMU information with the two cameras (specifically the infrared camera in this case) and reduce the dependency on the visible band in situations where it is not helpful. The IMU itself is prone to noisy measurements; in practice the measurements from different sensors including the IMU are fused in a Kalman filter which helps to smooth the noise.

### 4.3.6 Tracking in Different Image Resolutions

This section examines the effect which changing the resolution of the visible light image has on the success of tracking. For the initial detection of people with a part-based detection algorithm, higher resolutions are better. However, in the case of tracking it would appear that having a high resolution image is detrimental to tracking. Figure 4.18 shows the effect which downsampling the original $1280 \times 960$ visible light image by a factor of 2, 4 and 8 has on the success of tracking over all the videos. The best results were obtained using a resolution of $320 \times 240$, that is, a quarter the original size. Note that although tracking occurs in the reduced resolution, the result is then projected back into the original $1280 \times 960$ image to make the results comparable across different resolutions.

Tracking in the original high resolution image gives results as poor as if the image is down-sampled so much as to make the person unrecognisable. At high resolutions, the person appears in quite a bit of detail and their appearance can change significantly from frame to frame. This level of detail confuses the tracker, which is trying to learn a common appearance of the person as the video goes on.

If the video is down-sampled by a factor of 2, that is, the image is first blurred and then every second pixel is removed in order to create an image of half the size, this improves tracking results. The tracking is even better if the image is reduced by

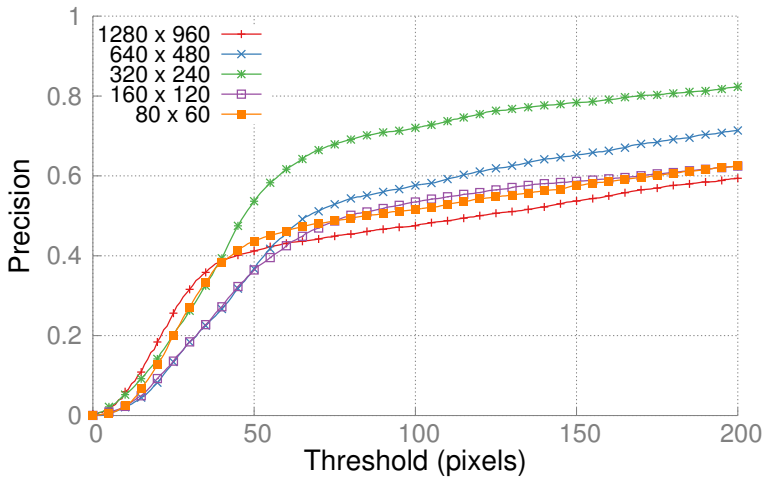



Figure 4.18: Effect of different visible light resolutions on tracking. This shows, for a given threshold in pixels (horizontal axis), what proportion of frames had tracking error which was less than that threshold. Tracking error is the Euclidean distance between the centres of the estimated and ground truth bounding boxes. Although tracking occurs in the reduced resolution, the result is then projected back into the original $1280 \times 960$ image to make the results comparable across different resolutions.

a factor of 4, to a size of $320 \times 240$. At that resolution, the appearance of a person is characterised mainly by colour and overall shape, and the trackers are better able to learn the general appearance of the person. While a resolution of $320 \times 240$ gives a noticeable improvement in tracking compared to the other image resolutions, below this resolution, there is no significant degradation in tracking quality, because the infrared takes over.

This suggests that the optimal image resolution for tracking in these videos is somewhere between $320 \times 240$ and $640 \times 480$. For this set of videos, this means the person has a bounding box of roughly $25 \times 50$ pixels. At that size, there is still enough colour information to track the person but the effect of background clutter which usually causes the tracker to fail is minimised. Tracking-by-detection algorithms work by learning the general appearance of an object, and this is best done by blurring or reducing the size of the image.

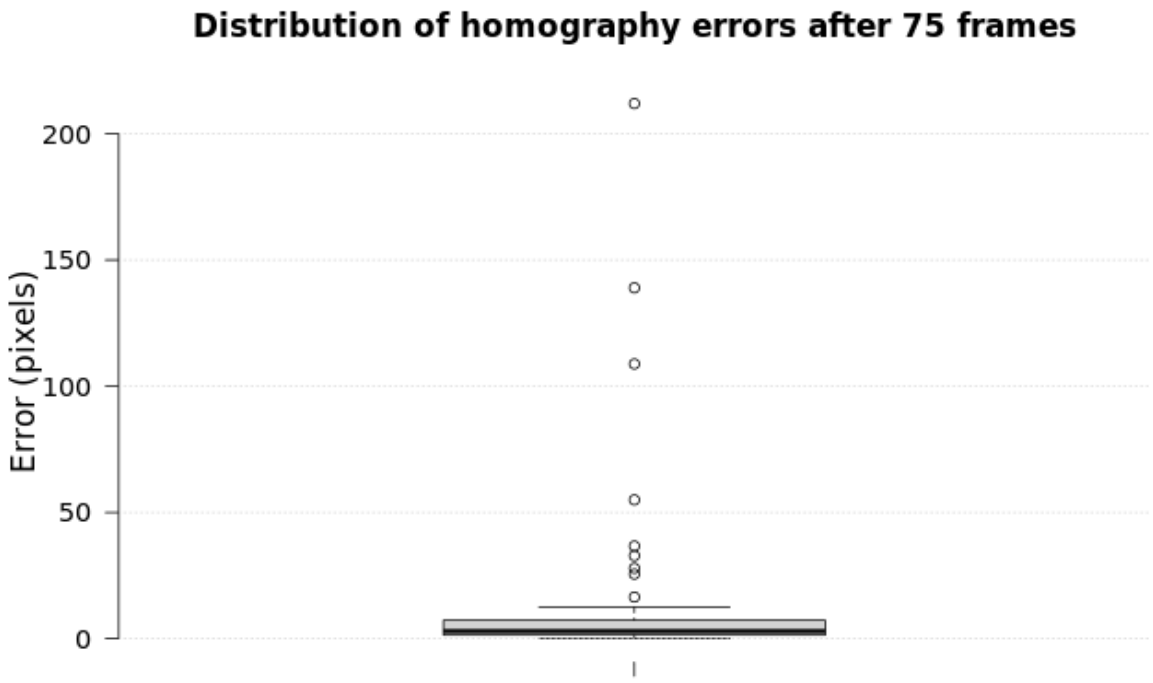



Figure 4.19: Distribution of homography errors over all videos after 75 frames.

### 4.3.7 Homographies as an Estimate of Camera Motion

Given that the scenes in our dataset are not planar, there was a suspicion that a 2D homography would not be a valid model of camera motion. Furthermore, for each of the videos recorded, there was a significant amount of camera shake induced in order to simulate the type of footage one would expect to get from a UAV. This section examines how valid the homography model is for approximating camera motion. To test it, a random pixel location is selected from a frame near the end of a video and reprojected back to the initial frame using the intermediate homographies as the transformation matrix. If the homographies are accurate, this point, when reprojected to the first frame, should represent the exact same scene point. We did this for each of the 47 videos recorded. Figure 4.19 shows the distribution of the 47 error values, where the error is calculated as the distance between the reprojected point and where it should be.

This shows that the majority of the videos had a homography error of less than around 10 pixels after 75 frames, which suggests that for the majority of the videos

Figure 4.20: Examples of where the homography model is not valid. Left: The roof in the foreground is much closer to the camera than the rest of the scene. Right: The boat and the foliage in the foreground result also result in an inaccurate homography being computed between pairs of consecutive frames.

the homography model is indeed valid. The saving grace is that for most of the videos, the effects of parallax are not as severe as if the videos had been recorded at a closer range. However, nearly 20% of the videos had an error of more than 20 pixels, and this goes some way to explain the poor tracking performance shown in Figure 4.3. This error, combined with the error in mapping points from visible light to infrared, resulted in the tracker estimates being way off for these videos. Figure 4.20 shows some examples of where the homography estimates were incorrect. In both examples shown, the scene is very clearly not planar. In Figure 4.20 (left), there is a roof near to the camera taking up a significant chunk of the image; in Figure 4.20 (right), the boat and the tree foliage in the foreground result in an incorrect homography estimate which is then propagated through the rest of the video.

Using homographies as a motion estimate works as long as the homography can be accurately computed, and there are cases when the homography cannot be computed accurately, such as at night time or when there is too much camera shake resulting in blurry images. The homographies are computed from consecutive pairs of visible-band images; one might ask why not use the infrared instead? If the infrared image is sharp and full of detail, then quite accurate homographies can be computed. However, given the nature of the infrared imaging process, a lot of the images tend to have

Table 4.2: Computation times for tracking

IR Segmentation (584×474 px)

| **Segmentation method** | Computation time (ms) | ms per 1000 px |
|---|---|---|
| Thresholding & connected components | 3 | 0.01 |
| HOG detector | 482 | 1.74 |
| LBP detector | 63 | 0.23 |

Visible band processing (1280×960 px)

| **Visible light processing** | Computation time (ms) | ms per 1000 px |
|---|---|---|
| Homography computation | 74 | 0.06 |
| Struck [58] | 26 | 0.02 |
| MILTrack [12] | 136 | 0.11 |
| TLD [66] | 43 | 0.03 |

some blur (especially if the camera is moving) and there tends not to be enough stable key points which are repeatable across images. This applies to the standard SIFT matching algorithm of [81], though of course it would be possible to compute a coarse homography estimate possibly using more context around individual key points. Given the results in [99], which is able to compute a coarse depth map by matching HOG descriptors across stereo infrared pairs, it is likely that one could also compute homographies using a similar optimisation procedure. This is a possible area for future improvement.

Another possibility is to track the homographies over time using a Kalman filter, and this would provide robustness in the case of spurious inaccuracies which could throw off the whole computed trajectory. Alternatively, given a history of the last $n$ frames, and assuming that there is some loop closure, i.e. seeing the same scene again, one could correct for errors by minimising the distance between scene points imaged in the current frame and their estimated location based on the homography trajectory.

## 4.4 Computation Times

Table 4.2 shows the computation times to track a person using a single core on a Macbook Pro 2.5 GHz Intel Core i5 with 4 GB of RAM and no GPU. The tracking code is currently unoptimised and therefore the figures shown are illustrative estimates of computational performance. To recap from Figure 4.1 in Section 4.1, the image processing pipeline of HeatTrack is:

1. Compute the homography between the last visible light frame and the current frame in order to estimate the camera motion. This is used to project current frame coordinates into the coordinate system of the Kalman filter.

2. Run the segmentation method on the infrared image. This is to find potential candidate bounding boxes which may correspond to the person being tracked.

3. Do data association in order to associate the correct infrared bounding box with the person being tracked.

4. If there is no infrared measurement, run the visible light tracker.

If the infrared is guiding the tracking i.e. the person shows up clearly in the infrared, then the computation time is approximately $74 + 63 = 137$ ms per frame, that is, the time to compute the homography plus the infrared segmentation, which in the current implementation means running a LBP detector over the image. If there is no useful information in the infrared, then the visible light image must be searched, and this requires the previous steps plus executing the visible light tracker. Assuming that Struck is used, then the total computation time if there are no infrared measurements is $74 + 63 + 26 = 163$ ms per frame. The bottleneck here is running the LBP detector, which as mentioned in the previous chapter, is parallelisable, and therefore, possible to run at standard frame rates.

## 4.5 Summary of Results

This section summarises the results that were observed in the experiments presented in this chapter.

**Correct data association is crucial**

HeatTrack is successful as long as the system does correct data association with measurements from either modality. A classic example is when a person becomes occluded and the infrared ceases to give measurements. The system must determine from the visible light image whether or not the person has become occluded or not. If it can be determined that tracking has failed in the visible light, then the system should use the Kalman filter prediction. Knowing whether to trust the infrared image is easier than for the visible light – there is either a white blob or there is not. Knowing whether to trust the visible light tracker currently boils down to determining what is a good score for the histogram match between the current hypothesis and the saved image patch. Knowing what this score should be determines whether the tracker can cope in the face of occlusion.

**Tune the Kalman filter according to requirements**

There is a trade-off between tracking accuracy in terms of pixel error and how well the tracker can continue to track during periods of occlusion. This is governed by the values chosen for the measurement and process noise covariances, $Q$ and $R$. With a higher $Q$, the Kalman gain is lower which means that the state does not change much from frame to frame and it can take a while for the estimated state to converge to the true state. This is because with larger $Q$, less credence is given to the measurements. If $R$ is high, then the measurements are followed more, but when there are no longer any measurements, the estimated state moves further and further away from the true state.

**The longer a person is successfully tracked, the longer they will be successfully tracked**

The longer tracking goes on successfully, the less likely it is to fail. This is because the more measurements the Kalman filter has received, the more likely it is to have converged to the correct velocity estimate, and this is useful during periods of occlusion. The visible light trackers are also more likely to continue tracking if they have successfully tracked an object for a while; this is because with more positive and negative training examples they have been able to learn a more accurate representation of the person which is robust to occlusion and changes in the background.

**The homography model is a valid model of camera motion for most of the videos**

Some of the tracking error results presented in this chapter can be attributed to inaccuracies in the homography model as an estimate of 2D camera motion between consecutive frames. Even if the tracker can determine correctly whether measurements can be trusted, and does data association correctly, the reported tracking error will be high if there is an error in the mapping between the current frame and the reference frame of the Kalman filter. The homography model is only valid as long as the scene is far enough away as to appear almost planar.

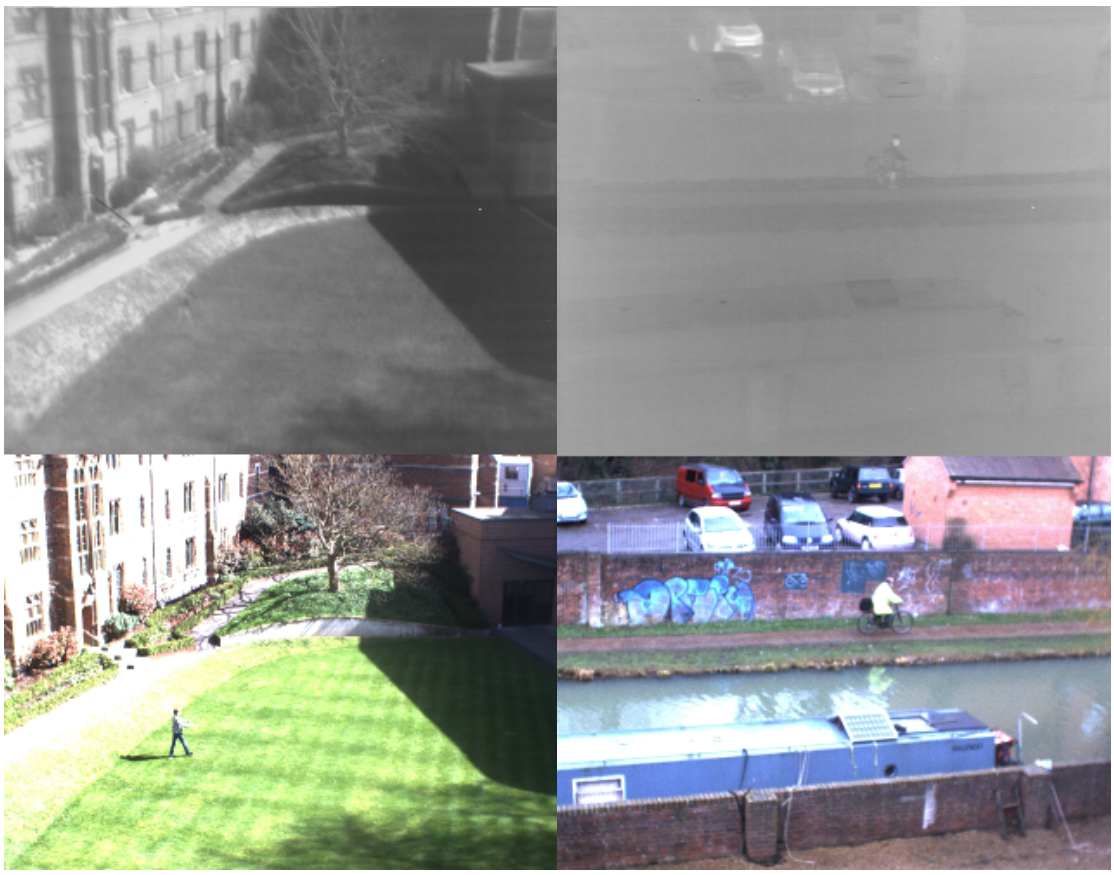
---

This concludes the explanation of the basic method used to track people using a combination of infrared and visual light imagery. As there is no prior published work on tracking people in aerial video footage by combining modalities, this work was compared against the three state of the art trackers operating on the visible band image only, namely [12], [66] and [58]. Overall, our method outperforms these trackers judging by the its lower average tracking error (Table 4.1) and its higher percentage of image frames for which the tracking error was under an acceptable threshold (Figure 4.3). Furthermore, we showed that our method outperforms the other trackers when the person becomes occluded (Figure 4.8) and provided an in-depth analysis with examples of where it performs well and where it fails.

# Chapter 5

# Discussion

This chapter discusses in more detail some of the main findings of the thesis, and delves into the issue of putting the system onboard a UAV, one of the potential applications of this thesis.



(a) (b)

Figure 5.1: Where infrared should not be trusted. (a) The person blends into a large hot region. (b) The person is wearing heavy clothing.

## 5.1 When to Trust One Modality over the Other

Given the complementary information often provided by the infrared and visible band modalities, a desirable capability would be to automatically know when to trust one modality over the other. Sometimes it is possible to do tracking in the infrared modality only; sometimes the infrared should be ignored altogether and the entire visible light image should be searched. Knowing that one modality could not be trusted would save needless computation time and prevent tracking errors arising from incorrect data association.

In the current implementation of HeatTrack, the infrared image is always searched first. The assumption is that it is easier to quickly filter out unlikely regions in the

infrared image than the visible band, and doing this will narrow down the search to the more likely regions of the visible light modality. It was an obvious approach to take at the outset, because one tends to think that the thermal image is less susceptible to confusion in person identification than the visible band. However, the analysis of Chapter 4 identified some situations in which the infrared should have been ignored and the entire visible light image searched instead. Take, for example, Figure 5.1a, which shows a person blending into the background in the infrared image on a hot day, or Figure 5.1b, which shows a situation where the person is barely visible in the infrared because of what they are wearing. Conversely, there were a couple of situations in which the visible light image should definitely have been ignored, such as when it was very dark outside. The circumstances under which a person shows up clearly in the thermal image is also dependent on the thermal sensitivity of the camera, that is, how much of a temperature difference in the scene the camera is able to discern, as discussed in Section 3.1.1. Is it possible to automatically know when to switch from one modality to the other from a quick analysis of the images?

**Temperature / Time of day**

Table 5.1 summarises the videos recorded for this thesis in terms of lighting level and temperature at the time of recording (with links to snapshots in Appendix A). From the experiments conducted it would appear that time of day is one of the most straight-forward indicators of when to trust one modality over the other. When it is very dark outside the visible light image should obviously not be trusted. Early mornings, evenings and nights tend to be cooler than daytime and these are the conditions under which a person shows up more clearly in the infrared. The particularly difficult videos to track (where the person blended in to the background in the infrared) were taken in mid-afternoon when the sun was beaming down directly on a large open grassy area. Previous work has attempted to deal with the issue of low-contrast thermal signatures against a warm background through the use of phase congruency maps [92] as discussed in Section 2.4.1.

Figure 5.2: Example of a large hot region in the infrared

### A large hot region in the infrared

A large hot region in the infrared image (of a scene a considerable distance away) usually suggests that it is hot outside. In warm/hot weather there is a likelihood the person may blend into the background. Large hot regions are easy to identify: a simple thresholding operation can do this in a single scan over the image, but it is not always the best indicator of whether the infrared should be trusted; in Figure 5.2 there is a large white blob in the foreground but the person in the background is still detectable with a classifier. A lot also depends on the scene. A grassy wilderness scenario will exhibit different thermal characteristics to a street scene with hot moving vehicles. Nonetheless, large hot regions should alert the system to the fact that the infrared may actually be camouflaging the person and that therefore it may not be as useful as it would be under cooler conditions.

### No infrared blobs of the right size

The infrared image is processed with a blob classifier in order to find candidate regions which should be subjected to further tests in the visible band. Currently, these are filtered based on their size given the height of the camera above the ground, similar to [51, 109]. However, this is not always a reliable indicator. Depending on the detection algorithm used, there might *always* be detections fired in the image. For example, the

classifiers trained for this DPhil were deliberately undertrained, that is, not trained with the lowest possible false positive rate in mind, because we were not aiming for precision detection in the infrared. The idea was to use these detections as initial candidates which would be subjected to more tests in the visible light. The result of under-training the classifiers is that there might always be false positive detections fired in the image.

In summary, although there are various ways one might think of deducing how useful a modality is going to be, there is no exact science to automatically determining this. Previous work such as [51] processes both modalities independently and then combines the results, avoiding the need to chose one modality over the other. Knowing the weather conditions at the time of recording seems to be the best indicator of all.

## 5.2 The Data Association Problem

One of the main failure modes of HeatTrack is incorrect data association with either infrared or visible light measurements, that is, given a hypothesised location in either image, does it correspond to the person being tracked or should it be ignored? As discussed in Section 4.1.3, the following criteria are taken into account when deciding whether a measurement is correct:

- size/aspect ratio
- proximity to previous estimate
- appearance match

The main issue in data association is determining what values to use for these different criteria, as so much depends on the scenario. Tracking someone in a grassy region is a lot easier than tracking someone against a complex background, and certain scores for an appearance match will be very different depending on the background.

Size is probably the easiest to filter on – blobs which are implausibly large can be discarded. The same cannot be said for blobs which are too small, because often only

Table 5.1: Summary of videos under different temperature and lighting conditions. Though it is tempting to look for a correlation between temperature/time of day and tracking performance, in reality these factors on their own are not enough to predict whether tracking will succeed. Rather, it comes down to whether the system does correct data association.

| | Daylight | Dusk | Night |
|---|---|---|---|
| <5 °C | A.20, A.46: In A.20, even though it is -4 °C, the person does not have a sharp infrared signature. In spite of this, tracking is successful because the person is clearly visible in the visible band. In A.46 the person has a very sharp infrared signature and this drives the tracking. | A.32, A.39: Tracking is successful because the person has a distinct infrared signature. Although daylight is fading, there is still enough detail to be able to accurately compute an estimate of camera motion between frames. | A.48: Tracking is successful because the camera was stationary, and therefore the camera motion does not need to be computed with the aid of visual information. |
| 6 - 10 °C | A.1 – A.6, A.21 – A.30: The infrared is driving the tracking here because the person has a sharp infrared signature. | A.47 Tracking is successful because there is still enough light to estimate the camera motion from visual information. | A.45: Tracking fails because camera motion cannot be estimated in the dark – feature matching is inaccurate. |
| 11 - 20 °C | A.7 – A.19, A.31, A.40 – A.44: At this temperature range we start to see people blending into the background in the infrared image. The success of tracking is dependent on whether HeatTrack is correct in deciding to switch from one modality to the other, or in doing correct data association after an occlusion. | | |
| 21 - 30 °C | A.33 – A.38: In these videos the person blends into the hot background and are therefore not detectable in the infrared image. Whether or not tracking succeeds depends on whether the visible light tracker can track them. | | |


Figure 5.3: Left: screen-grab from video. Middle: SURF key point matches between two image patches 3 frames apart. Right: SURF key point matches between two image patches, one before the two people overlap, and one after the overlap. The further apart in time two image patches are, the worse this matching method performs. This method tends to always produce matches however.

the exposed regions of a person show up clearly in infrared. Aspect ratio is arguably more difficult to filter on; much depends on whether we expect someone to appear standing up, lying down or in a crouched position. Someone standing up will have more of a rectangular aspect ratio than someone who is curled up.

The proximity measure is a measure of how far the measurement is from the previous known location of the person, and would ideally be related to the estimated velocity of the person in the video. Knowing what velocity a person was moving at, we can discard measurements which are too far from the expected location given the velocity. The difficulty with this is that early on in tracking, the Kalman filter has not yet converged to an accurate velocity estimate, so this can only be approximated.

Another way to test if a measurement corresponds to the tracked person – histogram matching – is only useful if we know what an appropriate score for the histogram match should be. If the background of the person has not changed much since the last frame in which tracking took place, then the histogram score is likely to be high. Where things go wrong is if the person is occluded for a couple of frames and then re-emerges. In that case, the background could have changed a lot from the previously tracked patch, and the histogram score is likely to be low even though it is the same person. Another method tested was to look at the percentage of SURF key points correctly matched between two image patches of a person: the saved image

patch and the hypothesised image patch in the current frame. The idea is to extract SURF key points from both patches, match them, compute the homography from the matches and then consider the percentage of inlying projected points. Figure 5.3 shows some examples of this. This works well as long as the two patches are similar, but runs into the same problems as histogram matching if the background has changed in any way, and this is to be expected if the person is moving. Another issue with SURF matching is that there *always* tends to be matches, even if the images are completely different. SURF key points are more suited to matching images of the exact same scene but under different viewpoints.

Closely related to the problem of matching two image patches is the problem of drift in the tracker's internal representation of the person. In the current implementation, each time a person is deemed to be successfully tracked, the current visible light image patch is saved as the new representation for the person. If this contains some of the background, then there can be a gradual process of the tracker matching more to the background than to the person, and the saved representation moves away from the person until tracking completely fails. Another difficulty is when two people overlap and the saved image patch includes some of the other person, resulting in incorrect data association in future frames. One way to circumvent this issue is to store a history of the last $n$ sightings of the person, and to compare new hypotheses with all the previous patches rather than just the most recent one. This makes it more robust to drift but it is still fallible in the case of changing backgrounds. Another way to deal with it is to compare the hypothesised location of the visible light tracker with the predicted location according to the Kalman filter. If the two are very different, or if the hypothesised location is outside the covariance ellipse, then that is a good indication that the visible light tracker is wrong. However, this goes back to the issue of whether the covariance of the Kalman filter is useful enough to be able to narrow down the search; if the wrong process and measurement noise covariances are chosen, the covariance ellipse may cover the entire image if no measurements have recently come in.

In summary, data association is a difficult problem, especially with people moving against a changing background. In the current implementation of HeatTrack, the score for a candidate measurement is a combination of different similarity measures including histogram match score and whether the hypothesis is within the covariance ellipse of the Kalman filter. A greedy approach to data association is adopted – taking the highest scoring patch – but it would be better to have a more statistically robust method of data association. An alternative to this method of data association is the JPDAF (Joint Probabilistic Data Association Filter) [48], which has previously been used with success for visual target tracking in computer vision. This puts the data association problem into a Bayesian framework and hence has a more theoretically sound grounding than the current greedy data association.

## 5.3 The Planar Assumption

A fundamental assumption of this DPhil is that the scene is planar or near planar; that is, the variation in depth to various points in the scene is negligible compared to the distance from the camera to the scene. In the videos I captured for this DPhil, the scene is not planar in most cases, and this introduces error in two main aspects of the method:

1. Mapping an infrared image to its visible light counterpart. If there are objects at different depths in the scene, then using a single homography to map all image points in one image to the other image will not given an accurate mapping because of the effects of parallax.

2. Approximating the camera motion between two visible light frames. This is required in order to cancel out the camera motion to provide a common 2D coordinate system for the Kalman filter. The assumption is that the camera undergoes an in-plane rotation and translation between two image frames.

In practice, this appears not to have been a huge issue for the videos recorded for this DPhil, as discussed in Section 4.3.7. However, if the camera were to move over

large distances and where the scene depths change significantly then we can expect the homography model to fail. Previous work in UAV-based detection such as [51, 103, 109] uses a homography to map from one image to another, but in those cases the camera was quite far from the scene (>60 metres), making the homography mapping more applicable than in our case where the video footage was captured roughly 20 metres from the scene. This section discusses the shortcomings of the current approach and discusses ways it could be improved.

## Mapping Infrared to Visible Light

In Section 3.3.3, we proposed a method to deal partially with the problem of mapping an infrared image to its corresponding visible light image: given a salient image patch in the infrared (a person's silhouette, for example), look for the most similar HOG descriptor match in the visible light image (after downscaling the visible light image so that the person should appear the same size in both images). In the majority of cases tested, this gives a more accurate estimate of point correspondence than the homography mapping. For this to work well, the distribution of gradients around the object of interest should be fairly similar in both images. As long as the object does not completely blend into the background in the visible light image, the method is successful.

The major downside of the matching method is that it only works when matching infrared to visible light, and not the other way round – matching a visible light image patch to infrared. The nature of the visible light modality means that a patch extracted from the visible light image will most likely be full of detail (and noise), and trying to match this to the less detailed infrared image using HOG descriptors tends to give incorrect results. Matching a simple template to a detailed image works better than matching a detailed template to a simple image.

## Approximating Image Motion with a Homography

Section 4.2 showed that if the inter-frame image motion could be estimated accurately, then the images could be warped to produce a sequence of frames which look like they

were taken from the same viewpoint. An alternative way of looking at it, if we do not wish to stabilise the images, is that by cancelling out the effect of image motion, a point in frame $n$ can be expressed in the same (image) coordinate system of frame 1. This means that we can treat the tracking problem as that of tracking an object in 2D image space with a stationary camera.

If the scene is planar, then a homography is a reasonable model to approximate the image motion between frames. If the scene is not planar, but still far away, the method still provides a reasonably accurate mapping between two consecutive frames, but the errors accumulate over time especially if the camera moves a lot and/or the scene structure changes a lot from what it was initially.

Ultimately the goal should be to track the object in 3D. This would eliminate the need to convert image coordinates to a common coordinate system, as there is one already - the world coordinate system – but the difficulty is in estimating the depth to a point in the scene. The next section describes how one might go about doing this.

## 5.4 Towards 3D Tracking in Aerial Views

It is possible to calculate the distance to a point in the scene from two views of the scene [60], provided the images were captured sufficiently far apart as to make it possible to triangulate the point accurately. The further away the camera is from the scene, the wider apart the views would need to be. This is because points which are far away have much less disparity between two views than points which are close-up.

In the camera rig used for this DPhil, there is not a sufficiently wide baseline between the two cameras in order to triangulate scene points accurately. The only way this could be done is by doing something similar to Parallel Tracking and Mapping (PTAM) [68] – triangulate scene points between two views from the same camera taken at different times (say, every $n$ seconds, where $n$ is chosen depending on the speed of the camera and the distance to the scene).

If the tracking method is eventually to be ported to a UAV, then an obvious thing

to do is use the onboard Inertial Measurement Unit (IMU) and altimeter in order to get estimates of the pitch/roll/yaw as well as approximate altitude above sea level. One caveat with the IMU is that the raw values from it are almost never reliable on their own. Instead, they give a reading that is somewhat close to the actual value, but with random noise added. There is also the issue of synchronising the measurements from different sensors which are coming in at different rates. Given the inaccuracies inherent in individual sensor measurements, in practice the outputs from multiple sensors would be fused with a Kalman filter.

Knowing the altitude of the camera can help in computing the distance to an object on the ground. The altitude above sea level is computed from the pressure and temperature sensors of the IMU; atmospheric pressure drops as you gain altitude, allowing the change in elevation to be calculated. However, barometric altimeters are affected by changes in the weather, such as when a high or low pressure system moves in, so they must be recalibrated at the start of every flight. GPS, on the other hand, is very accurate in terms of horizontal positioning, but it can give grossly inaccurate altitude estimates especially if the nearest satellite is near the horizon[1]. Both estimators give a height estimate above a certain point; this poses a problem for UAVs travelling in mountainous regions, the most likely application area of this DPhil. In that case the altitude estimates would need to be combined with a topographical map which would allow one to compute the distance to the ground at a given point.

A final depth estimation method not mentioned in this context is laser scanners. They send out a laser beam in all directions and measure the time it takes for the laser beam to bounce back. The time taken is then used a measure of the distance to the object. Laser scanners are widely used in robotics on ground robots, for generating a map of a building [120], say, and onboard large helicopters, for generating elevation maps or 3D city models [36]. They are however less applicable to the small robotic helicopters popular today due to their weight: small robotic helicopters typically have limited payloads which are usually taken up by processors, camera, battery

[1] https://support.garmin.com/support/searchSupport/case.faces?caseId=%7B66f1b0a0-4cd6-11dc-4733-000000000000%7D

Figure 5.4: AscTec Falcon 8 octocoper

etc. A laser scanner combined with other payload can make it impossible to fly or significantly hinder flight time, which at best is only around 20 minutes. There is also a significant synchronisation issue which must be solved – how to correctly match a laser reading with its corresponding image.

## 5.5 Implementation on a UAV

The results presented in this thesis are for a handheld camera rig with a certain amount of camera shake induced to simulate the footage one might get from a UAV. This section describes our initial experiments with getting the camera rig onto a UAV, and describes the challenges encountered when trying to put a multi-modal camera rig on a UAV with limited payload. The system developed as part of this DPhil was intended to fly onboard an AscTec Falcon 8 Octocopter[2] but, following a crash during initial experiments due to loss of a radio link with the controller, it was deemed premature to put the whole camera rig onboard. The infrared camera was the main item at stake due to its value (about $8,000).

UAVs are typically equipped with an IMU, GPS and wireless radio link, and

[2] http://www.asctec.de/en/uav-uas-drone-products/asctec-falcon-8/

some have an autonomous mode whereby the operator can set waypoints for the UAV to automatically follow. Some, such as the Ascending Technologies Falcon 8 shown in Figure 5.4, are designed with aerial photography in mind; this is made possible by a gimbal stabiliser for the camera which ensures that the camera stays pointing in the same direction even in high winds. Given the limited payload of small autonomous UAVs, there is a limitation to the amount of computing power it is possible to equip them with. Therefore, special attention must be paid toward getting the right hardware onboard which is light enough to fly but also powerful enough to capture images from two cameras and write them to disk without sacrificing frame rate or overflowing memory buffers.

As already mentioned, using a UAV to capture video footage has the added benefit of an IMU to provide better motion estimates than looking at visual information alone. One of the main issues in combining information from an IMU with camera images is synchronisation. This is especially important if the UAV is moving fast, because even a small shift in the platform can result in a large change in the field of view of the camera. The cameras used in this DPhil have frame rates of 15 and 25fps but IMUs typically have much faster refresh rates. To synchronise them to a reasonable degree of accuracy would require time-stamping sensors so that information from different sensors can be temporally aligned. Ideally, time-stamping would match the sensor acquisition times – when the sensor reads a value originating from the real world. However, time-stamping the sensor data is affected by various phenomena in the data acquisition chain, such as communication between the sensor and the CPU and operating system scheduling. State of the art approaches to clock synchronisation involve estimating a linear function between the timestamps of two clocks. One synchronisation algorithm widely used in mobile robotics is that of [59], which uses an efficient convex hull algorithm to estimate the relative skew and offset between two clocks. It operates by performing a linear programming optimisation on one-way offset measurements gathered from two clocks which are separated by a variable delay data network. The algorithm can also recover the offset between the clocks, up to but not including the minimum transport delay, which is unobservable from one-way

timing data alone. That algorithm typically achieves better than millisecond accuracy within a few seconds.

---

This chapter touched on the main problem which prevents successful tracking – incorrect data association. Various factors contribute to the HeatTrack algorithm losing a person, including hot weather masking the person in infrared, or occlusions of the person in either modality, but if there was perfect data association these factors would cease to affect the tracker. Incorrect data association is the main limiting factor of HeatTrack. Another key issue highlighted was the assumption that the imaged scene is planar. This chapter proposes an extension to the current work which would relax that assumption and instead estimate the distance to points in the scene. We highlighted the main implementation issues associated with tracking in 3D from onboard a UAV, and proposed several solutions which draw upon previous work in both ground and UAV-based detection/tracking.

# Chapter 6

# Conclusions

This chapter summarises the overall body of work, highlighting the most important contributions and discussing the limitations of the proposed techniques. Potential directions for future work are described, including further developments to the approaches presented and related open questions.

.

## 6.1 Summary of Contributions

The principal contributions of this thesis are (i) integrated infrared/visible light detection of people from aerial views, (ii) HeatTrack, an algorithm for tracking a person in such imagery by combining information from both modalities and (iii) a method of compensating for camera shake so that a velocity of a person can be estimated, which means that tracking can continue during occlusion.

### Integrated infrared/visible light detection

As an alternative to searching an entire visible light image for a person, we proposed an integrated approach which uses the infrared image to guide the search in the visible light. This demonstrated two useful characteristics: it leads to faster initial detection of people than if the whole visible light image was searched, and it leads to fewer false positives (and therefore higher confidence in the detection) than if there was no infrared camera. Two significant improvements to the method were introduced. First, the use of either local binary patterns or histogram of oriented gradients to find potential person candidates in the infrared does a better job than thresholding of segmenting out the person. Second, the use of HOG descriptors allows for better matching of the infrared signature of a person to their corresponding visible light image patch. The method was tested on 250 image pairs sampled randomly from the data set of 47 videos recorded specifically for the DPhil. This extends the previous work of [51, 104, 109], which dealt with video footage recorded at much higher altitudes, thus precluding the use of complex part-based algorithms for the detection of people.

### HeatTrack

To track the person from frame to frame we proposed HeatTrack, a novel algorithm for combining measurements from two modalities in order to localise a person in

subsequent frames. HeatTrack has several advantages over existing state of the art tracking algorithms operating in one modality only. First, it is a more computationally efficient way of localising a person in an image frame than if a part-based detection algorithm were to be run over an entire frame. Second, it is better able to track a person in situations where visible light trackers fail, due to the insensitivity of infrared to illumination. Third, it outperforms in situations where there is a high level of detail which normally confuses a tracker, specifically in close-up imagery where the person can change quite a lot from frame to frame. This is again due to the insensitivity of infrared to colour changes. To the best of our knowledge, this is the first published algorithm for tracking people in combined infrared/visible video footage taken from aerial views.

## Motion Compensation

HeatTrack was adapted to work in the case of a moving camera. If the scene is planar or far away, a homography can approximate the mapping between consecutive video frames. Using a sequence of intermediate homographies to convert image coordinates to a reference frame, tracking can proceed in a fixed 2D coordinate system. The image velocity of a person can be estimated, and this enables tracking even with major shifts in the camera and during periods of occlusion.

The method was tested on 47 different videos recorded specifically for the DPhil, and this data set has been made publicly available. To the best of our knowledge, this is the only combined infrared/visible light video data set publicly available where the videos were taken from aerial view points with a moving camera. The method was compared with a situation where there is no infrared modality available. In the case of detecting a person initially, HeatTrack works better than a full image search in cold weather, because the infrared is able to narrow down the search to just the warm parts of the image. In the case of tracking, HeatTrack outperforms three state of the art tracking-by-detection algorithms in all weather conditions tested. The additional modality means that HeatTrack can only do better than algorithms operating

in the visible light domain only. In cases where the infrared does not allow for a good segmentation of the person, HeatTrack is generally as good as the best algorithm operating on the visible band image. In particular, we demonstrated the improvement of HeatTrack over the three state of the art tracking-by-detection algorithms [12], [66] and [58].

## 6.2 Strengths and Weaknesses

The in-depth analysis of chapter 4 identified various situations where HeatTrack outperforms existing state of the art trackers. This is due to the additional information provided by infrared, and a method of camera motion cancellation which allows the velocity of a person to be estimated, which in turn makes it possible to track a person during periods of occlusion. The algorithm is robust to lighting changes, making it useful when a person goes into the shade, is camouflaged against a similar coloured background or when daylight levels are fading. When it comes to the initial detection of a person, HeatTrack is able to discard large parts of the visible light image and focus on a smaller region of interest which makes computation times much quicker than if the whole visible light image were searched.

An area of future improvement is in making the system less dependent on the visible band to provide motion estimates. Chapter 4 showed some videos taken at night where HeatTrack failed in spite of the clear infrared signature of the person. An inability to estimate the camera motion from dark detail-less images meant that measurements could not be expressed correctly in the common coordinate system of the Kalman filter. Another caveat of HeatTrack is the assumption that the scene is planar, which would not be the case in mountainous terrain or in close-up video footage. Chapter 5 proposed a solution which involved using inertial measurement unit data to estimate depth to points in the scene and to provide an alternative method of estimating camera motion. Finally, a major point of failure for HeatTrack is incorrect data association – not knowing with a reasonable degree of certainty whether image measurements should be trusted or whether the motion model of the

Kalman filter should be used instead. It suggests the use of an online machine learning algorithm to learn the appearance model of a person as tracking proceeds.

## 6.3 Future Work

There are many ways in which the work presented in this thesis could be extended or improved. This section explores several such possibilities.

### Better Data Association Methods

In the current implementation, once tracking has failed, no attempt is made to re-detect the person. An obvious next step would be to run the initial detection step again in order to re-localise the person. This would require knowing when exactly tracking has failed, and once a new detection comes in, determine whether it is a new person that needs to be tracked, or whether it is associated with an existing tracker. Both of these require comparing a new image patch with a previous one and determining whether they are referring to the same thing. As discussed in Section 5.2, matching a current image patch to a previously tracked person is difficult especially if the background of the person has changed significantly. Rather than use simplistic methods like histogram or key point matching, it would be better to use something akin to what tracking-by-detection algorithms like [12, 58, 66] do – use machine learning to learn the appearance of an object as tracking proceeds. HeatTrack already uses the Structured Output SVM tracker of [58], but this is for the visible band image only. It would be interesting to combine information from both modalities in such a framework.

### More Detailed Infrared Processing

This thesis uses the infrared to remove large regions of the scene from further consideration, with the idea of doing more complex image processing in the visible light modality. This is because of the perceived greater level of detail available in the visible modality. To this end it uses classifiers based on local binary patterns or histograms

of image gradients to find blobs of approximately the right shape in the infrared image. Given the relatively high resolution of the infrared camera, however, it may (and probably is) possible to get more detailed information from the infrared image itself. While most of the infrared detection literature focuses on rigid templates for human detection, there has been some work done on parts-based detection in reasonably close video footage [56]. That system is focused on localising individual body parts in infrared to the extent that it is possible to do human pose classification. With the lowering cost of higher resolution infrared cameras available outside of the military domain, it seems like an obvious next step to apply more complex vision algorithms, those traditionally used in visible band imagery, to infrared imagery.

### 3D Tracking

Each video recorded for this DPhil was recorded from a fixed location, but with a considerable amount of camera shake induced to simulate the type of footage one would expect to get from a UAV. However, if a UAV is travelling across a wide area of varying topology, the homography model for compensating image motion would most likely no longer work. Accurate tracking would only be possible by estimating the 3D scene structure. Doing 3D tracking would require a calibrated camera rig and an estimate of the distance to objects in the scene; as discussed in Section 5.4, this would require multiple views of the object with a sufficiently wide baseline in order to do triangulation, or using the estimated altitude of the UAV in combination with terrain maps. There is the added challenge of synchronising the outputs from various sensors such as IMU and GPS, which have varying latencies, and matching these up with video frames from cameras with different frame rates. It is a very challenging problem, but one which is receiving a lot of attention in the UAV research community, and seems to be an obvious next step in the drive to make UAVs useful for a wider range of tasks.

---

The idea of using combined infrared and visible light video footage in an aerial

setup is still a relatively unexplored area of research. To date, not much work has been done on in this area, and it is hoped that the ideas in this thesis provide an early stepping stone for future work. The reduction in cost of infrared technology outside the military and the increased use of small robotic helicopters both inside and outside of academia means there is likely to be a lot more interest in this area in the future.

# Appendices

# Appendix A

# Tracking Sequences

This section shows a series of tracking sequences which demonstrate the characteristic behaviour of HeatTrack in various different scenarios. In all of the examples which follow, a red box denotes the hypothesis of the visible light tracker operating independently in the visible light modality, but being re-initialised with every infrared measurement. A green cross denotes the prediction of the Kalman filter before taking into account any measurements; a red cross denotes the updated estimate of the Kalman filter having taken into account a measurement.

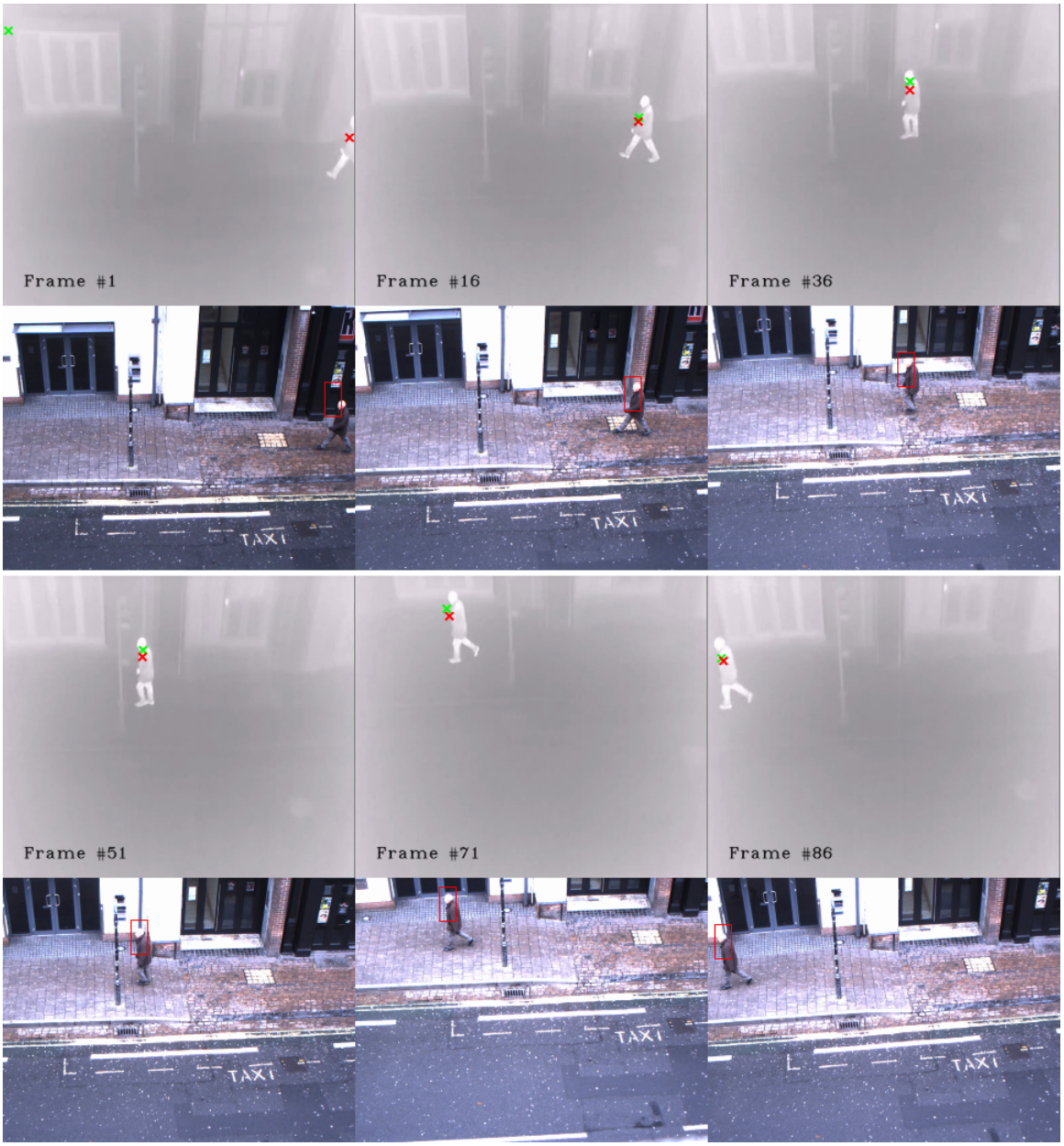



Figure A.1: This is the first of a number of videos which were taken on a reasonably cold winters' day, with temperatures of around 8°C. Here, the infrared is driving the tracking. The person shows up so clearly in the infrared in almost every frame, and therefore there is a well-defined infrared bounding box which is used as input to the Kalman filter. Incidentally, this set of videos is where the visible light tracker tends to fail, as the person is much closer to the camera than in the other locations, and it is thought that the extra amount of detail on the person confuses the tracker.


Figure A.2: Similar to previous video.


Figure A.3: Similar to previous video.


Figure A.4: An example of what happens when two people overlap. Initially there is one person in the scene, and a Kalman filter is initialised on that person. A new person enters the frame and a new Kalman filter is initialised on that person also. When they overlap, the system detects this and ceases taking infrared measurements. When the two people are re-detected separately, the system fails to associate one of the detections with one of the people. This is because of the particular score which was chosen for the histogram similarity criteria.


Figure A.5: This is a repeat of the previous video, but with a lower score chosen for the required histogram similarity. In this case the system does the correct data association when the two people are re-detected after the overlap.

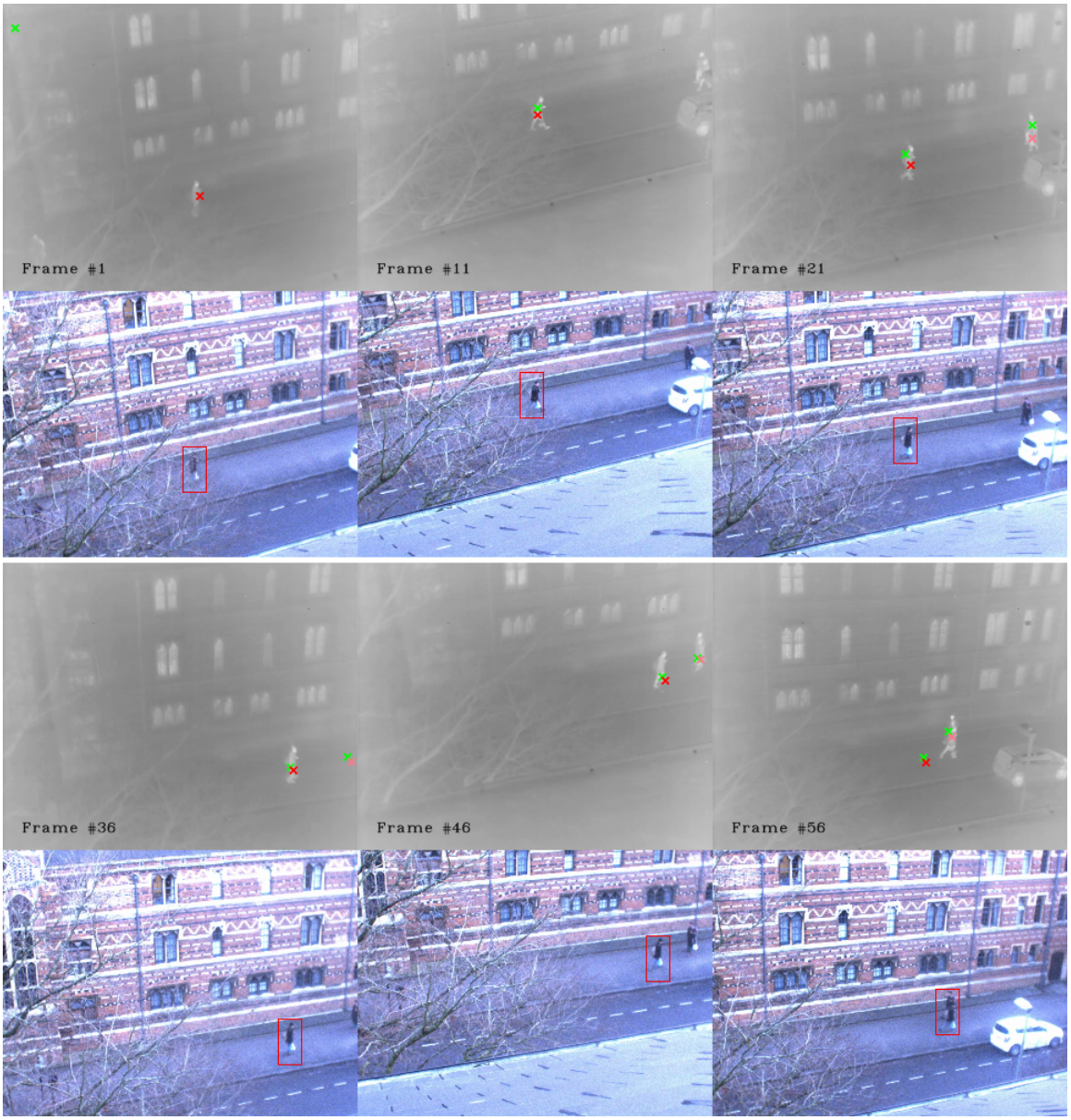



Figure A.6: Here is another example of incorrect data association after two people overlap.

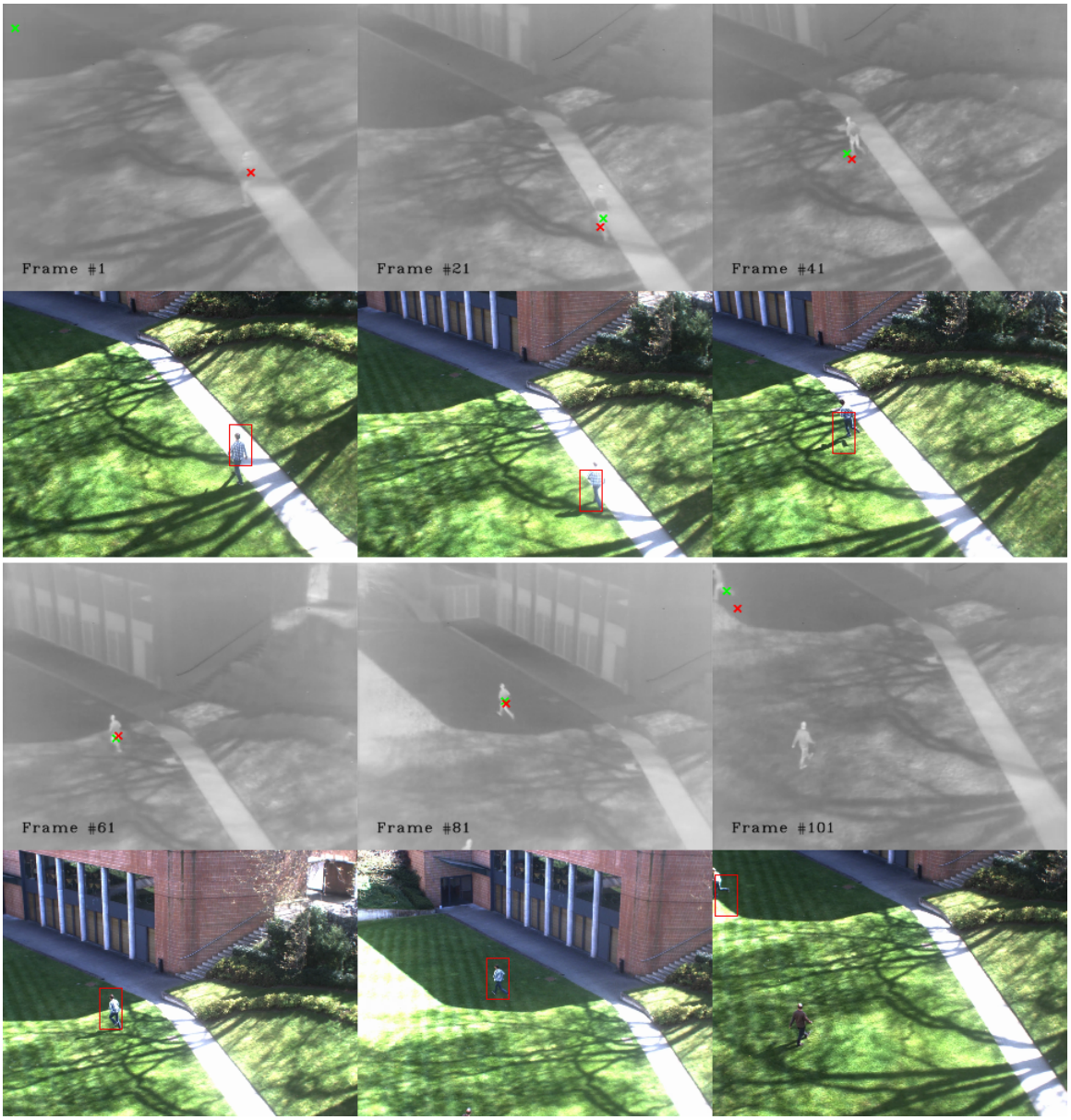



Figure A.7: This video is an example of the two modalities complementing each other. In the beginning, the person is standing in direct sunlight and therefore blends into the hot footpath in the infrared. The Struck tracker is able to track them up to about frame 40. When the person moves into the shade, they are detected in infrared and the system switches to using infrared measurements to track for the remainder of the video.

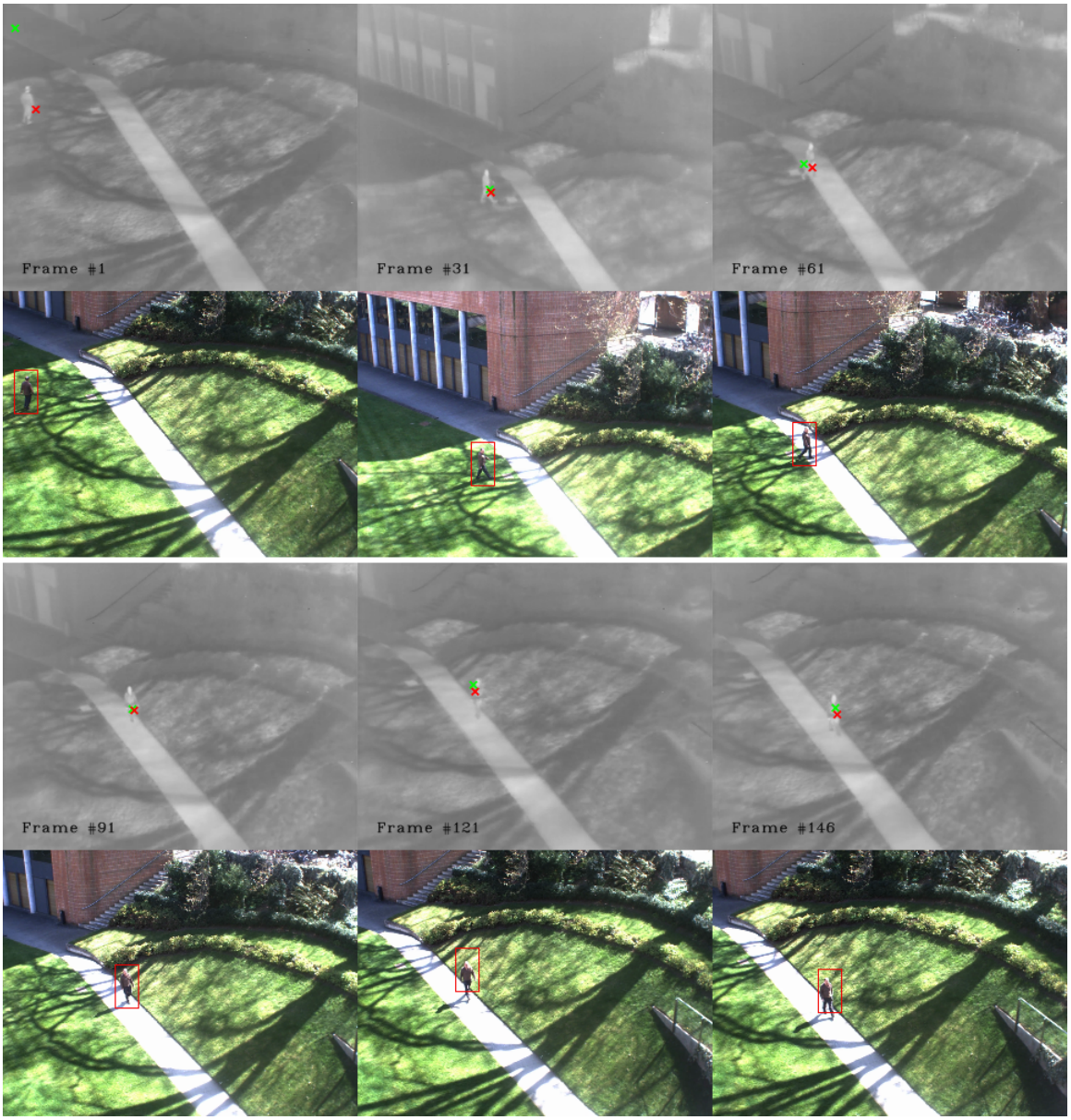



Figure A.8: Similar to previous.

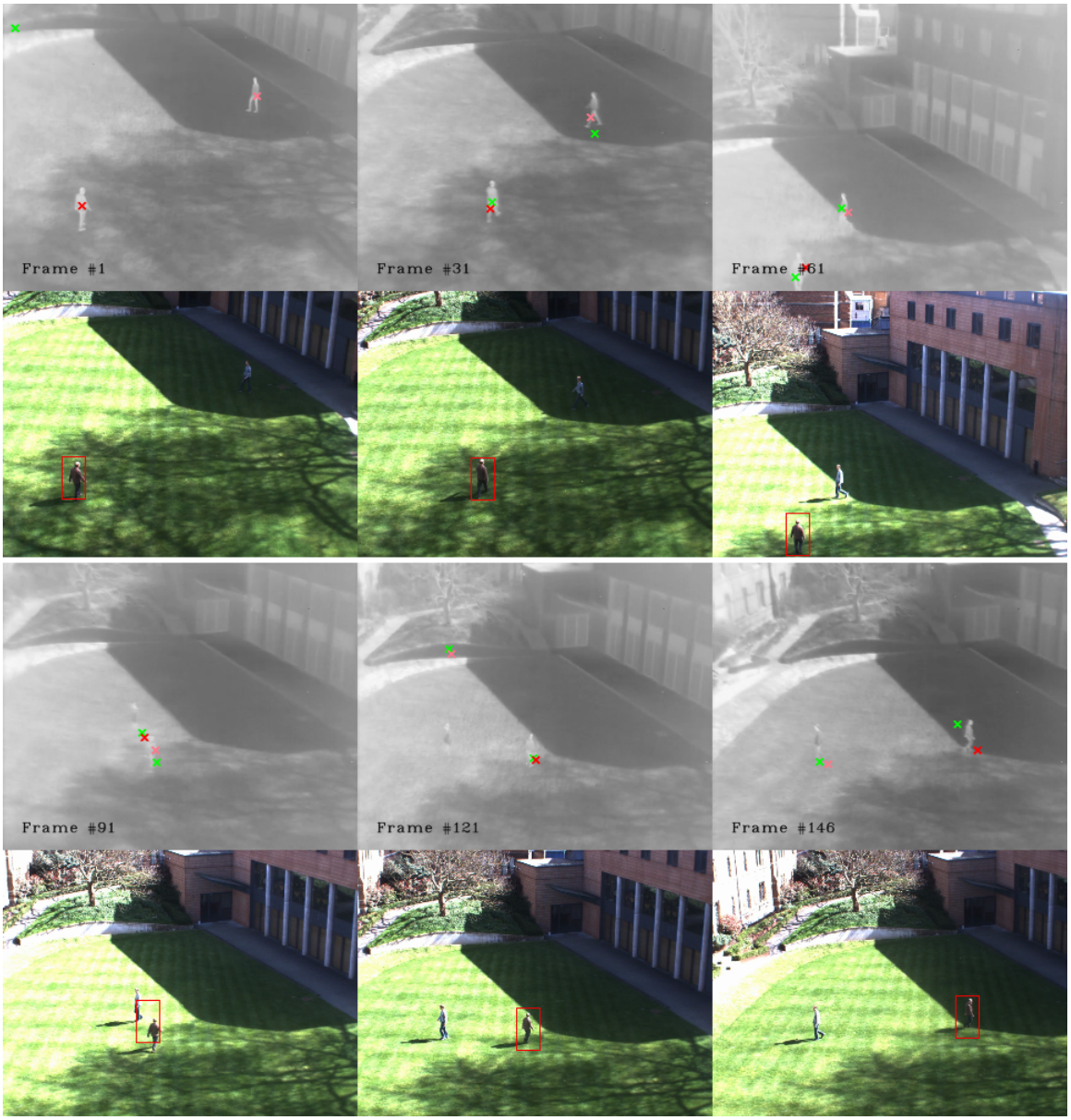



Figure A.9: Similar to previous.

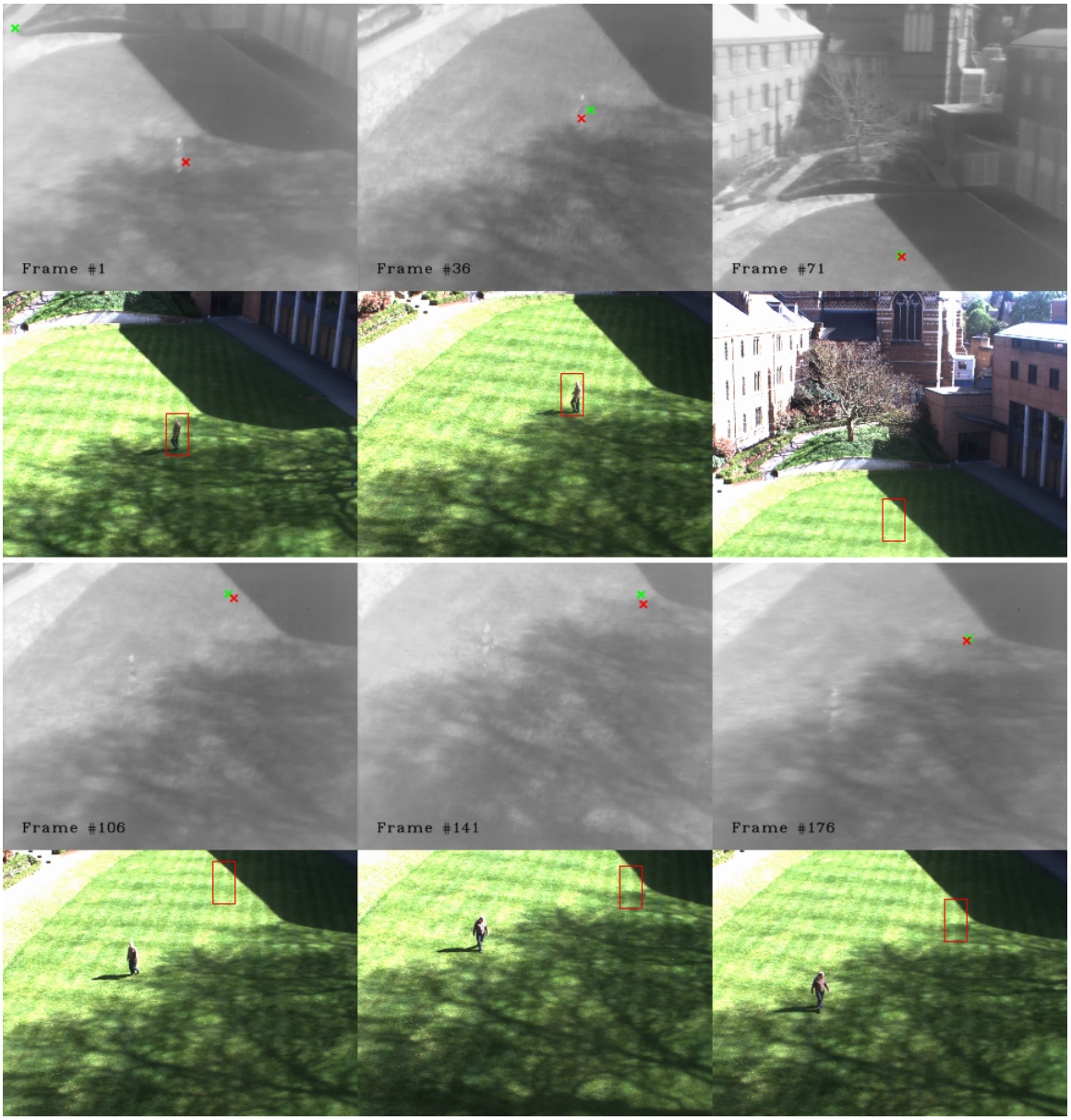



Figure A.10: Early on in the video, the field of view of the camera shifts significantly and the person goes out of view for approximately 17 frames. When the person reappears, they have moved into the hotter part of the scene and are therefore not detectable in infrared. The visible light tracker fails to pick them up either.

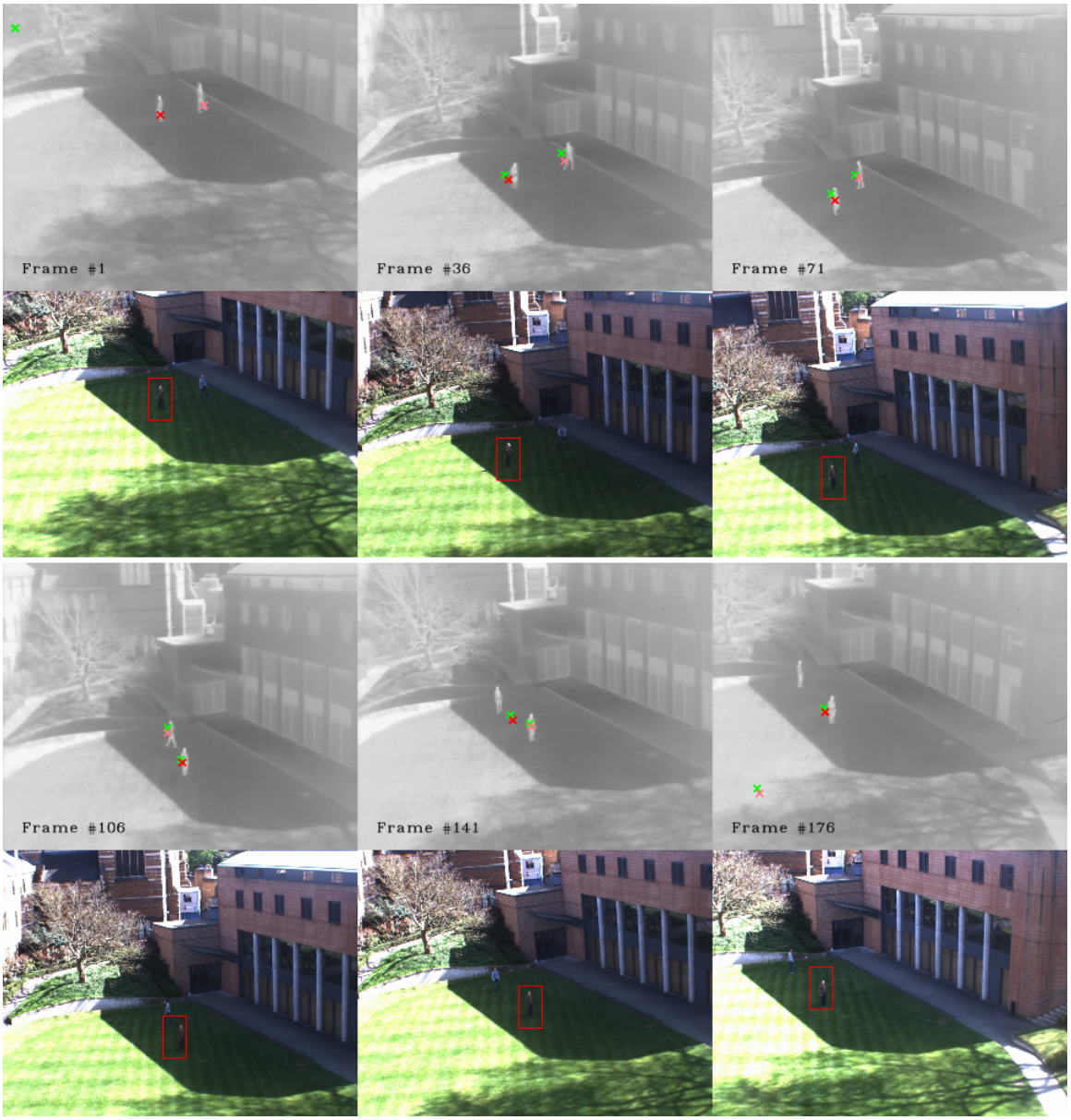



Figure A.11: An example of incorrect data association after the two people overlap. One of the trackers latches onto the wrong person.


Figure A.12: Another example of where infrared helps tracking. In this case the person is hardly distinguishable against a similar-coloured background, but because their infrared signature is so sharp, the system can continue tracking them right up until they become fully occluded. In the last 15 or so frames, the system is reliant on the prediction of the Kalman filter in the absence of any measurements from either modality.


Figure A.13: An example of successful tracking.

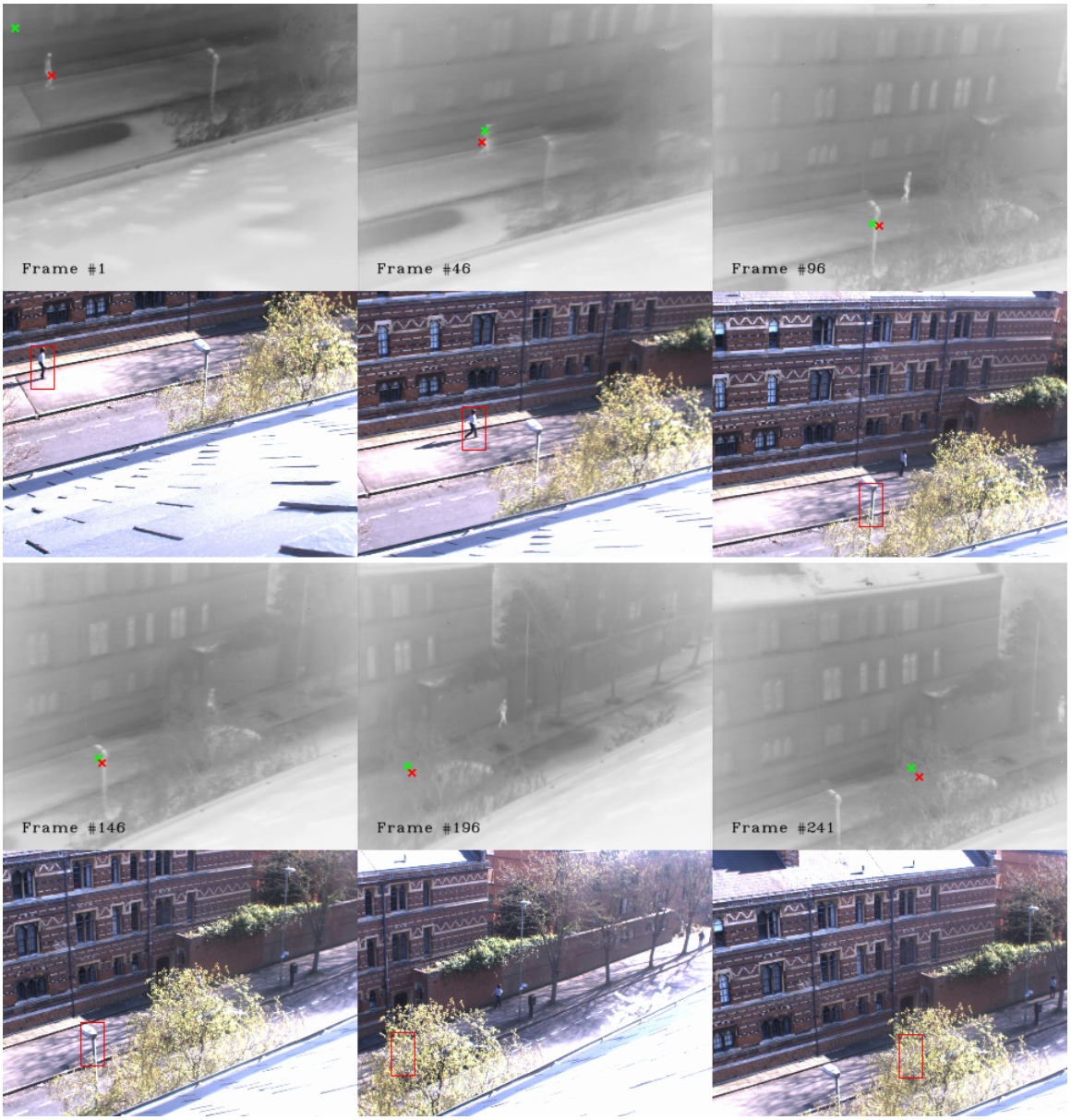



Figure A.14: Another example of incorrect data association – where trusting the visible light tracker too much can result in tracking failure.


Figure A.15: An example of successful tracking before and after two people overlap.

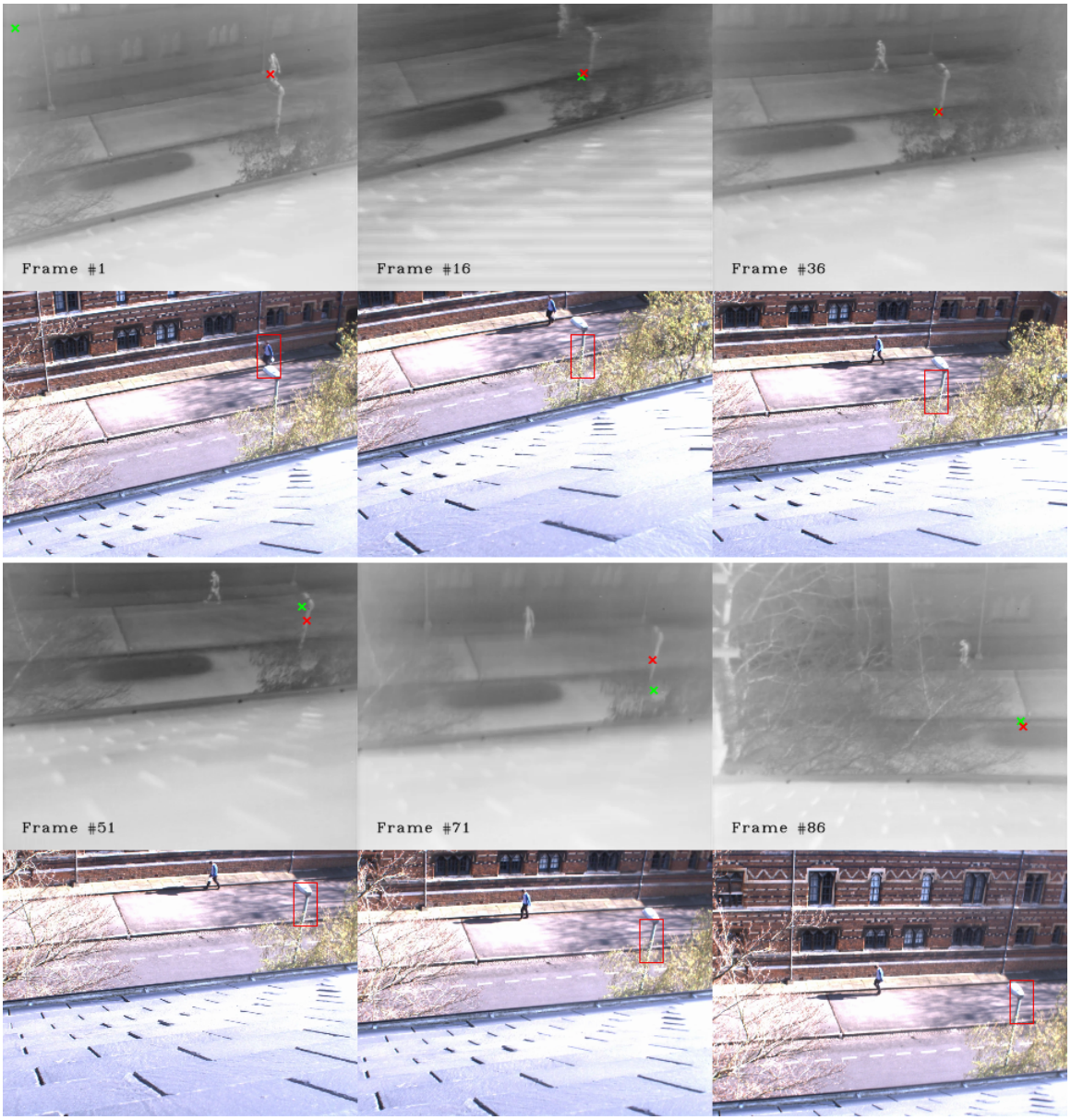



Figure A.16: An example of what happens when the person becomes occluded very early on, before the system has had time to converge to an accurate velocity estimate. The camera moves abruptly and the infrared image becomes blurred. The person is detected in infrared and the system attempts to find the corresponding match in the visible light image, by searching for the image patch which matches the HOG descriptor of the person the best. Unfortunately, the estimate of the corresponding location in the visible light image is wrong. This patch is then compared with the saved image patch of the person, produces a poor match, and consequently, the infrared measurement is not used. The reason it produces a match is presumably because there is a sudden movement in the camera and both images become very blurred. The system checks the output of the visible light tracker, and this is also wrong, also because of the blur. Unfortunately, the system cannot tell that this is wrong, because it passes the histogram match test.

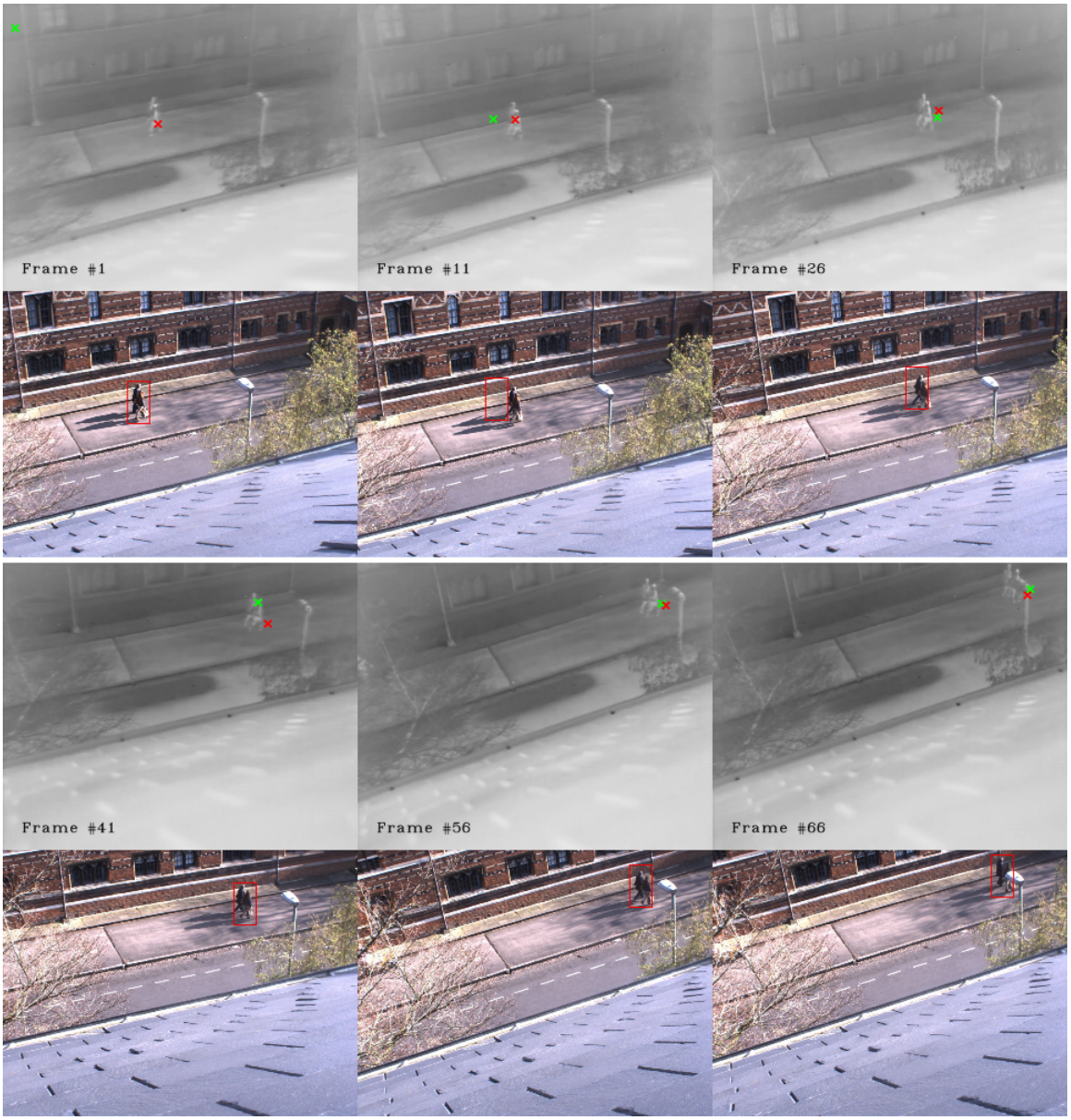



Figure A.17: Another example of successful tracking.

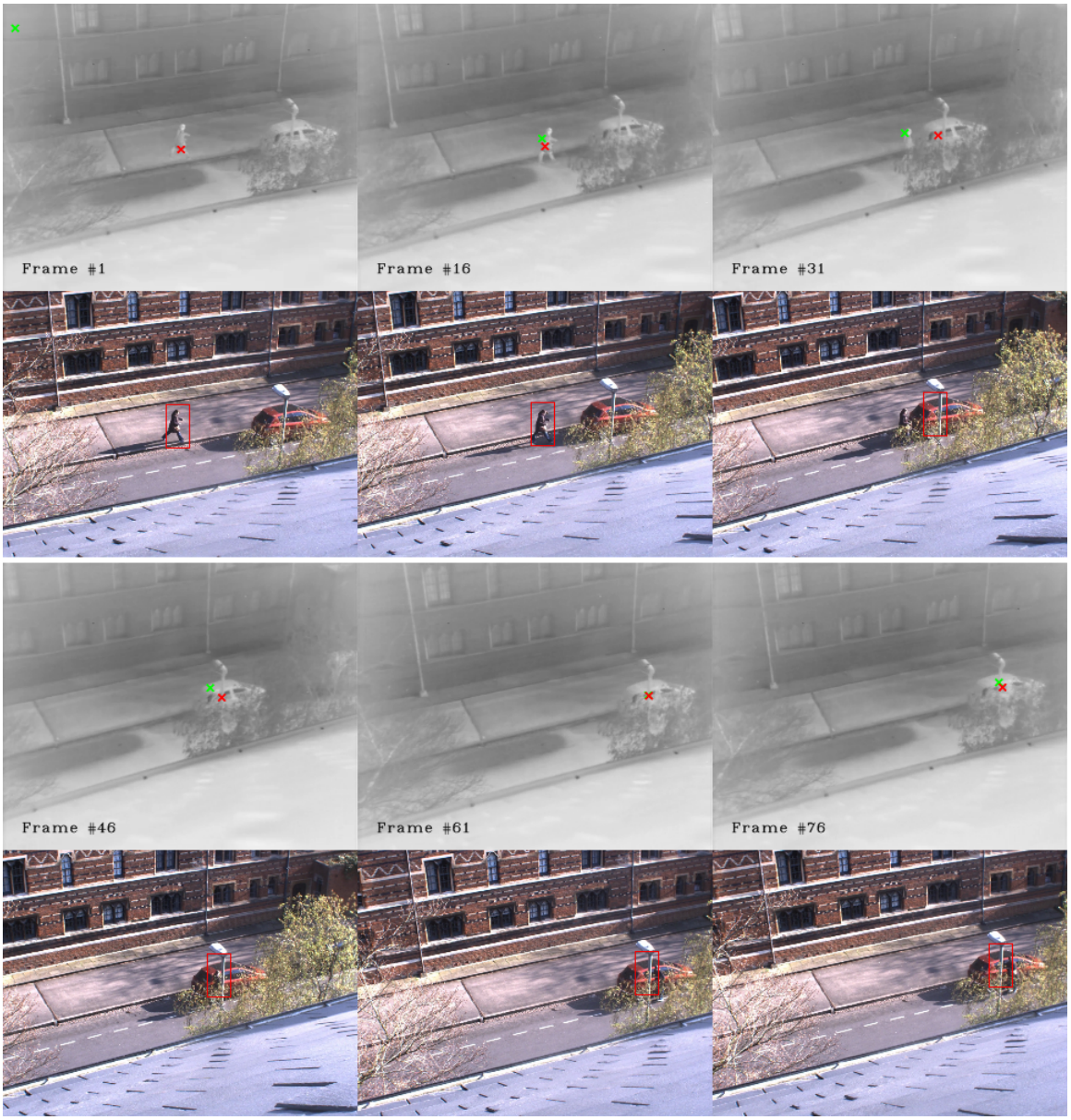



Figure A.18: An example of successful tracking under occlusion.

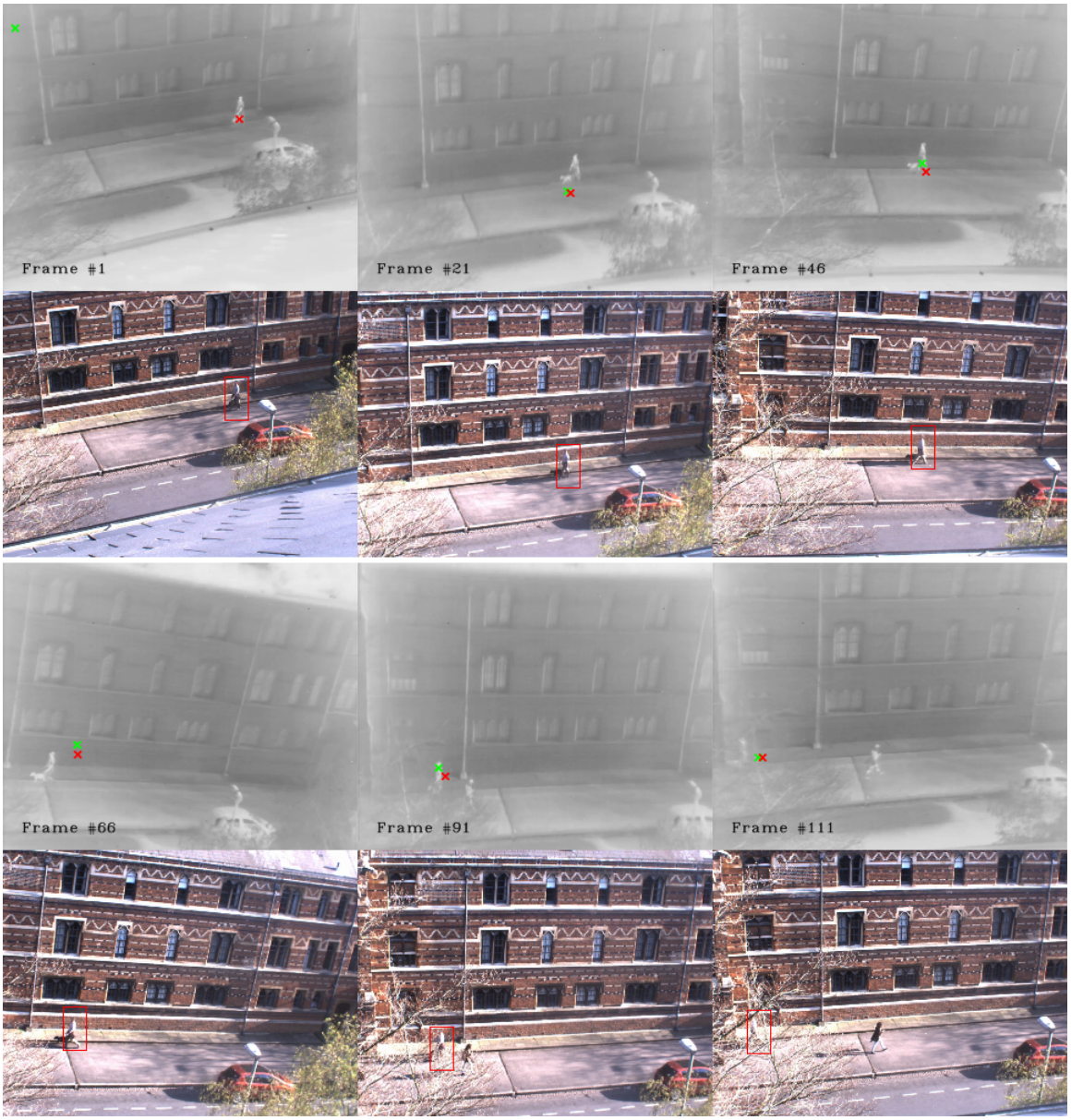



Figure A.19: Another example of successful tracking.

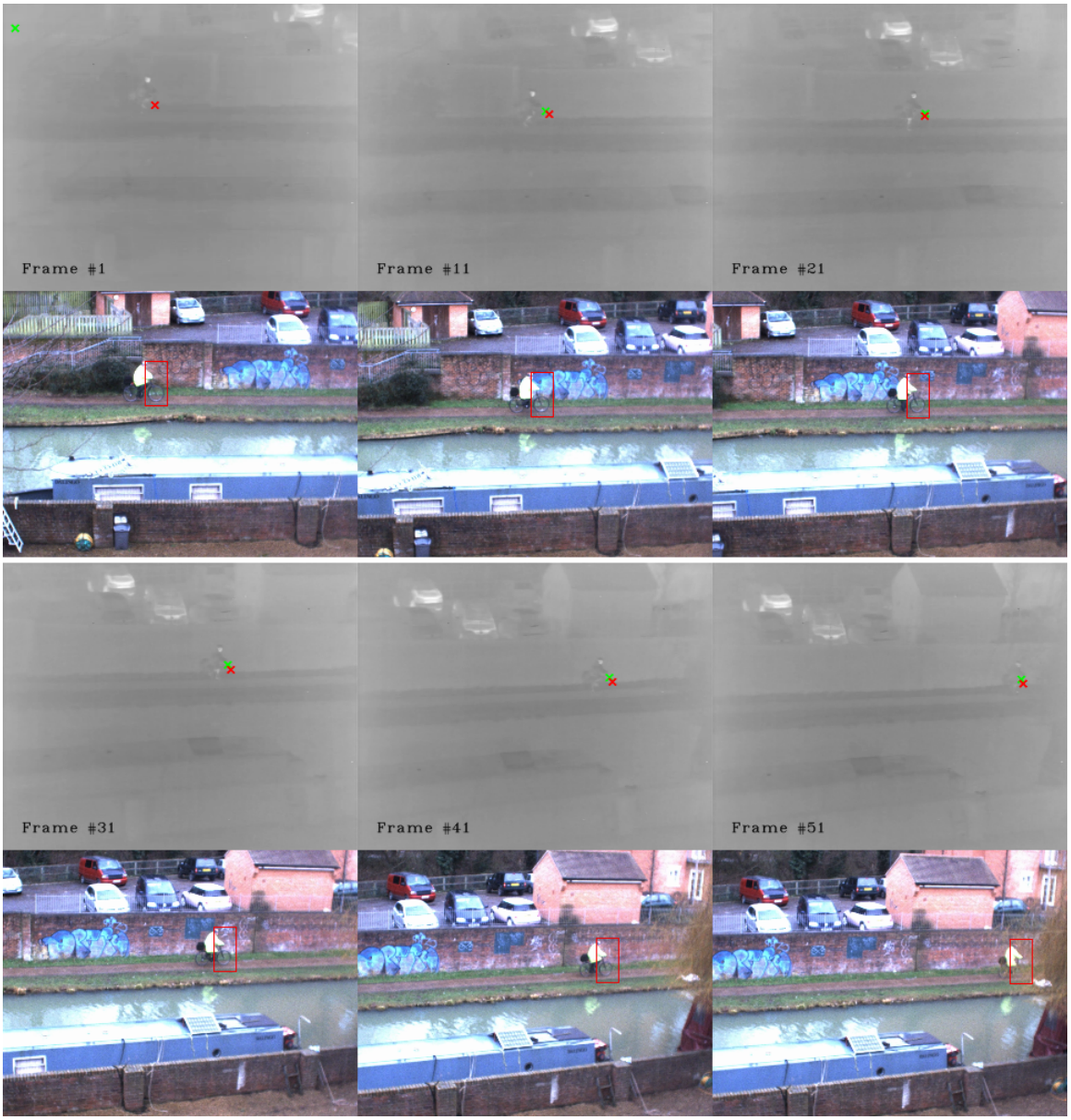



Figure A.20: This is an example of how clothing affects the infrared signature of a person. The video was recorded when the outside temperature was -4 °C, but because the man was wearing a very heavy coat, he was barely perceptible in infrared. However, lighting conditions were sufficient to be able to track the man entirely using the visible band.


Figure A.21: Another example of successful tracking.

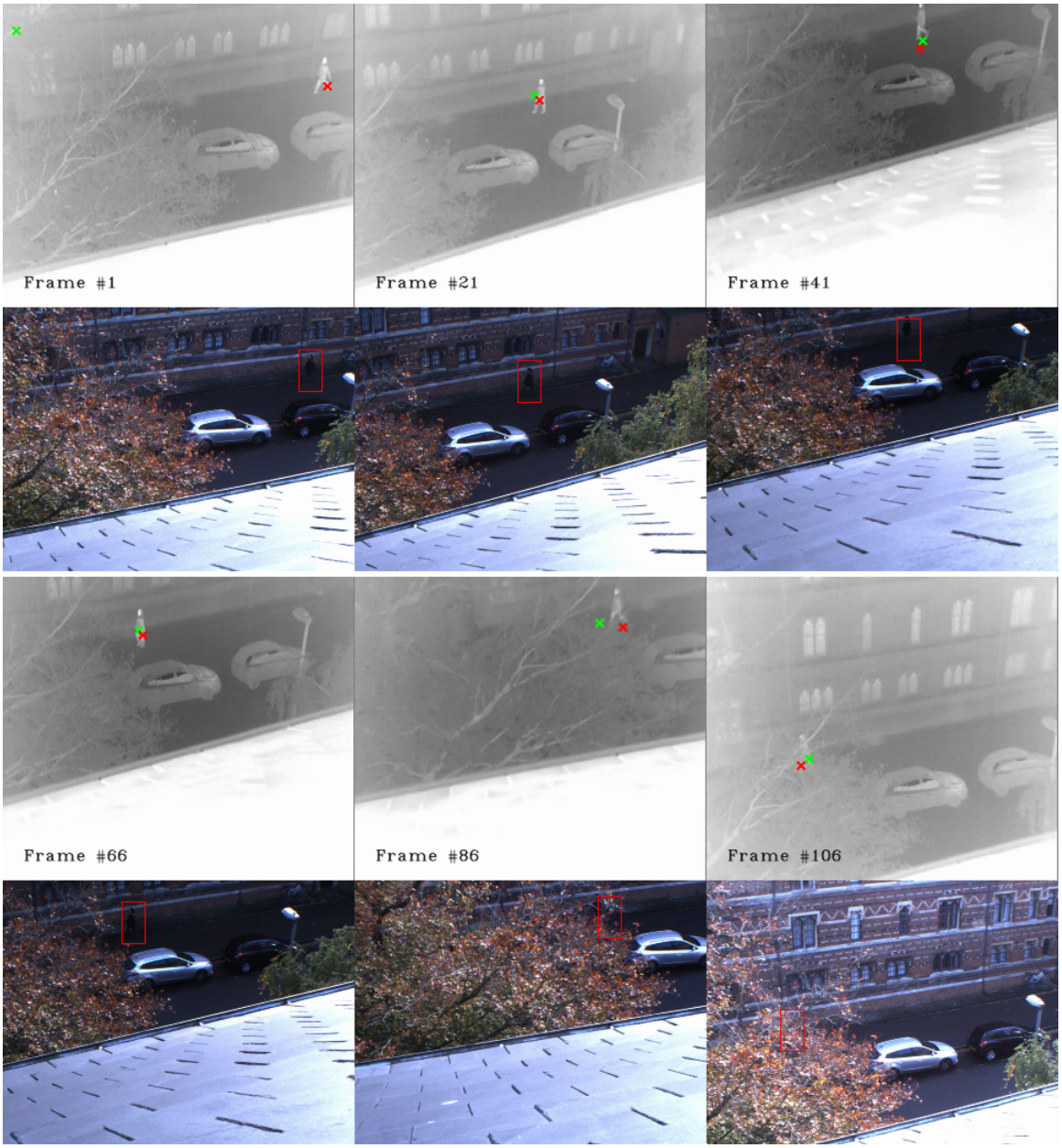



Figure A.22: Another example of successful tracking under occlusion.


Figure A.23: An example of where tracking fails

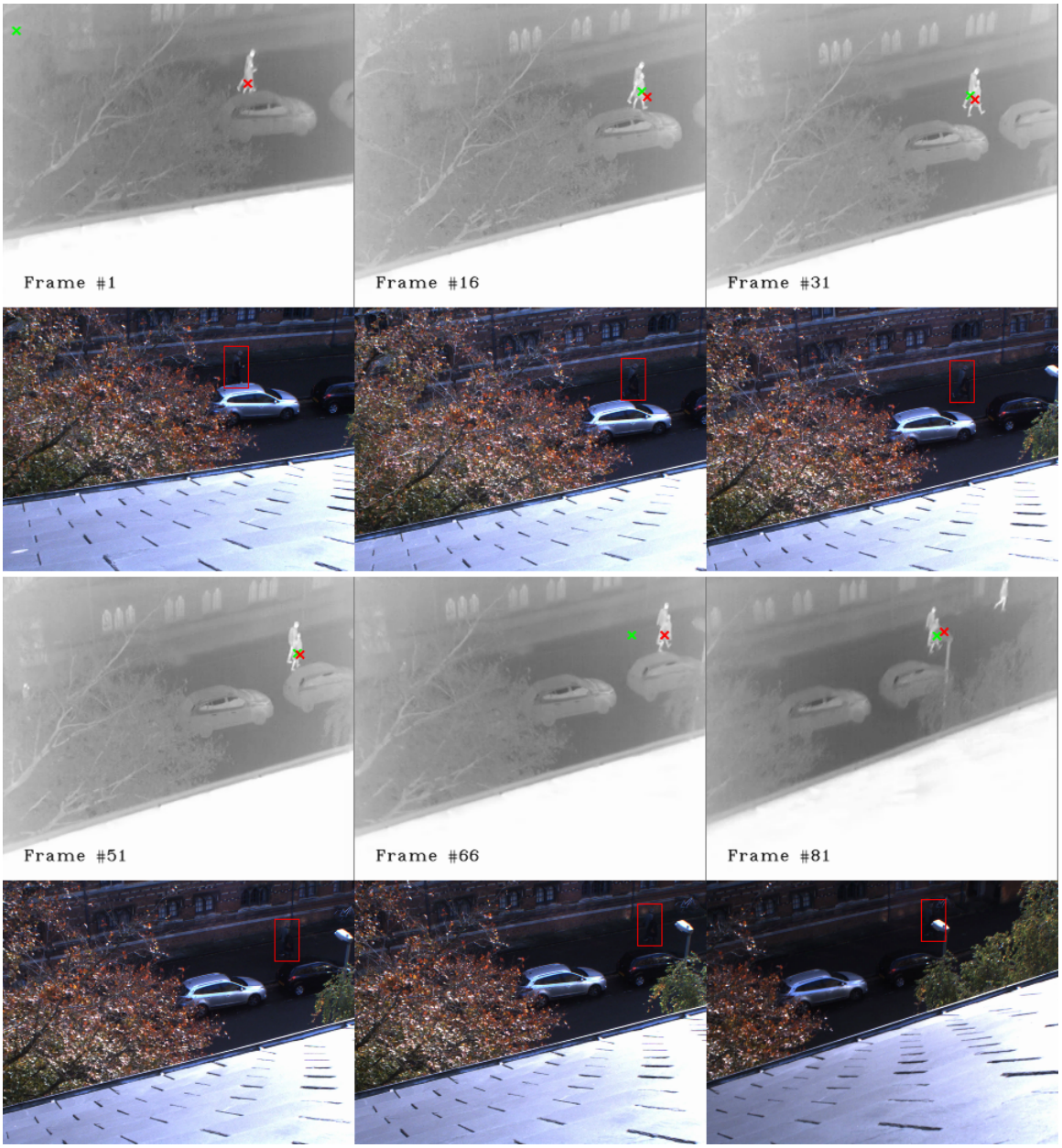



Figure A.24: Another example of successful tracking.

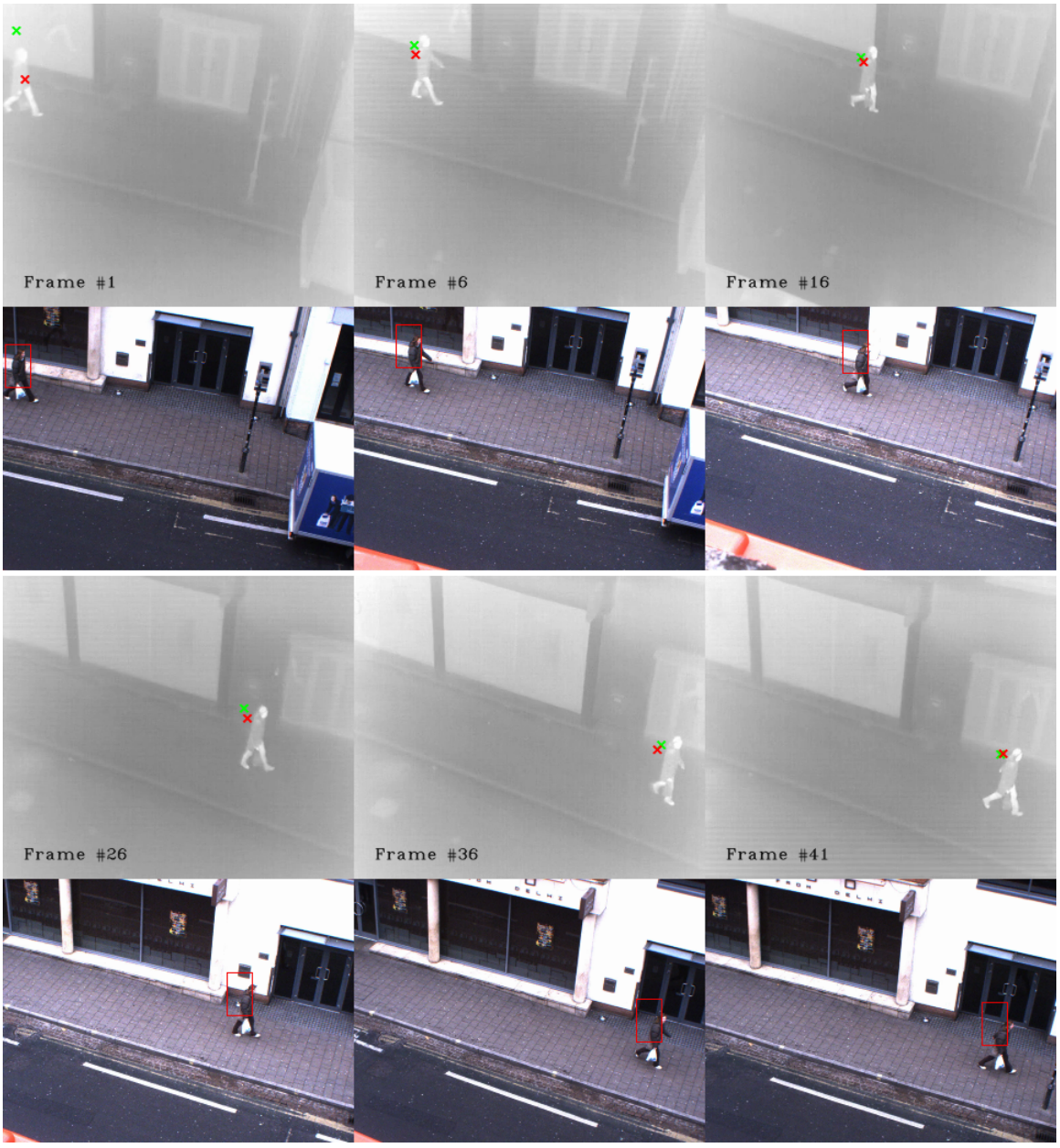



Figure A.25: Another example of successful tracking. This and the following videos A.26 to A.30 are characteristic examples of successful tracking in closer range video footage where the person appears in much greater detail. Tracking a moving person at close range in the visible band often fails due to the extra amount of detail; in these examples the tracking is driven by the sharp infrared signature of the person.


Figure A.26: Another example of successful tracking.

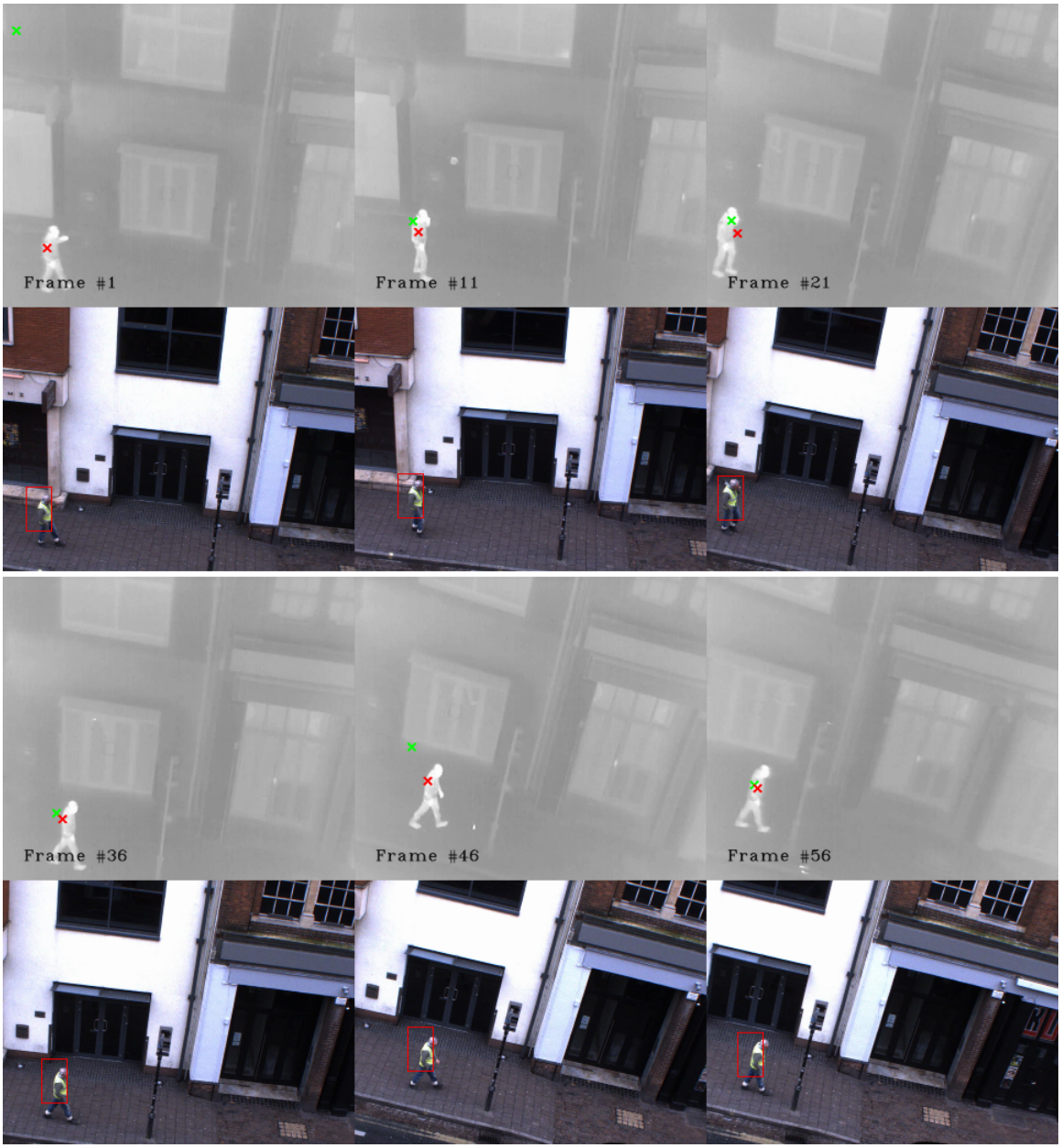



Figure A.27: Another example of successful tracking.


Figure A.28: Another example of successful tracking.

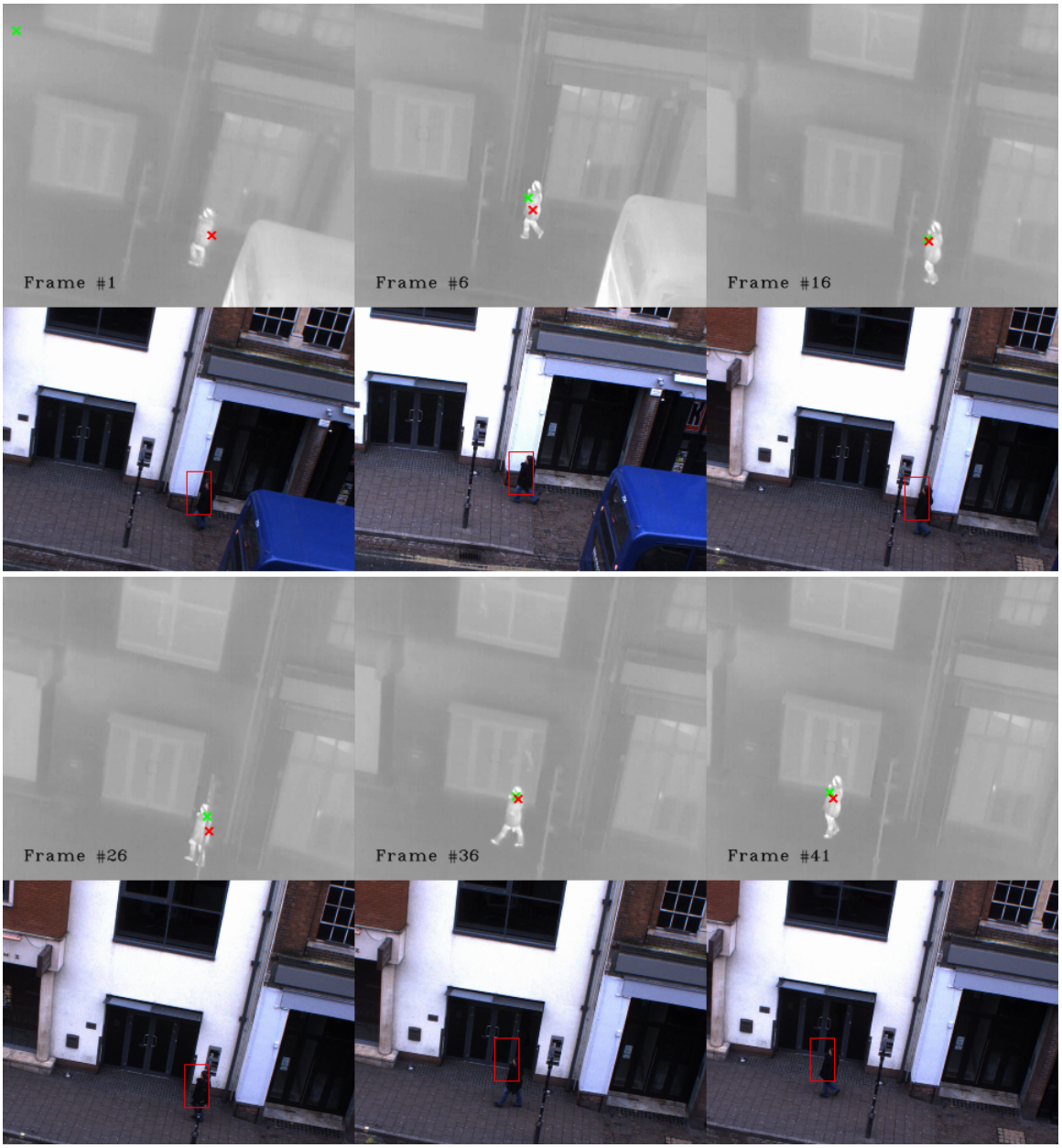



Figure A.29: Another example of successful tracking.

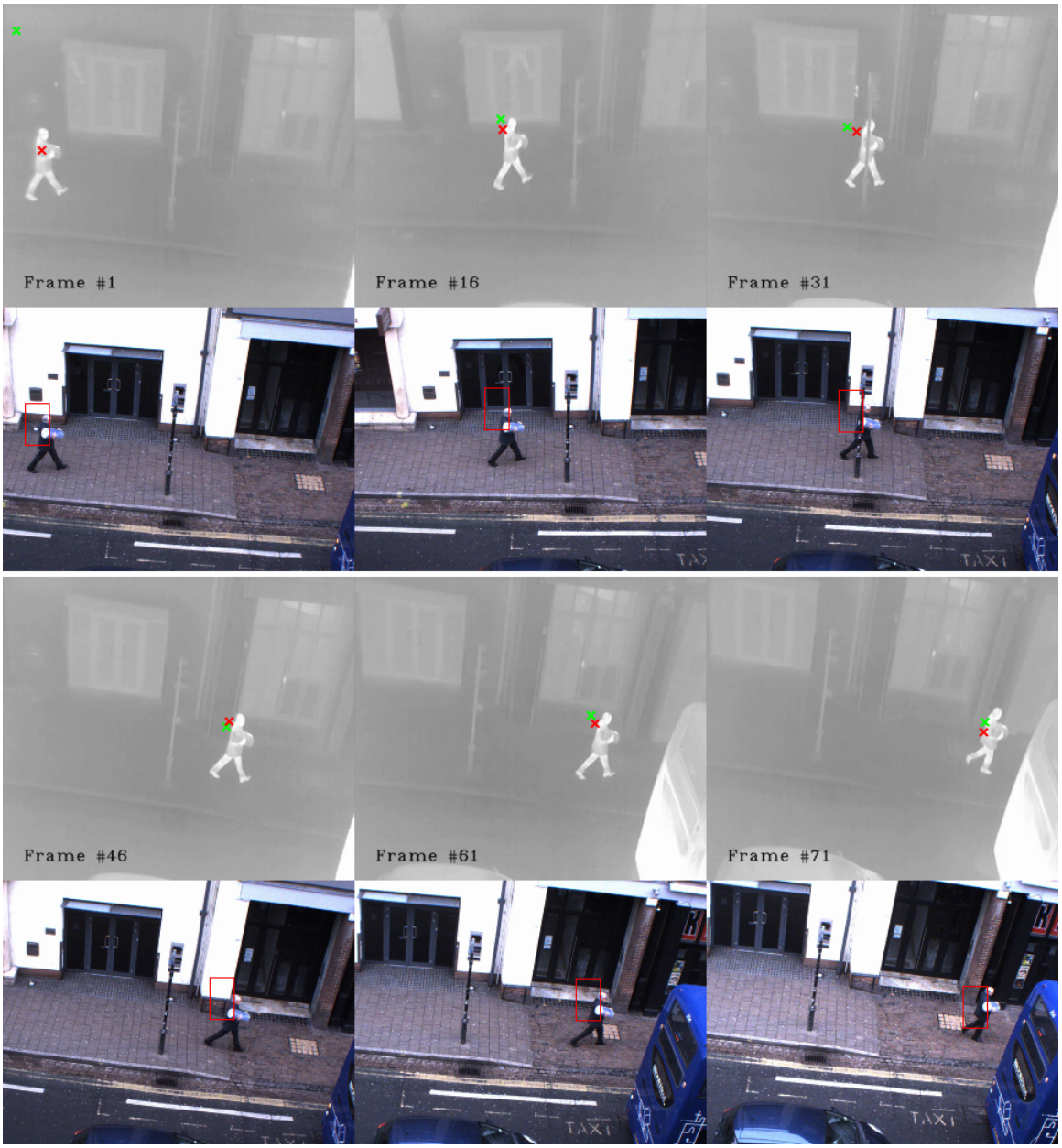



Figure A.30: Another example of successful tracking.

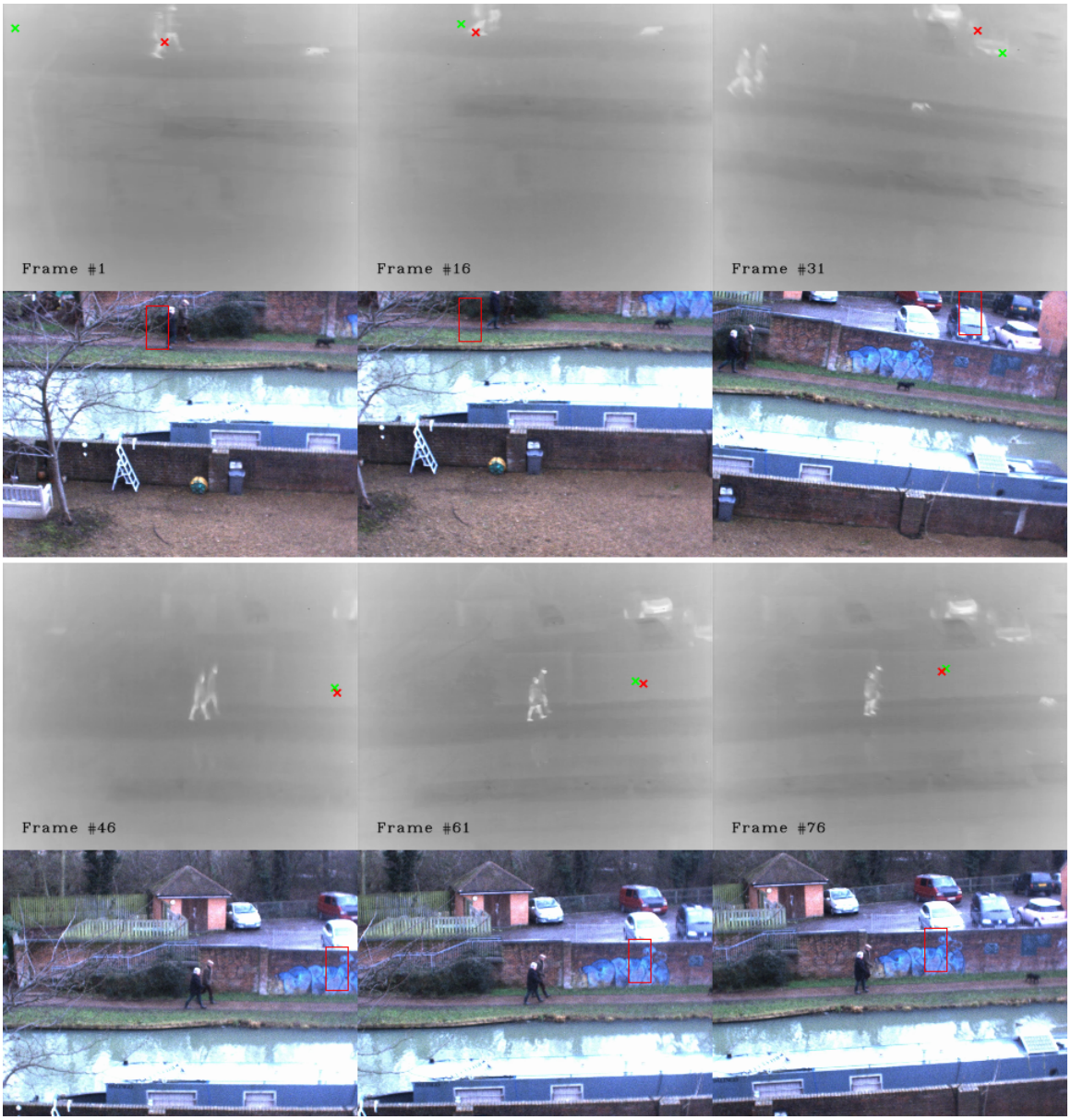



Figure A.31: Excessive camera vibration early on in the video meant that the person was lost beyond recovery.

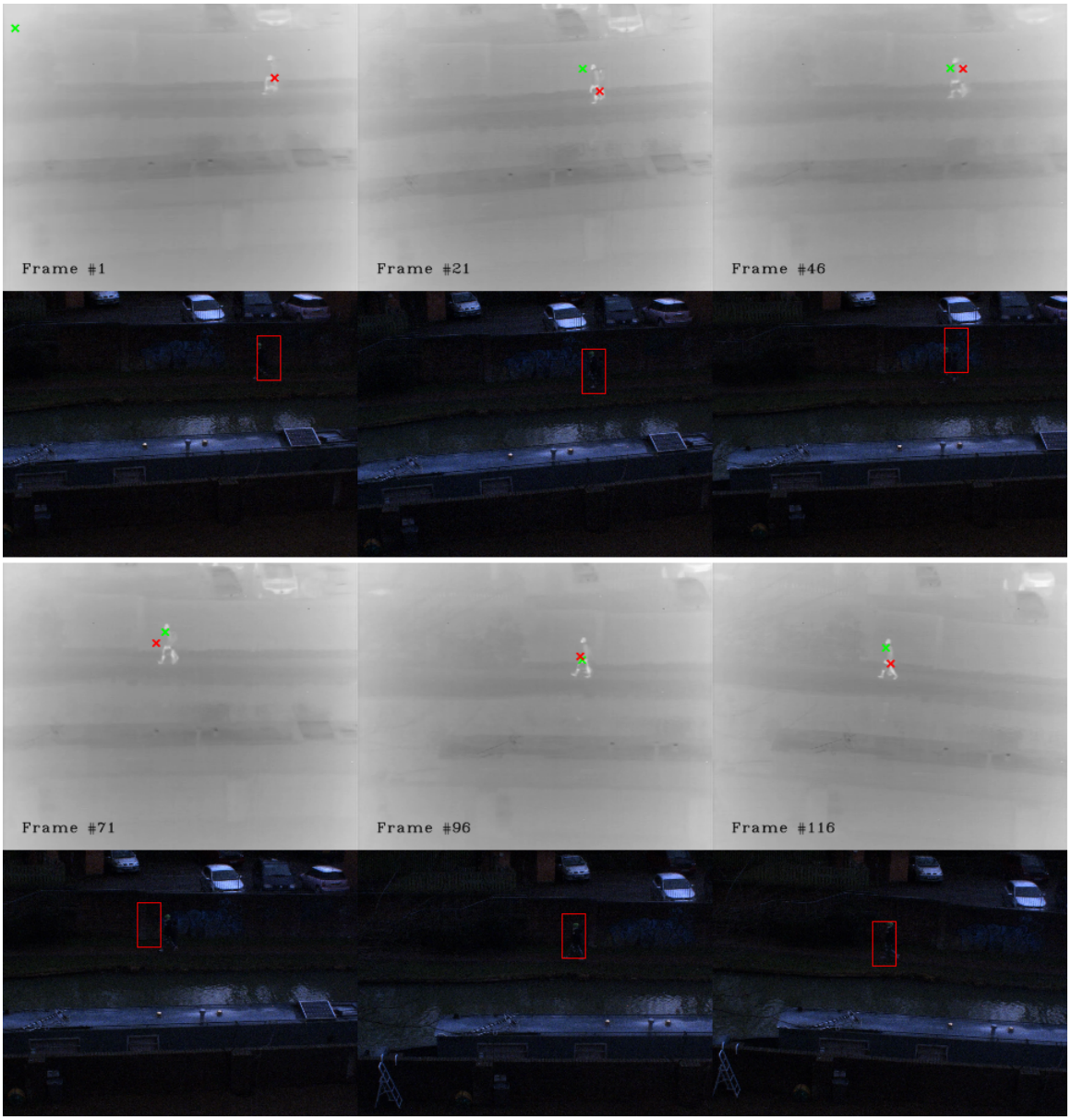



Figure A.32: A video taken at dusk. It is dark but there is still enough light to compute accurate homographies between consecutive visible frames. The infrared is driving the tracking here; without infrared, the visible light tracker will lose the person.


Figure A.33: Without a clear infrared signature, the system is reliant on the visible band to track the person. This is an example of tracker drift, where the tracker gradually includes more and more of the background in its internal representation until it finally loses the person.


Figure A.34: An example of mostly successful tracking until the person becomes occluded. In this video the person is never detected in infrared, and the system is totally reliant on the visible band. When the person goes behind the tree, the tracker loses them, and the system cannot correctly determine whether tracking has failed, or whether it should use the estimated location of the visible light tracker.


Figure A.35: An example of successful tracking, even though it was entirely dependent on the visible band.


Figure A.36: Similar to previous.

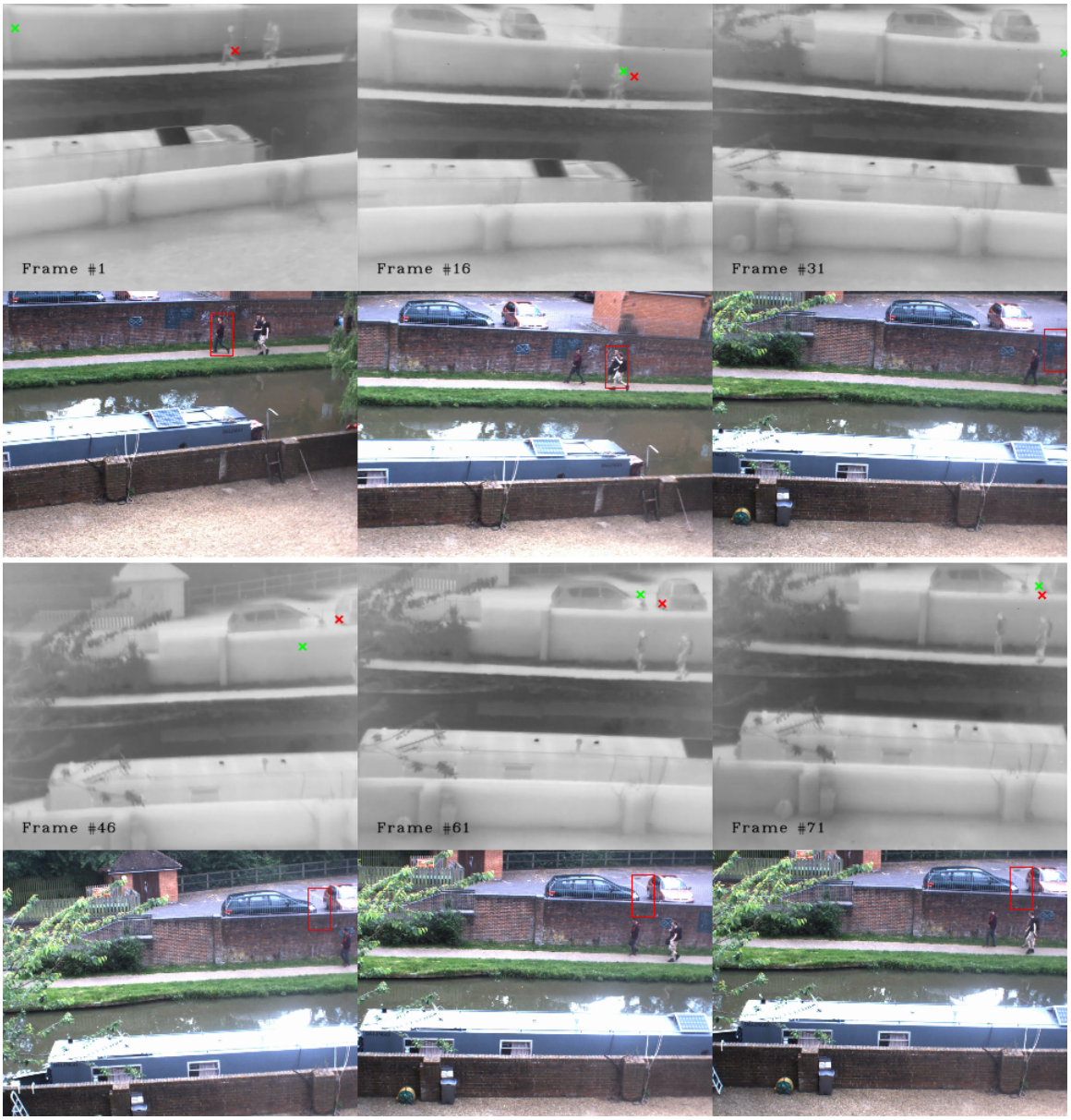



Figure A.37: A tricky example where there are no infrared measurements to help with tracking. Here, the visible light tracker fails very early on and cannot recover.

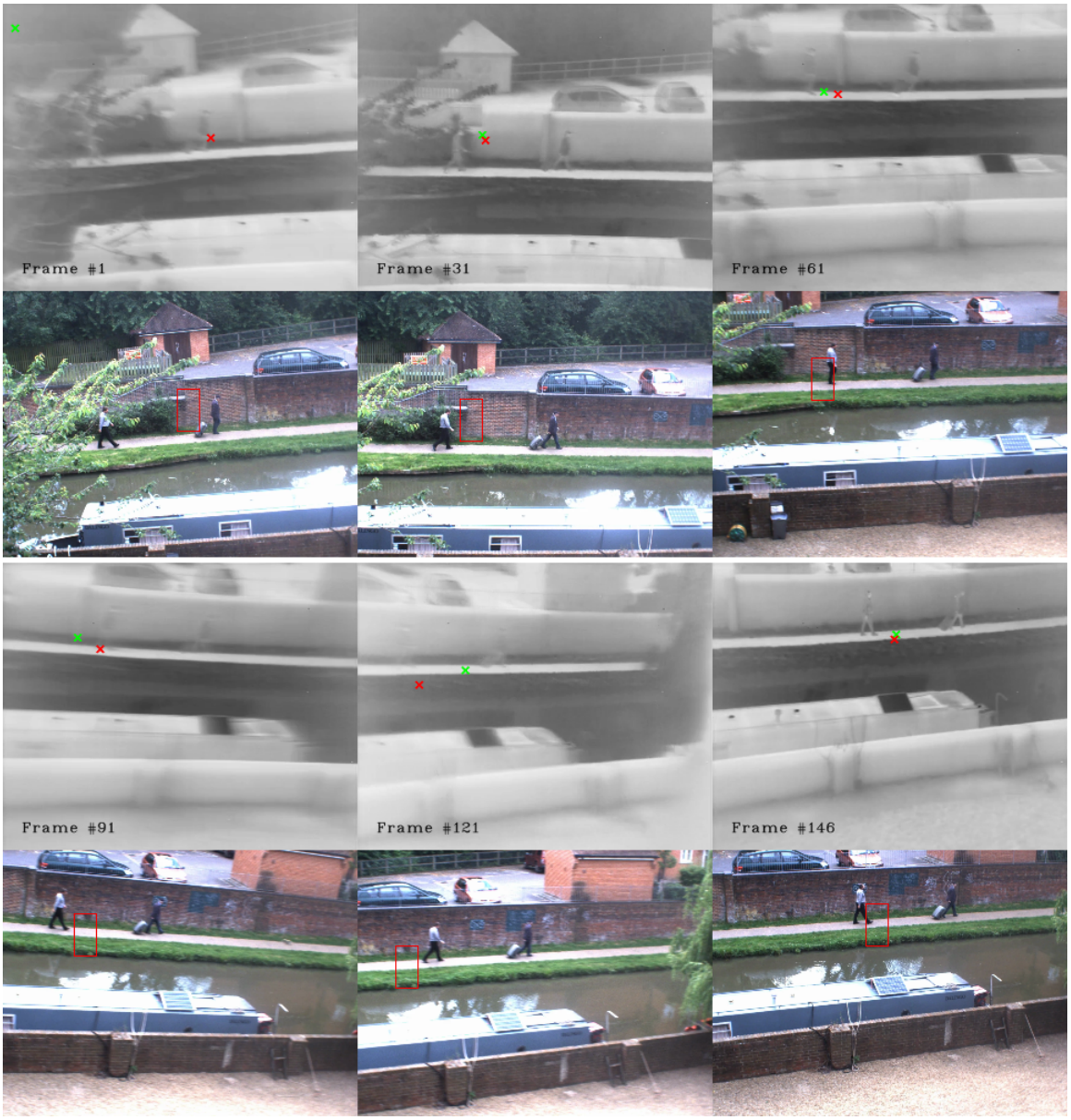



Figure A.38: An example of where an incorrect bounding box initialisation leads to tracking failure. The original bounding box is in the infrared image. The system looks for the corresponding location in the visible light image based on the closest HOG descriptor. The match is incorrect.


Figure A.39: This video was taken when it was -4°C outside and snowing – perfect conditions for tracking in infrared. It was dusk and, therefore, dark outside so the tracking is driven entirely by infrared.

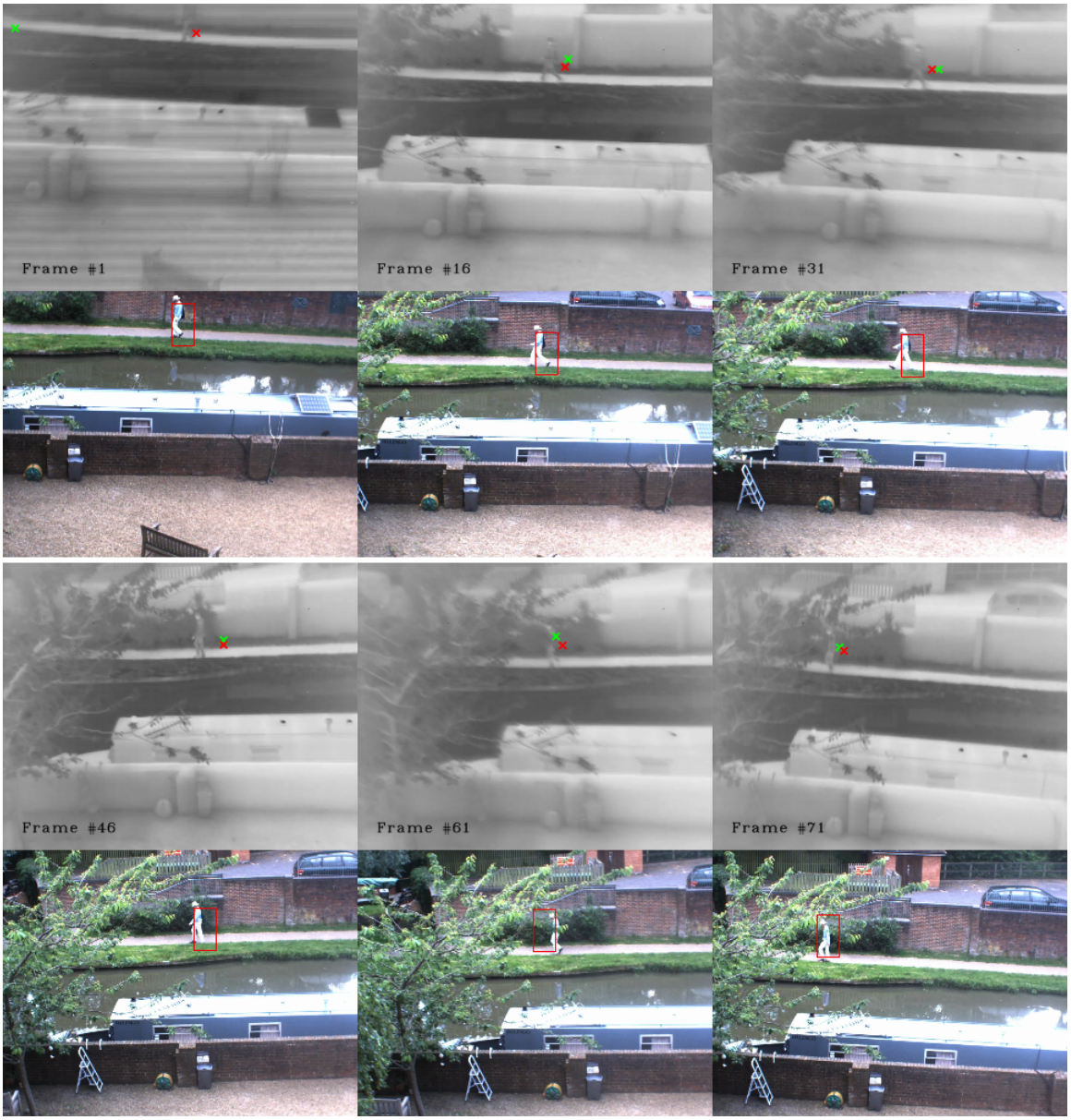



Figure A.40: An example of successful tracking.


Figure A.41: Similar to previous.


Figure A.42: Similar to previous.


Figure A.43: Similar to previous.


Figure A.44: Similar to previous.

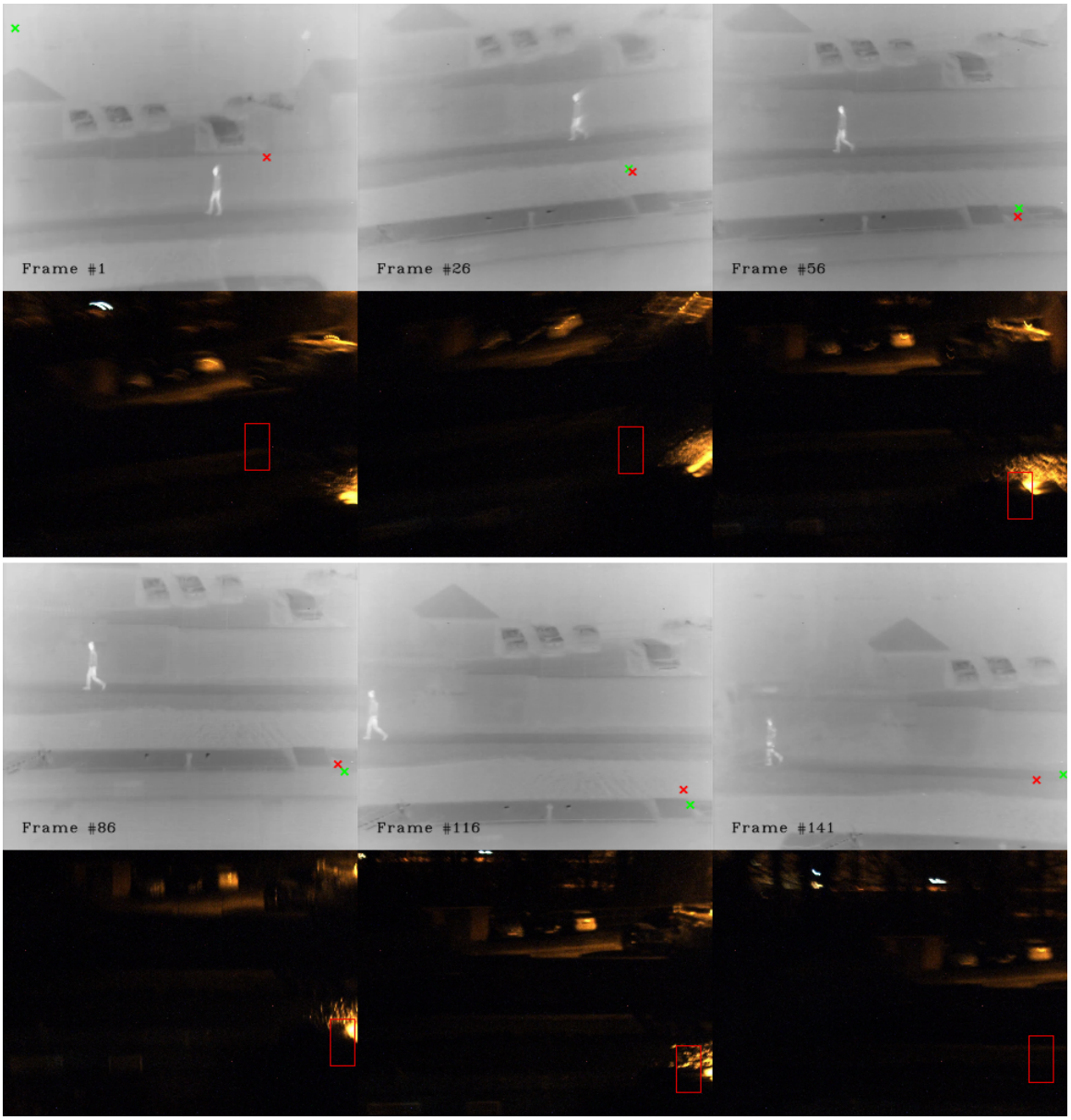



Figure A.45: This video is an example of how relying on the visible band alone to do motion estimates will result in tracking failure. It was taken at night, so there is not enough detail to compute homographies between consecutive frames. As the Kalman filter is reliant on measurements being converted into a fixed coordinate system using the estimated motion of the camera, tracking fails. This example calls for the use of alternative ways to estimate the camera motion between frames.


Figure A.46: An example of successful tracking.

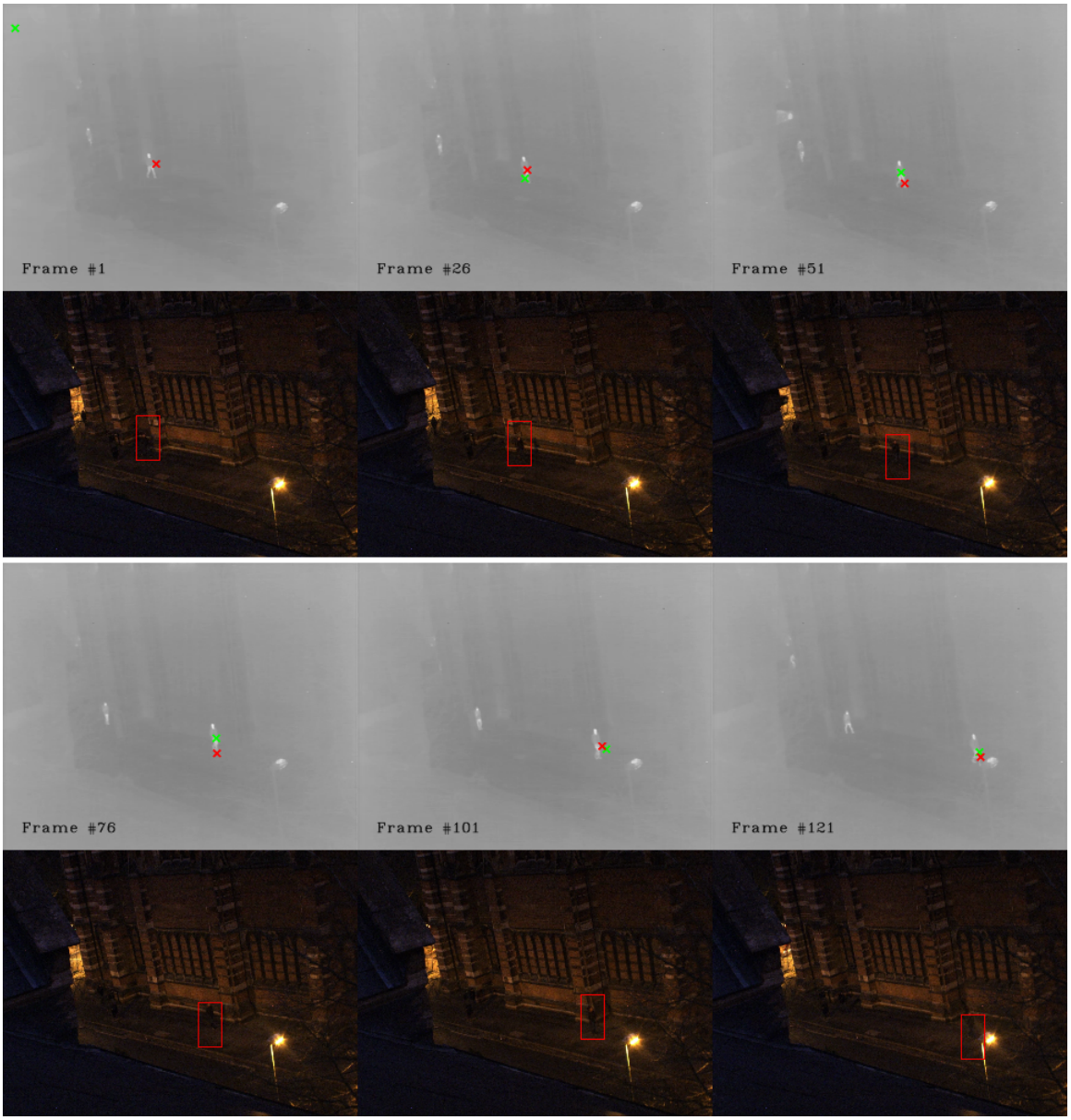



Figure A.47: This video was taken at dusk when light was fading. This and the next video are examples of how HeatTrack performs with a stationary camera in poor lighting conditions. Due to the fact that the camera was not moving, the identity matrix is used as the motion estimate of the camera and hence there are no errors arising out of incorrect motion estimates.

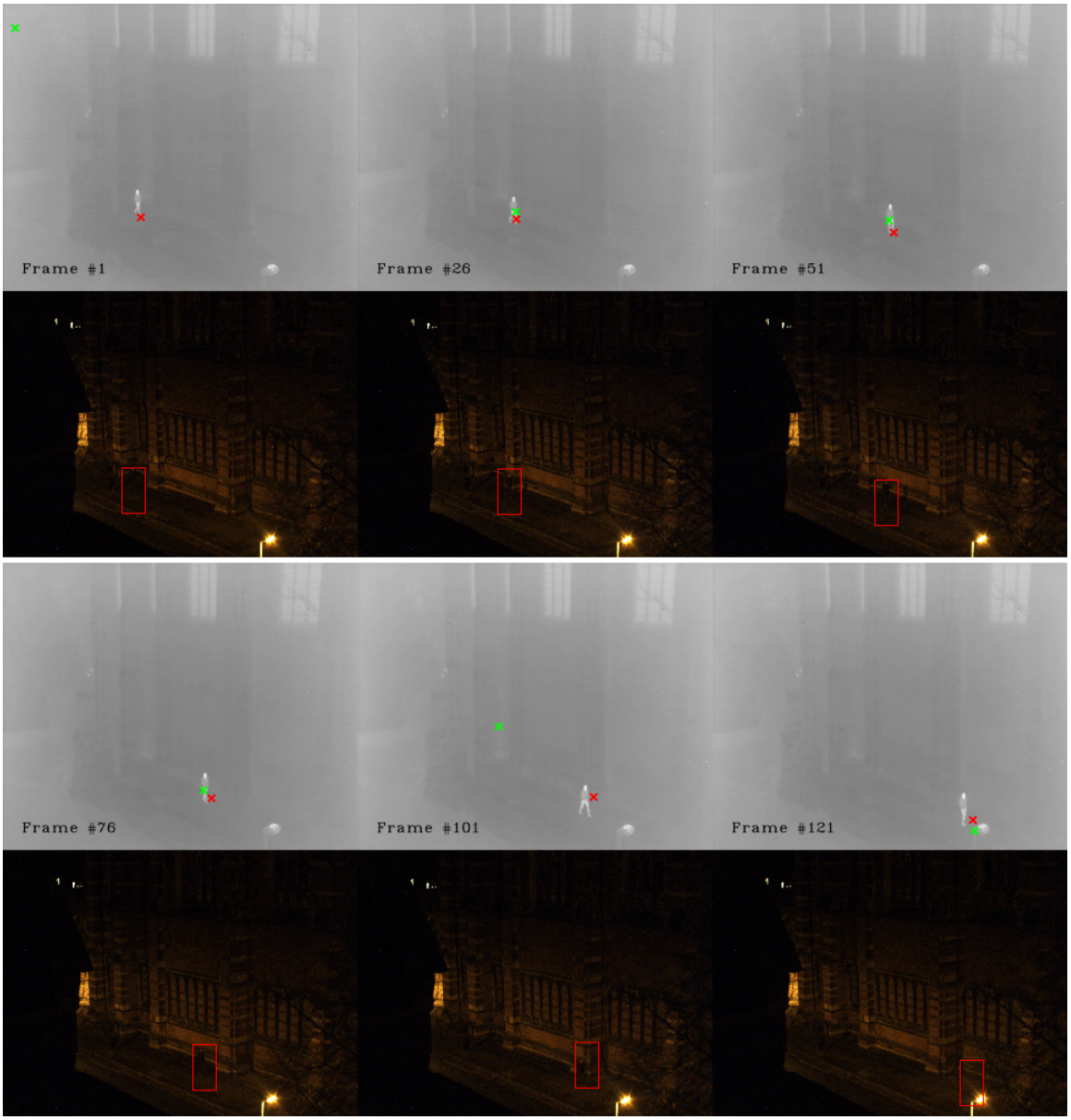



Figure A.48: Same as previous, but recorded 2 hours later.

# Bibliography


[1] Amit Adam, Ehud Rivlin, and Ilan Shimshoni. Robust Fragments-based Tracking using the Integral Histogram. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 798–805, 2006.

[2] Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen. Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:2037–2041, 2006.

[3] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. FREAK: Fast Retina Keypoint. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 510–517, 2012.

[4] Nicolas Alt, Stefan Hinterstoisser, and Nassir Navab. Rapid Selection of Reliable Templates for Visual Tracking. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1355–1362, 2010.

[5] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support Vector Machines for Multiple-Instance Learning. *Advances In Neural Information Processing Systems*, 15:561–568, 2003.

[6] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Pictorial Structures Revisited: People Detection and Articulated Pose Estimation. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pages 1014–1021, 2009.

[7] Mykhaylo Andriluka, Paul Schnitzspan, Johannes Meyer, Stefan Kohlbrecher, Karen Petersen, Oskar Von Stryk, Stefan Roth, and Bernt Schiele. Vision based Victim Detection from Unmanned Aerial Vehicles. In *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pages 1740–1747, 2010.

[8] ArduPlane. http://plane.ardupilot.com/. Accessed: January 2015.

[9] Shai Avidan. Support Vector Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:1064–1072, 2004.

[10] Shai Avidan. Ensemble Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:261–271, 2007.

[11] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual Tracking with Online Multiple Instance Learning. *IEEE transactions on pattern analysis and machine intelligence*, pages 983–990, 2010.

[12] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Robust Object Tracking with Online Multiple Instance Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:1619–1632, 2011.

[13] Chenglong Bao, Yi Wu, Haibin Ling, and Hui Ji. Real Time Robust L1 Tracker using Accelerated Proximal Gradient Approach. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1830–1837, 2012.

[14] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110:346–359, 2008.

[15] Rodrigo Benenson, Markus Mathias, Tinne Tuytelaars, and Luc Van Gool. Seeking the Strongest Rigid Detector. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3666–3673, 2013.

[16] Rodrigo Benenson, Mohamed Omran, Jan Hosang, and Bernt Schiele. Ten Years of Pedestrian Detection, What Have we Learned? In *Computer Vision for Road Scene Understanding and Autonomous Driving (CVRSUAD, ECCV workshop)*, pages 613–627, September 2014.

[17] Massimo Bertozzi, Alberto Broggi, Mike Del Rose, Mirko Felisa, Alain Rakotomamonjy, and Frederic Suard. A Pedestrian Detector using Histograms of Oriented Gradients and a Support Vector Machine Classifier. In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pages 143–148, 2007.

[18] Michael J. Black and Allan D. Jepson. EigenTracking: Robust Matching and Tracking of Articulated Objects Using a View-Based Representation. *International Journal of Computer Vision*, 26:63–84, 1996.

[19] Cristoph Bodensteiner, Wolfgang Huebner, Kai Juengling, Julius Mueller, and Michael Arens. Local Multi-modal Image Matching based on Self-Similarity. *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 937–940, 2010.

[20] Anna Bosch, Andrew Zisserman, and Xavier Muñoz. Scene Classification using a Hybrid Generative/discriminative Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:712–727, 2008.

[21] Lubomir Bourdev, Subhransu Maji, Thomas Brox, and Jitendra Malik. Detecting People using Mutually Consistent Poselet Activations. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6316 LNCS, pages 168–181, 2010.

[22] Toby P. Breckon, Anna Gaszczak, Jiwan Han, Martin L. Eichner, and Stuart E. Barnes. Multi-Modal Target Detection for Autonomous Wide Area Search and Surveillance. In *Proc. SPIE Emerging Technologies in Security and Defence: Unmanned Sensor Systems*, volume 8899, pages 1–19. SPIE, September 2013.

[23] Leo Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, August 1996.

[24] Leo Breiman. *Random Forests*, volume 45. Kluwer Academic Publishers, 1999.

[25] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time Tracking of Non-rigid Objects using Mean Shift. *IEEE Conference on Computer Vision and Pattern Recognition*, 2:142–149, 2000.

[26] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:564–577, 2003.

[27] CompuLab. http://www.fit-pc.com. Accessed: January 2015.

[28] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20:273–297, 1995.

[29] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, volume I, pages 886–893, 2005.

[30] James W. Davis and Mark A. Keck. A Two-Stage Template Approach to Person Detection in Thermal Imagery. In *Proceedings - Seventh IEEE Workshop on Applications of Computer Vision, WACV 2005*, pages 364–369, 2007.

[31] James W. Davis and Vinay Sharma. Background-Subtraction Using Contour-based Fusion of Thermal and Visible Imagery. *Computer Vision and Image Understanding*, 106:162–182, 2007.

[32] Simon Denman, Todd Lamb, Clinton Fookes, Vinod Chandran, and Sridha Sridharan. Multi-spectral Fusion for Surveillance Systems. *Computers and Electrical Engineering*, 36:643–663, 2010.

[33] Piotr Dollár, Ron Appel, Serge Belongie, and Pietro Perona. Fast Feature Pyramids for Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:1532–1545, 2014.

[34] Piotr Dollár, Serge Belongie, and Pietro Perona. The Fastest Pedestrian Detector in the West. *Procedings of the British Machine Vision Conference 2010*, pages 68.1–68.11, 2010.

[35] Piotr Dollár, Zhuowen Tu, Pietro Perona, and Serge Belongie. Integral Channel Features. *BMVC 2009 London England*, pages 1–11, 2009.

[36] Peter Dorninger and Norbert Pfeifer. A Comprehensive Automated 3D Approach for Building Extraction, Reconstruction and Regularization from Airborne Laser Scanning Point Clouds. *Sensors*, 8:7323–7343, 2008.

[37] Geoffrey Egnal. Mutual Information as a Stereo Correspondence Measure. Technical report, Department of Computer and Information Science, University of Pennsylvania, January 2000.

[38] Pedro F. Felzenszwalb, Ross B. Girshick, and David McAllester. Cascade Object Detection with Deformable Part Models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2241–2248, 2010.

[39] Pedro F. Felzenszwalb, Ross B. Girshick, David Mcallester, and Deva Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20, 2009.

[40] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance Transforms of Sampled Functions. *Cornell Computing and Information Science Technical Report TR20041963*, 4:1–15, 2004.

[41] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial Structures for Object Recognition. *International Journal of Computer Vision*, 61:55–79, 2005.

[42] Pedro F. Felzenszwalb, David McAllester, Ross B. Girshick, and Deva Ramanan. Visual Object Detection with Deformable Part Models. *Communications of the ACM*, 56:97, 2013.

[43] Pedro F. Felzenszwalb, David McAllester, and Deva Ramanan. A Discriminatively Trained, Multiscale, Deformable Part Model. In *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1–8, 2008.

[44] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: a Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24:381–395, 1981.

[45] Martin A. Fischler and Robert A. Elschlager. The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers*, C-22, 1973.

[46] Helen Flynn and Stephen Cameron. Multi-modal People Detection from Aerial Video. In *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013*, volume 226, pages 815–824. Springer, 2013.

[47] Helen Flynn and Stephen Cameron. Multi-modal People Detection from Aerial Video Footage. In *TAROS, Oxford*, pages 190–191, 2013.

[48] Thomas E. Fortmann, Yaakov Bar-Shalom, and Molly Scheffe. Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association. *IEEE Journal of Oceanic Engineering*, 8:173–184, 1983.

[49] Yoav Freund and Robert E. Schapire. Experiments with a New Boosting Algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.

[50] Yoav Freund and Robert E. Schapire. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. *Journal of Computing Systems and Science*, 55:119–139, 1997.

[51] Anna Gaszczak, Toby P. Breckon, and Jiwan Han. Real-time People and Vehicle Detection from UAV Imagery. In *Intelligent Robots and Computer Vision XXVIII: Algorithms and Techniques*, volume 7878, pages 78780B–1–13, 2011.

[52] Noel J. Gordon, David J. Salmond, and Adrian F. M. Smith. Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. *IEE Proceedings Radar and Signal Processing*, 140:107–113, 1993.

[53] Helmut Grabner, Michael Grabner, and Horst Bischof. Real-Time Tracking via On-line Boosting. *Technology*, 1:1–10, 2006.

[54] Alfred Haar. Zur Theorie Der Orthogonalen Funktionensysteme. *Mathematische Annalen*, 71:38–53, 1911.

[55] Gregory D. Hager and Peter N. Belhumeur. Efficient Region Tracking with Parametric Models of Geometry and Illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:1025–1039, 1998.

[56] Jan Han, Anna Gaszczak, Ryszard Maciol, Stuart E. Barnes, and Toby P. Breckon. Human Pose Classification within the Context of Near-IR Imagery Tracking. In *Proc. SPIE Optics and Photonics for Counterterrorism, Crime Fighting and Defence*, volume 8901, pages 1–10. SPIE, September 2013.

[57] Ju Han and Bir Bhanu. Fusion of Color and Infrared Video for Moving Human Detection. *Pattern Recognition*, 40:1771–1784, 2007.

[58] Sam Hare, Amir Saffari, and Philip H. S. Torr. Struck: Structured Output Tracking with Kernels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 263–270, 2011.

[59] Alastair Harrison and Paul Newman. TICSync: Knowing When Things Happened. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 356–363, 2011.

[60] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, Second edition, 2004.

[61] M. Heikkilä, M. Pietikäinen, and J. Heikkilä. A Texture-based Method for Detecting Moving Objects. *British Machine Vision Conference*, pages 21.1–21.10, 2004.

[62] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7575 LNCS, pages 702–715, 2012.

[63] Michal Irani, Benny Rousso, and Shmuel Peleg. *Recovery of Ego-motion Using Image Stabilization*. Technical report (Leibniz Center for Research in Computer Science). Department of Computer Science, Hebrew University of Jerusalem, 1993.

[64] Xu Jia, Huchuan Lu, and Ming Hsuan Yang. Visual Tracking via Adaptive Structural Local Sparse Appearance Model. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1822–1829, 2012.

[65] Kai Jüngling and Michael Arens. Feature based Person Detection Beyond the Visible Spectrum. In *2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009*, pages 30–37, 2009.

[66] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-Learning-Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:1409–1422, 2011.

[67] Rudol E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME-Journal of Basic Engineering*, 82:35–45, 1960.

[68] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*, pages 1–10, 2007.

[69] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pages 1–9, 2012.

[70] Stephen J. Krotosky and Mohan M. Trivedi. Mutual Information Based Registration of Multimodal Stereo Videos for Person Tracking. *Computer Vision and Image Understanding*, 106:270–287, 2007.

[71] Praveen Kumar, Ankush Mittal, and Padam Kumar. Fusion of Thermal Infrared and Visible Spectrum Video. In *ICVGIP 2006*, pages 528–539, 2006.

[72] Mikolaj E. Kundegorski and Toby P. Breckon. A Photogrammetric Approach for Real-time 3D Localization and Tracking of Pedestrians in Monocular Infrared Imagery. In *Proc. SPIE Optics and Photonics for Counterterrorism, Crime Fighting and Defence*, volume 9253, pages 1–16. SPIE, September 2014.

[73] Junseok Kwon and Kyoung M. Lee. Tracking of a Non-rigid Object via Patch-based Dynamic Appearance Modeling and Adaptive Basin Hopping Monte Carlo Sampling. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pages 1208–1215, 2009.

[74] Junseok Kwon and Kyoung M. Lee. Tracking by Sampling Trackers. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1195–1202, 2011.

[75] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86:2278–2323, 1998.

[76] Bastian Leibe, Ales Leonardis, and Bernt Schiele. Combined Object Categorization and Segmentation with an Implicit Shape Model. In *ECCV04 Workshop on Statistical Learning in Computer Vision*, pages 1–16, 2004.

[77] Bastian Leibe, Edgar Seemann, and Bernt Schiele. Pedestrian Detection in Crowded Scenes. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 878–885, 2005.

[78] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2548–2555, 2011.

[79] Rainer Lienhart and Jochen Maydt. An Extended Set of Haar-like Features for Rapid Object Detection. *Proceedings. International Conference on Image Processing*, 1:900–903, 2002.

[80] Andrew Litvin, Janusz Konrad, and William C. Karl. Probabilistic Video Stabilization using Kalman Filtering and Mosaicking. In *Proceedings of SPIE Conference on Electronic Imaging*, pages 663–674, 2003.

[81] David G. Lowe. Distinctive Image Features from Scale-invariant Keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[82] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. *Imaging*, 130:674–679, 1981.

[83] David Marr. Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. *Phenomenology and the Cognitive Sciences*, 8:397, 1982.

[84] Yasuyuki Matsushita, Eyal Ofek, Weina Ge, Xiaoou Tang, and Heung Yeung Shum. Full-Frame Video Stabilization with Motion Inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1150–1163, 2006.

[85] Iain Matthews, Takahiro Ishikawa, and Simon Baker. The Template Update Problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:810–815, 2004.

[86] Warren S. McCulloch and Walter Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. *The Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[87] Roland Miezianko and Dragoljub Pokrajac. People Detection in Low Resolution Infrared Videos. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops*, pages 1–6, 2008.

[88] Loris Nanni, Alessandra Lumini, and Sheryl Brahnam. Local Binary Patterns Variants as Texture Descriptors for Medical Image Analysis. *Artificial Intelligence in Medicine*, 49:117–125, 2010.

[89] Ciaran O'Conaire, Eddie Cooke, Noel O'Connor, Noel Murphy, and Alan Smeaton. Background Modelling in Infrared and Visible Spectrum Video for People Tracking. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, 2005.

[90] Ciaran O'Conaire, Noel E. O'Connor, and Alan Smeaton. Thermo-visual Feature Fusion for Object Tracking using Multiple Spatiogram Trackers. *Machine Vision and Applications*, 19:483–494, 2008.

[91] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution Gray-scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:971–987, 2002.

[92] Daniel Olmeda, Arturo de la Escalera, and Jose M. Armingol. Contrast Invariant Features for Human Detection in Far Infrared Images. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 117–122, 2012.

[93] Patrick Ott and Mark Everingham. Implicit Color Segmentation Features for Pedestrian and Object Detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 723–730, 2009.

[94] Wanli Ouyang and Xiaogang Wang. A Discriminative Deep Model for Pedestrian Detection with Occlusion Handling. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3258–3265, 2012.

[95] Wanli Ouyang and Xiaogang Wang. Joint Deep Learning for Pedestrian Detection. *2013 IEEE International Conference on Computer Vision*, pages 2056–2063, 2013.

[96] Constantine Papageorgiou and Tomaso Poggio. A Trainable System for Object Detection. *International Journal of Computer Vision*, 38:15–33, 2000.

[97] Dennis Park, Deva Ramanan, and Charles Fowlkes. *Multiresolution Models for Object Detection*, volume 6314. Springer Berlin Heidelberg, 2010.

[98] Marco Pedersoli, Andrea Vedaldi, and Jordi Gonzàlez. A Coarse-to-Fine Approach for Fast Deformable Object Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1353–1360, 2011.

[99] Peter Pinggera, Toby Breckon, and Bischof Horst. On Cross-Spectral Stereo Matching using Dense Gradient Features. In *Proc. British Machine Vision Conference*, pages 526.1–526.12, 2012.

[100] Jan Portmann, Simon Lynen, Margarita Chli, and Roland Siegwart. People Detection and Tracking from Aerial Thermal Views. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1794–1800, 2014.

[101] Bin Qi, Vijay John, Zheng Liu, and Seiichi Mita. Use of Sparse Representation for Pedestrian Detection in Thermal Images. In *The IEEE Conference on*

*Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 274–280, June 2014.

[102] Donald B. Reid. An Algorithm for Tracking Multiple Targets. *IEEE Transactions on Automatic Control*, 24:843–854, 1979.

[103] Vladimir Reilly, Berkan Solmaz, and Mubarak Shah. Geometric Constraints for Human Detection in Aerial Imagery. In *ECCV*, pages 252–265, 2010.

[104] Vladimir Reilly, Berkan Solmaz, and Mubarak Shah. Shadow Casting Out Of Plane (SCOOP) Candidates for Human and Vehicle Detection in Aerial Imagery. *International Journal of Computer Vision*, 101:350–366, 2013.

[105] Frank Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, 65:386–408, 1958.

[106] David A. Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental Learning for Robust Visual Tracking. *International Journal of Computer Vision*, 77:125–141, 2008.

[107] Rotorkonzept. https://www.rotorkonzept.de/enigma-portfolio/octocopter-rkm-8x-surveillor/. Accessed: May 2015.

[108] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2564–2571, 2011.

[109] Piotr Rudol and Patrick Doherty. Human Body Detection and Geolocalization for UAV Search and Rescue Missions using Color and Thermal Imagery. In *IEEE Aerospace Conference Proceedings*, pages 1–8, 2008.

[110] Pierre Sermanet, Koray Kavukcuoglu, Soumith Chintala, and Yann Lecun. Pedestrian Detection with Unsupervised Multi-Stage Feature Learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3626–3633, 2013.

[111] Arnold W. M. Smeulders, Dung M. Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual Tracking: An Experimental Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:1442–1468, 2014.

[112] Hyun Oh Song, Stefan Zickler, Tim Althoff, Ross Girshick, Mario Fritz, Christopher Geyer, Pedro F. Felzenszwalb, and Trevor Darrell. Sparselet Models for Efficient Multiclass Object Detection. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7573 LNCS, pages 802–815, 2012.

[113] Björn Stenger, Thomas Woodley, and Roberto Cipolla. Learning to Track with Multiple Observers. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pages 2647–2654, 2009.

[114] Peter Swerling. A Proposed Stagewise Differential Correction Procedure for Satellite Tracking and Prediction. Technical report, RAND Corporation, Boston, MA, USA, January 1958.

[115] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, first edition, 2010.

[116] Feng Tang, Shane Brennan, Qi Zhao, and Hai Tao. Co-tracking using Semi-supervised Support Vector Machines. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[117] Ascending Technologies. http://www.asctec.de. Accessed: January 2015.

[118] Michael Teutsch, Wolfgang Kruger, and Norbert Heinze. Detection and Classification of Moving Objects from UAVs with Optical Sensors. In *Signal Processing, Sensor Fusion, and Target Recognition XX*, volume 8050, pages 80501J–1–14, 2011.

[119] Michael Teutsch, Thomas Muller, Marco Huber, and Jurgen Beyerer. Low Resolution Person Detection with a Moving Thermal Infrared Camera by Hot Spot Classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2014.

[120] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A Real-time Algorithm for Mobile Robot Mapping with Applications to Multi-Robot and 3D Mapping. *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 1:321–328, 2000.

[121] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[122] Kinh Tieu and Paul Viola. Boosting Image Retrieval. *International Journal of Computer Vision*, 56:17–36, 2004.

[123] Atousa Torabi and Guillaume Alexandre Bilodeau. Local Self-Similarity as a Dense Stereo Correspondence Measure for Thermal-Visible Video Registration. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 61–67, 2011.

[124] Alexander Toshev and Christian Szegedy. DeepPose: Human Pose Estimation via Deep Neural Networks. *CoRR*, abs/1312.4659, 2013.

[125] Paul Viola and Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1:I–511–I–518, 2001.

[126] Paul Viola, Michael Jones, and Daniel Snow. Detecting Pedestrians using Patterns of Motion and Appearance. *International Journal of Computer Vision*, 63:153–161, 2005.

[127] Dong Wang, Huchuan Lu, and Ming Hsuan Yang. Online Object Tracking with Sparse Prototypes. *IEEE Transactions on Image Processing*, 22:314–325, 2013.

[128] Weihong Wang, Jian Zhang, and Chunhua Shen. Improved Human Detection and Classification in Thermal Images. In *Proceedings - International Conference on Image Processing, ICIP*, pages 2313–2316, 2010.

[129] Xiaoyu Wang, Tony X. Han, and Shuicheng Yan. An HOG-LBP human Detector with Partial Occlusion Handling. *Computer Vision, 2009 IEEE 12th International Conference on*, pages 32–39, 2009.

[130] Greg Welch and Gary Bishop. An Introduction to the Kalman Filter. Technical report, Department of Computer Science, University of North Carolina, January 2006.

[131] Christian Wojek and Bernt Schiele. A Performance Evaluation of Single and Multi-Feature People Detection. In *Pattern Recognition (DAGM)*, pages 82–91, May 2008.

[132] Bo Wu and Ram Nevatia. Optimizing Discrimination-efficiency Tradeoff in Integrating Heterogeneous Local Features for Object Detection. In *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1–8, 2008.

[133] Bo Wu, Ram Nevatia, and Li Zhang. Pedestrian Detection in Infrared Images based on Local Shape Features. *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, 2007.

[134] Yi Wu, Jongwoo Lim, and Ming Hsuan Yang. Online Object Tracking: A Benchmark. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2411–2418, 2013.

[135] Junjie Yan, Xucong Zhang, Zhen Lei, Shengcai Liao, and Stan Z. Li. Robust Multi-Resolution Pedestrian Detection in Traffic Scenes. In *Proceedings of the*

*IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3033–3040, 2013.

[136] Yi Yang and Deva Ramanan. Articulated Human Detection with Flexible Mixtures-of-Parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–15, 2012.

[137] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object Tracking: A Survey. *ACM Computing Surveys*, 38:1–45, 2006.

[138] Shanshan Zhang, Christian Bauckhage, and Armin B. Cremers. Informed Haar-like Features Improve Pedestrian Detection. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 947–954, 2014.

[139] Tianzhu Zhang, Bernard Ghanem, Si Liu, and Narendra Ahuja. Robust Visual Tracking via Multi-Task Sparse Learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2042–2049, 2012.