Queensland University of Technology
Brisbane Australia

This is the author's version of a work that was submitted/accepted for publication in the following source:

Ryan, David, Denman, Simon, Sridharan, Sridha, & Fookes, Clinton B. (2012)
Scene invariant crowd counting and crowd occupancy analysis.
In *Video Analytics for Business Intelligence [Studies in Computational Intelligence, Volume 409].*
Springer-Verlag, Germany, pp. 161-198.

This file was downloaded from: http://eprints.qut.edu.au/47054/

© Copyright 2012 Springer-Verlag

The original publication is available at SpringerLink http://www.springerlink.com

Notice: *Changes introduced as a result of publishing processes such as copy-editing and formatting may not be reflected in this document. For a definitive version of this work, please refer to the published source:*

http://doi.org/10.1007/978-3-642-28598-1_6

Scene Invariant Crowd Counting and Crowd Occupancy Analysis

David Ryan, Simon Denman, Sridha Sridharan, Clinton Fookes

Abstract In public places, crowd size may be an indicator of congestion, delay, instability, or of abnormal events, such as a fight, riot or emergency. Crowd related information can also provide important business intelligence such as the distribution of people throughout spaces, throughput rates, and local densities. A major drawback of many crowd counting approaches is their reliance on large numbers of holistic features, training data requirements of hundreds or thousands of frames per camera, and that each camera must be trained separately. This makes deployment in large multi-camera environments such as shopping centres very costly and difficult. In this chapter, we present a novel scene-invariant crowd counting algorithm that uses local features to monitor crowd size. The use of local features allows the proposed algorithm to calculate local occupancy statistics, scale to conditions which are unseen in the training data, and be trained on significantly less data. Scene invariance is achieved through the use of camera calibration, allowing the system to be trained on one or more viewpoints and then deployed on any number of new cameras for testing without further training. A pre-trained system could then be used as a turn-key solution for crowd counting across a wide range of environments, eliminating many of the costly barriers to deployment which currently exist.

David Ryan
Image and Video Laboratory, Queensland University of Technology, Brisbane, Australia, e-mail: david.ryan@qut.edu.au

Simon Denman
Image and Video Laboratory, Queensland University of Technology, Brisbane, Australia, e-mail: s.denman@qut.edu.au

Sridha Sridharan
Image and Video Laboratory, Queensland University of Technology, Brisbane, Australia, e-mail: s.sridharan@qut.edu.au

Clinton Fookes
Image and Video Laboratory, Queensland University of Technology, Brisbane, Australia, e-mail: c.fookes@qut.edu.au

1 Introduction

In large public places such as railway stations, airports, shopping centres and sporting events, it is often not possible to monitor every person's individual behaviour due to crowd size. Instead, crowd properties such as the distribution of people throughout the space, throughput rates and local densities can be monitored.

While an individual is capable of causing damage, a criminal will usually choose to do so in an uncrowded environment where he will not be caught. The threats posed in crowded environments are of a different nature, and arise from the crowd's collective properties: "a crowd is something other than the sum of its parts" [12]. These threats include fighting, rioting, violent protest, mass panic and excitement. The most common indicator of such behaviour is crowd size.

Even in peaceful crowds, size may be an indicator of congestion, delay or other abnormality. Crowd related information can also be used to provide important business intelligence. For example, the distribution of crowds throughout a shopping complex or large retail store may be used to analyse consumer shopping patterns, while the overall crowd size may be monitored to assess store performance over time. These measurements are difficult to collect and to quantify without employing considerable manpower, therefore researchers have turned to computer vision based surveillance technologies to collect this data automatically from closed-circuit television footage.

As crowd size is a *holistic* description of the scene, the majority of crowd counting techniques have utilised holistic image features to estimate crowd size. A major drawback of the holistic approach is the large amount of training data required. Due to the wide variability in crowd behaviours, distribution, density and overall size, it becomes necessary to annotate a very large number of frames in order to achieve proper generalisation. In a facility containing numerous cameras, each viewpoint must be trained independently. It is not practical to supply hundreds of frames of ground truth for every viewpoint.

In this chapter, we propose a novel scene-invariant approach that uses *local* features, which are specific to individuals and groups within an image, to estimate the crowd size and its distribution across a scene. While existing techniques have used similar features such as foreground pixels, they are analysed at a holistic level. Local features are used here to estimate the number of people within each *group*, so that the total crowd estimate is the sum of all group sizes. Because local features are used, training data must also be annoted with local information. We propose a unique method of localised ground truth annotation which allows the system to be trained rapidly, and greatly reduces the required training data.

Even though large-scale CCTV networks are becoming increasingly common, automated crowd counting is not widely deployed. One of the largest barriers to full deployment of this technology is the requirement to train each camera independently, which is both time-consuming and expensive. The algorithm proposed in this chapter uses camera calibration to scale features between multiple viewpoints, by taking into account the relative sizes of objects in these scenes. This results in a scene-invariant crowd counting system which may be trained on one camera and

then deployed for counting on another. In this chapter, we train our system on a large bank of data from various cameras, before testing it on a new viewpoint. In practice, a system which has been pre-trained on numerous camera viewpoints can operate as a turn-key solution for crowd counting across a wide range of unseen environments. Whenever more training data is annotated, the system may be re-trained on this larger training database, and then each deployment of the system can be upgraded accordingly.

The proposed system is tested on eight crowd counting datasets, including the UCSD database [7], PETS 2009 [2], PETS 2006 [1], and a custom database collected from our university campus (Figure 1). These datasets feature crowds of size 1-45 people, and capture a wide variation in scene properties, including lighting conditions, lens distortion, camera angle and camera distance. The proposed technique is compared to two holistic techniques, and is shown to outperform holistic techniques in terms of accuracy, scalability and practicality. The system is shown to be highly scalable, as it is capable of extrapolating to crowd sizes which are smaller or larger than those encountered during training; and highly practical, as it can count crowds when trained on as few as 10 frames of training data. Finally, the system is demonstrated to be scene invariant by performing training and testing on different datasets.

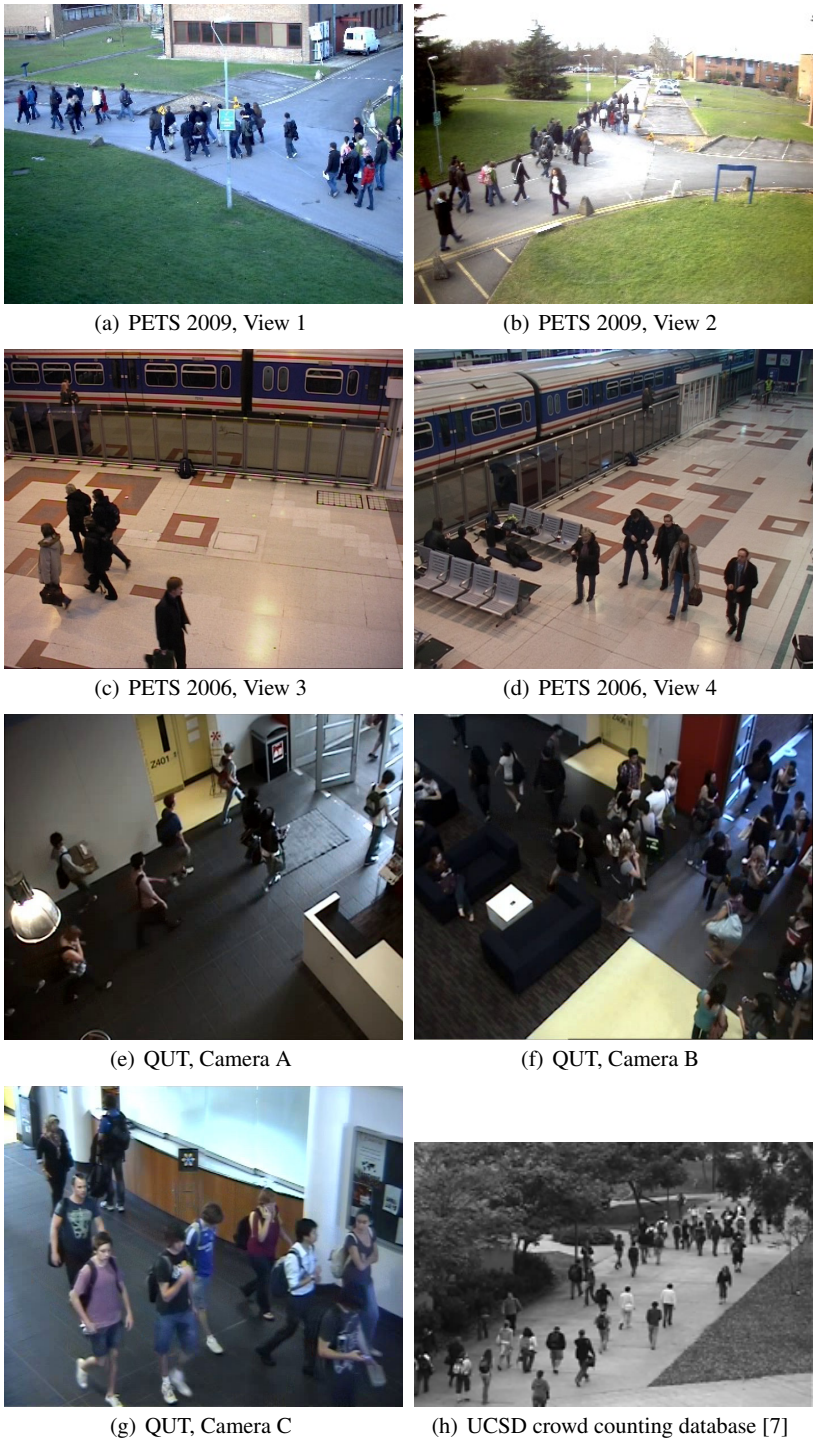
The remainder of the chapter is structured as follows: Section 2 provides an overview of existing crowd counting techniques, Section 3 outlines the proposed algorithm, Section 4 presents experimental results and Section 5 presents conclusions and directions for future work.

2 Background

Computer vision techniques to estimate crowd size generally involve feature extraction followed by a classification or regression stage. Important considerations include the quantity of features extracted, the complexity of the classifier, computation time required, and the size of the training data set. Each new feature introduces an additional level of dimensionality and therefore necessitates a wider set of training data, which reduces a system's practicality.

Background subtraction is a typical first step in automated visual surveillance, and has been used extensively in crowd counting because the foreground pixels in the image usually correspond to humans [12, 10, 30, 29, 25, 20, 6, 19, 37]. A number of other features have been employed such as textural information [35, 28, 27, 32, 42], while some systems use a combination of these features [7]. Earlier systems sought only to classify a crowd's size on a five-point scale (from 'very low' to 'very high' density) whereas recent approaches attempt to estimate the actual number of pedestrians in a scene.

Because crowd size is a holistic description of the scene, a common approach is to extract holistic image features which ideally describe its level of crowdedness. The basis for this approach is summarised by Davies [12]:



(a) PETS 2009, View 1 (b) PETS 2009, View 2

(c) PETS 2006, View 3 (d) PETS 2006, View 4

(e) QUT, Camera A (f) QUT, Camera B

(g) QUT, Camera C (h) UCSD crowd counting database [7]

Fig. 1 Eight datasets were used to evaluate our crowd counting algorithm.

Our objective for the models is that they should not involve actual counting of individuals or tracking of the movements of individuals but should be based on a collective description of crowds (e.g. analogous to the ideal-gas theory which ignores individual molecules).

The following review covers two general categories of crowd size monitoring:

1. Holistic approaches which utilise image features to obtain an estimate. These can also be described as "mapping-based" approaches, because they map directly between the feature space and the crowd size estimate. This works best for large crowds, in which the analogy to the ideal-gas theory holds. Holistic approaches are discussed in Section 2.1.
2. Local approaches which utilise local image features. These may sometimes be described as "detection-based" approaches, because they detect, track or otherwise classify pedestrians on an individual or group level. Local approaches are discussed in Section 2.2.

2.1 Holistic Approaches

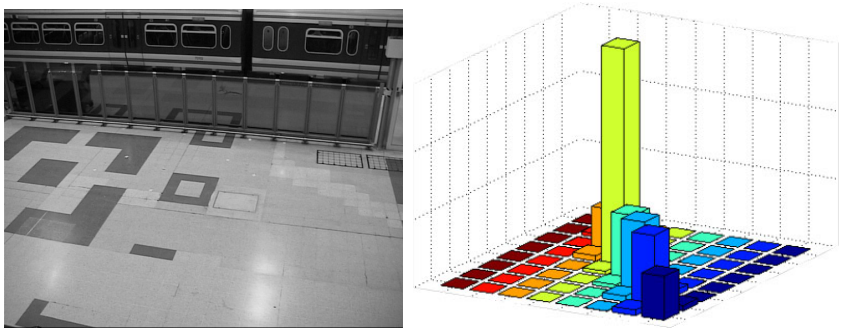
An early system developed by Marana [28, 26] used holistic image features for crowd density estimation, derived from the textures present in the image. The textures present in low density scenes tend to be course whereas those in high density scenes are fine. For this algorithm, it is assumed that the scene's background is relatively smooth compared to the human textures in the foreground. When considered with this assumption in mind, textures are helpful because the introduction of human crowding will disrupt those textures of the background. While textural information will be altered by the presence of crowding, it cannot explicitly segment the foreground from the background.

Haralick [16] proposed a number of well-known textural statistics, derived from the Grey Level Cooccurrence Matrix (GLCM), which measures the quantity of co-occuring pixel values in a greyscale image I, at a specified offset $\delta = (\delta_x, \delta_y)$:

$$G(r,c) = \sum_{(x,y)\in I} \begin{cases} 1 & \text{if } I(x,y) = r \text{ and } I(x+\delta_x, y+\delta_y) = c \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

A three dimensional illustration of the GLCM is depicted in Figure 2. It can be seen that the GLCM is a histogram of grey level cooccurrences, and that for a low frequency image such as the background in Figure 2(a), the histogram bins are greatest along the diagonal because grey levels of equal value frequently occur beside one another. When normalized (by dividing by the number of pixels in the image) the GLCM, G, becomes a second-order joint conditional probability density function, f, such that $f(r,c)$ is the probability of the grey levels r and c occurring beside one another.

From the normalized GLCM, holistic textural properties may be calculated. Those proposed by Haralick [16] include: contrast, homogeneity, energy and entropy. Using four different offset directions, Marana [28] applied these properties to



(a) (b)

Fig. 2 (a) Background image from PETS 2006 database [1]. (b) Three dimensional representation of the corresponding GLCM with $\delta = (1,0)$, after the image is quantized to 8 grey levels.

provide a total of sixteen holistic features for each image. The mapping from feature space to crowd density was performed using a Kohonen self organising map neural network. Five levels of crowd density were considered, from 'very low' to 'very high' density, and a correct classification rate of 82% was reported. Other texture-like holistic features include Minkowski Fractal Dimension [27] and Translation Invariant Orthonormal Chebyshev Moments [32]. These achieve similar results using a five-point scale of crowd density, in the range of 70-90% correct classification.

Holistic features such as these are highly sensitive to external changes, such as lighting conditions. Consequently, the system would have to be re-trained after any significant changes in the environment took place. For an indoor environment, it may be desirable to position a large number of cameras throughout a facility. Using holistic features such as these would require that each camera be trained independently, perhaps on hundreds of frames or more. For outdoor environments, the natural fluctuations in lighting between morning and afternoon have been shown to reduce system performance [32].

Xiaohua [43] proposed a pre-processing stage of histogram equalisation to compensate for changing lighting conditions and camera gain. This involves non-linearly mapping the pixel intensities to fully occupy the available range. Xiaohua also proposed the use of the 2D discrete wavelet transform (DWT) as a basis for extracting textural features. Correct classification of 95% was obtained on a testing database of 150 images, demonstrating improved performance compared to previous holistic methods.

A number of algorithms attempt to segment the foreground using background subtraction techniques. Davies [12] utilised a static 'reference' background image of a scene in which crowd levels were to be monitored. The reference image was subtracted from each frame before applying a threshold to extract a foreground boolean mask (similar to the one shown in Figure 3(b)). The relationship between the number

(a) (b) (c)

Fig. 3 (a) Frame 1280 of the UCSD crowd counting database [7]. (b) Corresponding motion segmentation (foreground detection), obtained using an adaptive background model [14, 15]. (c) Region of interest for this scene.

of people in the scene and the total number of foreground pixels was approximately linear, while the correlation with the total number of edge pixels was not as strong.

The crowd estimate was therefore obtained by using linear regression to model the relationship between foreground pixels and crowd size, and filtering the output in time. The mean relative error was less than 8%. However, the use of a static background image means that the system is sensitive to lighting changes over longer periods of time, whether sudden or gradual. Adaptive background models such as Stauffer-Grimson [39], Zivkovic [46] and Denman [14, 15] are robust against such changes, and have been adopted in more recent crowd counting applications [25, 20, 36, 37].

In practice, the use of foreground detection and linear regression is not sufficient for counting crowds, due to the effects of perspective and occlusion. Paragios [30] introduced the use of a density estimator to account for perspective in an image. The focus of this approach was on change detection rather than crowd counting. Perspective was also considered by Ma [25], who computed a 'density map' from four points in an image corresponding to the corners of a rectangle on the ground plane. The density map weighted each pixel according to the area it represented on the ground plane. Hou [17] utilised this approach and estimated the crowd size using a neural network, with the weighted foreground pixel count as an input.

Kong [20] proposed the use of histogram based features to capture the various levels of occlusion present in a scene. The foreground detection result was divided into distinct segments, commonly referred to as 'blobs', using a connected components algorithm. A blob size histogram was then populated to capture the range of object sizes in the image. It is expected that individual pedestrians and small groups contribute to the lower histogram bins, while groups of larger sizes contribute to the higher ones. The histogram is a holistic description of the scene, but it captures more directly the range of blob sizes present in the image.

Kong also used the Canny edge detector [5] to extract edge pixels and their angle of orientation. These pixels are masked by the foreground so that those edges in the background are ignored. An edge angle histogram is constructed with eight bins between 0° and 180°. The edge orientation histogram "can distinguish edges caused by pedestrians, which are usually vertical, with other scene structures such

as noise, shadows and cars" [20]. There is support for this statement in other visual surveillance research. For example, Dalal [11] described a similar descriptor called the histogram of oriented gradients (HOG), which uses edge detection to populate a histogram of edge orientations for the explicit purpose of human detection.

Wu [42] used the textures present inside 'multi-resolution density cells', spaced across an image at various locations. Within each cell, the textural features used previously by Marana [28] were calculated from the GLCM, enabling the system to estimate crowding for that particular region of the scene. The purpose of this system was to detect local abnormality due to overcrowding or undercrowding. These conditions were detected using a support vector machine (SVM) classifier.

A unique segmentation technique was used by Chan [7, 9] to identify foreground motion. The segmentation is based on dynamic textures (an extension of textures into the temporal domain). For the purpose of crowd counting, Chan considers an outdoor pathway on which pedestrians were classified as walking either toward or away from the camera. Treated as different instances of dynamic textures, these two classes of pedestrian were segmented from the background and from one another.

Chan extracts a large number of holistic image features from the foreground mask for each direction, including foreground area, perimeter pixel count, edge orientation histogram and textural features. In total, 29 features are extracted and Gaussian Process regression is used to predict the number of pedestrians walking in each direction. While this algorithm counts crowds with high accuracy, it comes at the expense of additional training data requirements. The implementation described by Chan utilised 800 frames of training data, which were be manually annotated with ground truth (the number of pedestrians moving in each direction). This would be a burdensome task to perform for every camera in a large facility where crowd size monitoring was required.

Also, dynamic textures can only segment moving pedestrians, and not those who have stopped in the middle of the scene. Pedestrians stop frequently in surveillance footage, and this can even be caused by excessive congestion, which is what we seek to detect in the first place. An adaptive background model such as [15] can continue to detect stationary objects for some time after they have come to a stop.

2.2 Local Approaches

Local approaches utilise detectors or features which are specific to individuals or groups of people within an image. These groups are independently analysed, so that the total crowd estimate is the sum of its parts.

Lin [23] has proposed the use of head detection for crowd counting. The Haar wavelet transform (HWT) is used in conjunction with the Support Vector Machine (SVM) to classify head-like contours as either a human head or not. This approach may be useful in dense crowds where only the head of each individual is visible.

Dalal [11] introduced the histogram of oriented gradients to represent images, using an SVM classifier to detect humans. Tuzel [41] introduced the use of co-

variance features as human descriptors, which may be represented as a connected Reimannian manifold. Classification is then performed using a boosting algorithm.

Celik [6] proposes a person-counting algorithm which does not require training. It assumes proportionality between the number of pixels within a blob segment and the number of people represented by that segment, in order to obtain an estimate for each group. Kilambi [18, 19] models a group of pedestrians as an elliptical cylinder, assuming a constant spacing between people within the group. Tracking a large blob over several frames increases the robustness of the group size estimate. It is unclear how the cylinder model assumption would hold up in larger crowds under various configurations, such as that shown in Figure 3(b).

Rabaud [31] used a parallelised LKT tracker [38] to determine partial tracks of interesting feature points across an image. These trajectories are then conditioned and clustered in order to estimate the number of pedestrians walking in a scene. Similarly, Yang [44] used the KLT tracker to establish trajectories of people entering a door from an overhead camera; in this case, counting was performed as a cumulative total over time, rather than in a single image.

Lempitsky [22] proposes an object counting algorithm which sought to estimate a *density function* F, as a function of the pixel intensities in an image, so that integrating the density over any region would yield the number of objects in that region. This is a localised approach in which every pixel $\mathbf{p}$ is represented by a feature vector $\mathbf{x_p}$, containing local foreground and gradient information. A linear model is used to obtain the density at each pixel, $F(\mathbf{p}) = \mathbf{w}^{\mathrm{T}}\mathbf{x_p}$, so that the count is obtained across a region of interest R by integrating over F as follows: $\sum_{\mathbf{p}\in R} F(\mathbf{p})$.

3 Scene Invariant Crowd Counting

Existing approaches to crowd counting are scene specific, as they are designed to operate in the same environment that was used to train the system. Because crowd size is a *holistic* description of a scene, the majority of crowd counting techniques have utilised holistic features to estimate crowd size. However, due to the wide variability in crowd behaviours, distribution, density and overall size, holistic systems require a very large training set in order to generalise properly. Indeed, some methods have substantial training requirements of hundreds [12, 20] or even thousands [9] of frames. It would not be practical to annotate this many frames for each installation.

In this chapter, a novel scene-invariant approach is presented which uses *local* features, defined here as features which are specific to an individual or group within an image. While existing techniques have used similar local features such as foreground pixels, they are analysed at a holistic level. Local features are used here to estimate the number of people within each *group*, so that the total crowd estimate is the sum of all group sizes. As local features are used, training data must also be annotated with local information. To provide appropriate training data, a unique

method of localised ground truth annotation has been developed which greatly reduces the required training data.

As well as the reduced training requirement, a localised approach also enables the estimation of crowd densities at different locations *within* the scene (unlike holistic systems, which can only provide a density for the whole scene), and allows for a simplistic extension to a multi-camera environment. The ability to determine local crowd densities greatly improves the systems ability to detect abnormalities in a scene. While the overall number of people in a scene may be considered normal, there may be an abnormally high concentration of people in a small area. Holistic systems are unable to detect such an abnormality, however the proposed local approach can detect such an occurrence.

The proposed approach also utilises camera calibration to achieve scene invariance by scaling features appropriately between viewpoints.This enables the system to be deployed on different training and testing sets.

The remainder of this section is structured as follows: Section 3.1 describes the background subtraction and group detection algorithm; Section 3.2 discusses camera calibration, perspective normalisation and how this relates to scene invariance; Section 3.3 explains the feature extraction process; Section 3.4 presents a ground truth annotation strategy that greatly simplifies the training process; Section 3.5 details the chosen regression model; and Section 3.6 proposes an extension to our system which uses group tracking to refine and improve the crowd size estimate.

3.1 Background Subtraction and Group Detection

The crowd counting system presented in this chapter has been developed using local rather than holistic features. These features are 'local' with respect to the blob segments in a foreground mask, obtained using a foreground segmentation technique proposed by Denman [13, 14, 15].

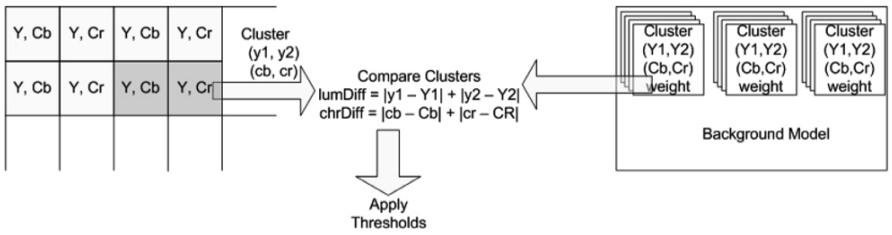


Fig. 4 Pixel pairs are grouped into clusters. The adaptive background model contains a group of clusters, each assigned a weight indicating its likelihood, against which incoming clusters are compared.

This background segmentation routine operates in the YCbCr 4:2:2 colour space, which provides some invariance to lighting changes through the separation of colour

and intensity. Each pixel in the incoming image, I, has two values: a luminance and a single chrominance, which alternates between blue chrominance and red chrominance in the horizontal direction (Figure 4). Pixels are paired horizontally so that for each pair there are four values (two luminance, one blue chrominance and one red chrominance):

$$P(i,j,t) = [y_1, y_2, c_b, c_r] \tag{2}$$

where $P(i,j,t)$ denotes a pixel pair, or 'cluster', formed by grouping the two pixels, $I(2i,j,t)$ and $I(2i+1,j,t)$. This pairing results in motion detection being effectively performed at half the horizontal resolution of the original image, with the benefit being increased speed.

A multi-modal background model is then constructed, for each pixel pair, by storing a set of possible modes representing the distribution of colours at that location (Figure 4). These are stored as a group of clusters, each accompanied by a weight, w_k, where k is used to denote the mode. The weight describes the likelihood of the colour described by that cluster being observed at that position in the image. Each cluster in the background model is represented by:

$$C(i,j,t,k) = [Y_{1k}, Y_{2k}, C_{bk}, C_{rk}, w_k] \tag{3}$$

Clusters in the background model are stored in order of highest to lowest weight. Incoming clusters, $P(i,j,t)$, are compared to all possible modes, $C(i,j,t,k)$, to determine a match. A match is found by finding the highest-weighted mode which satisfies:

$$|Y_{1k} - y_1| + |Y_{2k} - y_2| < T_{Lum} \tag{4}$$

$$|C_{bk} - c_b| + |C_{rk} - c_r| < T_{Chr} \tag{5}$$

where T_{Lum} and T_{Chr} denote the luminance and chrominance thresholds, respectively. Foreground motion is detected if the probability of the matching mode, m, falls below a threshold, T_{fg}:

$$F(i,j,t) = \begin{cases} 1 & \text{if } \sum_{k=0}^{m} w_k < T_{fg} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The matching cluster in the background model is adjusted to reflect the current pixel colour, and the weights of all clusters in the model at this location are adjusted and normalised to reflect the new state [15]. If no match is found, then the lowest weighted cluster is replaced with a new cluster representing the incoming pixels (and foreground is detected at this location). Clusters are gradually adjusted and removed as required, allowing the system to adapt to slow changes in the background.

In surveillance situations, particularly outdoor scenarios, lighting levels can also change rapidly resulting in large amounts of erroneous motion. When these levels fluctuate, it is the luminance values in an image which undergo significant change, whereas chrominance values remain relatively unchanged. Therefore, to improve performance in real world lighting conditions the luminance threshold T_{Lum} can be

adjusted. It would be ideal to be able to use a single value to adjust for the luminance change in a given frame. However, as the luminance change is not constant across a scene, images are divided into several small regions, and each is treated separately. (In our application, images are divided into a 5×5 grid of sub-regions.)

Thresholds for detection are varied within each region, according to the lighting conditions in that part of the scene. We define the luminance difference, Δ_{Lum}, at a cluster to be:

$$\Delta_{Lum} = |Y_{1m} - y_1| + |Y_{2m} - y_2| \tag{7}$$

where Y_{1m} and Y_{2m} denote the luminance values of the matching mode m, and y_1 and y_2 are the luminance values of the incoming cluster. Attaching coordinate and time information, we use $\Delta_{Lum}(i,j,t)$ to represent the luminance difference at a specific frame and location. Thus the weighted average of luminance changes is calculated across an image region, R:

$$O_{Lum}(R,t) = \frac{\sum_{(i,j)\in R} \Delta_{Lum}(i,j,t) \times w_m(i,j,t)}{\sum_{(i,j)\in R} w_m(i,j,t)} \tag{8}$$

The use of weighted sum allows pixels that are only recently created, potentially under the present lighting conditions, to be weighted less relative to those that have been present longer. An acceptable range for the luminance offset at time t, with respect to the previous frame $(t-1)$, is defined:

$$\chi \leq \frac{O_{Lum}(R,t)}{O_{Lum}(R,t-1)} \leq \frac{1}{\chi} \tag{9}$$

where $\chi \in [0,1]$ is the change threshold for the luminance offset. If the change in luminance offset falls outside of this acceptable range, a rapid fluctuation in luminance has been detected across the region, and equation 4 is modified as follows:

$$|Y_{1k} - y_1| + |Y_{2k} - y_2| < T_{Lum} + O_{Lum} \tag{10}$$

where O_{Lum} is the luminance offset of the region to which the cluster being matched belongs. Loosening the threshold enables improved performance when dealing with both global lighting changes (such as changes in camera gain), or local changes such as variable cloud cover. This approach is robust in various environments, including both indoor and outdoor scenes such as those shown in Figure 1. The full details of this background model are presented in [13, 14, 15].

Following foreground detection, a morphological closing operation is applied to the binary mask in order to obtain 'cleaner' and less fragmented blob segments. A connected components algorithm is subsequently run to identify the locations of each blob in an image.

In the proposed crowd counting algorithm, a crowd estimate is obtained for each blob in an image, so that the total estimate for the scene is the sum of the estimates for each individual blob. In order to train the system, ground truth annotation is performed *after* the first stage of image processing, once the foreground is extracted.

The group size is explicitly labeled for each blob in an image, therefore each frame provides several instances of ground truth. The details of this annotation process are discussed in Section 3.4.

This approach is built on the intuition that it is easier for a system to estimate the number of people in each group than to estimate the entire crowd at once. It is possible for a crowd of twenty people to be distributed as two large groups or as ten pairs (for example). Viewed from a holistic perspective, these various crowd distributions can give rise to vastly different image features. Existing techniques cope by extracting a larger quantity of holistic features (for example, 29 features are used by Chan [7]), necessitating more training data and intensive classification strategies. We have found that the relationship between image features and group size is more reliable and consistent when analysed on a local scale. Results comparing the local and holistic approaches are presented in Section 4.1.

3.2 Camera Calibration and Perspective Normalisation

Before local features can be used for crowd counting, the effects of perspective and camera distortion must be taken into account. The algorithm presented in this chapter is also designed to operate over multiple viewpoints, therefore the features selected to represent a person or group should be invariant to camera distance and illumination properties. Indeed, these properties are also desirable in a single-viewpoint system for the following reasons:

1. Certain objects appear closer to the camera than others.
2. The angle of observation from the camera to an object, with respect to the ground plane, is not constant throughout an image. It can vary greatly, particularly when the camera is placed close to the scene (Figure 5).
3. Illumination can change within a scene over time.

It is therefore important that features are normalised appropriately so that the trained system can count other objects within the scene as well as objects in other scenes. Two methods are proposed to achieve this:

1. Camera calibration is used to compensate for changes in camera positioning.
2. Recent advances in adaptive background modeling are used to compensate for changes in illumination. (Section 3.1)

A number of camera calibration methods have been described [3, 40, 45], although the most popular of these is Tsai's model [40], which is frequently used on visual surveillance databases such as PETS 2006 and PETS 2009 [1, 2]. Tsai's model incorporates camera position, rotation angle, focal length and radial lens distortion parameters to map between the real-world coordinate system (x, y, z) and the image plane coordinate system (i, j). These parameters are estimated from a manually-specified set of point correspondences between image pixels and real-world locations on the ground plane.

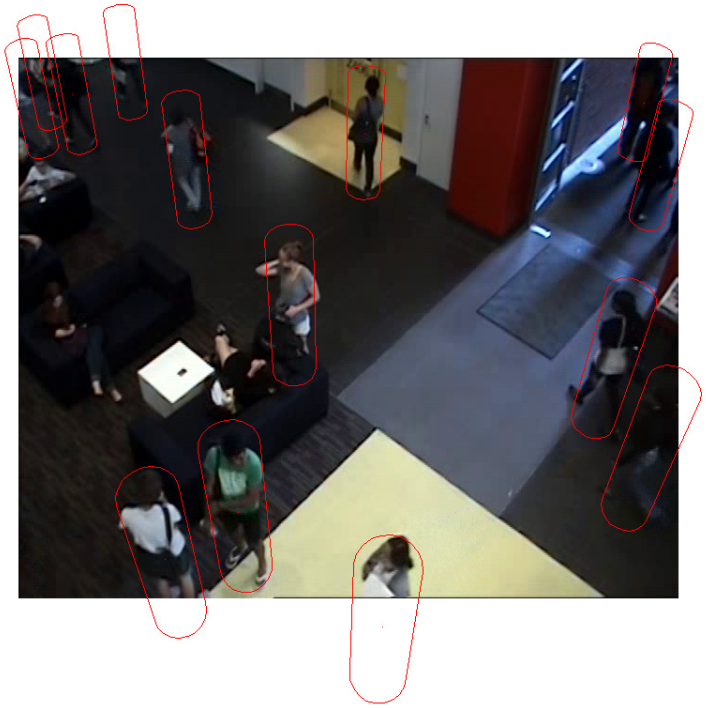


Fig. 5 Ground truth annotations on Camera B of the QUT dataset. A camera calibration technique [40] is used to project a human-sized cylindrical object into the scene. Perspective normalisation is performed by compensating for the area of these projections at any location in the image.

In addition to Tsai's calibration model, a number of automated procedures exist for estimating camera calibration based on human or object tracking [4, 24, 21]. For example, Lv [24] proposes a camera calibration model using vanishing points, and describes a self-calibration method based on moving humans where the head and feet positions can be located in multiple frames. Similarly, Krahnstoever [21] presents a Bayesian autocalibration algorithm which includes uncertainty estimates for each of the camera parameters. These approaches could readily be incorporated into our proposed framework to create a truly 'turn-key' crowd monitoring system. However, as Tsai calibration parameters are already available for public visual surveillance datasets, and the method is widely used and well understood, we continue to use this model in our experiments.

As outlined in Section 2.1, a common approach to perspective normalisation is to calculate a density map which assigns to each pixel a weight to compensates for perspective [30, 25, 9]. Typically, a reference pixel near the bottom of the image is assigned the value 1.0 and all other pixels are weighted with respect to this reference.

For example, pixels higher in the image will be given a larger value because they represent a greater area in the scene.

In the proposed system, a cylinder model to approximate the size of a human, with radius $r = 0.25$ and height $h = 1.7$ metres. As depicted in Figure 5, this cylinder may be projected into a scene centered around any pixel (i, j). The area of this projected shape in the image plane is denoted $S(i, j)$. This procedure is used to generate a density map D_2 which provides a weight to each pixel inversely proportional to the projected area of an object centered at that location:

$$D_2(i,j) = \frac{1}{S(i,j)} \tag{11}$$

This density map, D_2, provides a weight to each pixel so that an object occupying a set of pixels, B, has a weighted area of $\sum_{(i,j)\in B} D_2(i,j)$. Consequently distant objects occupying fewer pixels are compensated by their larger weights in the density map. The calibrated density map is advantageous because it is defined in terms of real-world objects rather than an arbitrary reference pixel. This approach can scale readily between different camera angles and is inherently scene invariant.

It does not matter that the cylinder model does not match a human size or shape precisely, as its role is only to normalise 2D and 1D features across viewpoints. A number of such features are described in Section 3.3, including the weighted area of each blob. The density map D_2 is suitable for such two-dimensional features as area. However, one-dimensional features such as perimeter and edges are also considered, therefore a density map D_1 for these features is also defined:

$$D_1(i,j) = \sqrt{D_2(i,j)} = \frac{1}{\sqrt{S(i,j)}} \tag{12}$$

If camera calibration is unavailable, a density map can be estimated using alternative approaches such as the method described by Chan [7]. Because this approach utilises a reference pixel instead of a real-world reference (e.g. cylinder model), it is not suitable for performing *scene invariant* crowd counting. However, the system may be trained and tested on the same viewpoint, and retains all of the other benefits of a local features based approach to crowd counting.

3.3 Feature Extraction

Several features are extracted from each blob segment to estimate the number of people in the group. The features extracted are similar to those used by Kong [20] and Chan [7], taken at a local level rather than the holistic level. We enumerate all of the blobs in a scene's region of interest using the subscript n, so that B_n and P_n denote the set of pixels inside the nth blob, and on its perimeter, respectively. The following features are extracted from each blob:

- **Area:** The weighted area of the nth blob segment is calculated using the two-dimensional density map D_2:

$$A_n = \sum_{(i,j) \in B_n} D_2(i,j) \tag{13}$$

where B_n denotes the set of pixels in the blob. The area directly captures the size of the object, normalised for perspective.

- **Perimeter:** The weighted perimeter of the nth blob segment is calculated using the one-dimensional density map D_1:

$$L_n = \sum_{(i,j) \in P_n} D_1(i,j) \tag{14}$$

where P_n denotes the set of pixels on the blob's perimeter. The perimeter is another normalised measure of object size.

- **Histogram of Oriented Gradients:** Edges have been commonly used in previous crowd counting systems. For example, Kong [20] introduced the use of an edge angle histogram on a holistic scale, while Dalal [11] introduced the histogram of oriented gradients (HOG) for person detection. Our system utilises a similar feature, based on the horizontal and vertical derivatives:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \tag{15}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I \tag{16}$$

where $*$ denotes convolution between a Sobel kernel and the image, I, which has been converted to greyscale. The gradient magnitude and orientation at each pixel is therefore:

$$|G(i,j)| = \sqrt{G_x(i,j)^2 + G_y(i,j)^2} \tag{17}$$

$$\angle G(i,j) = \arctan\left(\frac{G_y(i,j)}{G_x(i,j)}\right) \tag{18}$$

For the nth blob segment, a histogram of oriented gradients E_n is constructed by allocating each pixel to one of H histogram channels, based on the pixel's orientation $\angle G(i,j)$. The orientation bins are evenly divided over the range $0^\circ - 180^\circ$, and a total of $H = 6$ bins are used. Each pixel within the blob, $(i,j) \in B_n$, contributes a weighted vote to a histogram bin. The contribution (or vote) is proportional to the gradient magnitude, $|G(i,j)|$; and it is also weighted by the one-dimensional density estimator $D_1(i,j)$ to normalise for perspective. If the

value of the hth histogram bin is denoted $E_n[h]$, and the orientation angle for that bin is lower-bounded by θ_h:

$$E_n[h] = \sum_{(i,j)\in B_n} \begin{cases} D_1(i,j) \times |G(i,j)| & \text{if } \theta_h \leq \angle G(i,j) < \theta_{h+1} \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

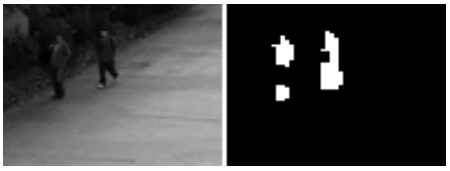
The histogram of oriented gradients is used to help distinguish between humans and other structures in the scene [20]. It also helps to identify occlusions when multiple pedestrians partially block one another from view. Although the blob's area and perimeter are reduced by such occlusions, the image gradients become stronger due to the overlapping body parts, differing skin tones and conflicting clothing.

The full feature vector for the nth blob is therefore:

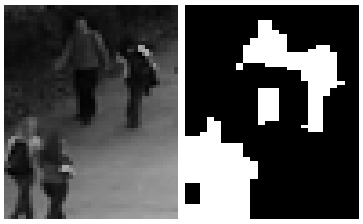
$$\mathbf{x}_n = [A_n, L_n, E_n[1], \ldots, E_n[H]] \tag{20}$$

3.4 System Training

As the proposed algorithm calculates crowd size by determining the number of people in each blob, training is performed on the local level and ground truth annotation must specify a person count for each blob. Due to imperfect foreground segmentation, some blobs are prone to errors such as splitting, fading and noise. This makes annotation difficult and tedious when attempting to allocate fractional counts (as depicted in Figure 6).



(a) Person is fragmented into two blobs (left).



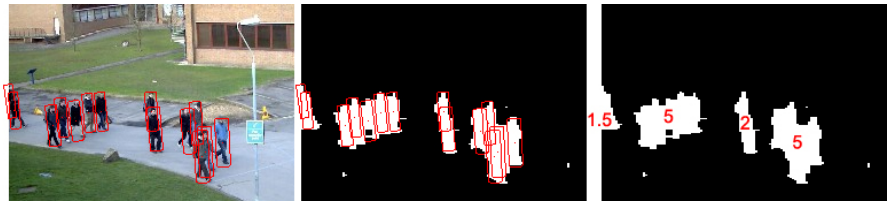
(b) Person is fragmented into two parts (top, centre), one of which is merged with a nearby blob.

Fig. 6 Typical errors in foreground extraction.

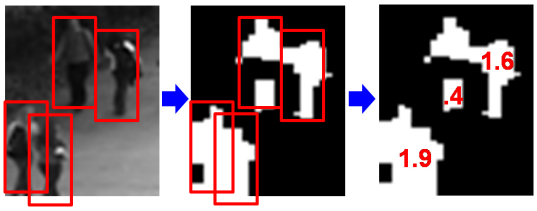
It is desirable for the ground truth to be annotated independently of the processing stage. This is done in a more conventional manner, by simply identifying the image coordinates of each person in the scene. This process is referred to as 'dotting' by Lempitsky [22] because it only requires the user to click once on the centre of each

object in the scene, thereby providing a 'dot' annotation. The surrounding region of a person is then approximated by the outline of a cylinder model (Figures 5, 7).

The *blob* annotations are then performed automatically by the system, by assigning the annotated pedestrians to their corresponding foreground blobs. This is done by considering the overlap between foreground blobs and the pedestrian bounding regions. For example, in the case of large groups, multiple bounding regions will overlap the same blob (Figure 7(a)). On the other hand, when blob fragmentation occurs, multiple blobs will overlap a single bounding region (Figure 7(b)).



(a) Annotations on the PETS 2009 dataset, View 1 [2].



(b) Annotations on the UCSD dataset [7], demonstrating how incorrect foreground segmentation is handled.

Fig. 7 The ground truth annotation process. Manual annotations (left) are overlayed on the foreground segmentation results (centre), and the region overlaps are used to automatically determine ground truth counts for each blob (right). Tiny blobs (noise) are assigned zero.

Using set notation, we define a number of regions as sets of pixels in Table 1. The region of interest mask is denoted M, while the foreground detection result is denoted F, such that their intersection $B = M \cap F$ contains the set of blobs $\{B_n\}$. Each annotated person has a surrounding region R_i approximated by a cylinder model, from which we calculate the following values:

- Q_i: the 'quantity' of person i within the scene's region of interest:

$$Q_i = \frac{|M \cap R_i|}{|R_i|} \tag{21}$$

- C_{in}: the 'contribution' of person i to blob n:

$$C_{in} = \frac{|R_i \cap B_n|}{|R_i \cap B|} \times Q_i \tag{22}$$

Notation	Description
M	Mask of scene (region of interest/ROI).
F	Foreground pixels detected using an adaptive background model [14].
B	Foreground pixels within the ROI mask, i.e. $B = M \cap F$. Consists of blobs $\{B_n\}$.
B_n	Blob n within B, where $B = \bigcup_n B_n$.
R_i	Bounding region of person i. (This may be inside the ROI, partially inside at the edge, or outside.)
$R_i \cap B_n$	The foreground pixels inside R_i belonging to blob B_n, of which there are $\lvert R_i \cap B_n \rvert$.
$R_i \cap B$	The foreground pixels inside R_i, of which there are $\lvert R_i \cap B \rvert = \sum_n \lvert R_i \cap B_n \rvert$.

Table 1 Various regions in an image. Regions are treated as sets of pixels, and set notation is used.

- f_n: the total number of people represented by blob n. This is the sum of 'contributions' from all pedestrians to blob n:

$$f_n = \sum_i C_{in} \tag{23}$$

Thus $\{f_n\}$ are the target counts for the blobs in the scene, computed automatically from the pedestrian coordinates ('dot' annotations). This procedure simplifies the annotation process (as the user merely need to click once on each person in a GUI); and separates the annotation stage from the segmentation stage. A graphical depiction of this ground truth annotation process is displayed in Figure 7.

An advantage of this methodology is that small blobs generated by noise are assigned an annotation of zero, while fragmented blobs are assigned fractional counts in proportion to their size. This allows some tolerance for errors in the foreground detection result.

The holistic ground truth can be measured in two ways. We consider 'hard' and 'soft' values. Hard ground truth, Q_h, is the number of pedestrians whose manual dot annotations lie within the region of interest. This measurement introduces an ambiguity when classifying a pedestrian as either 'in' or 'out' of a region. Soft ground truth, Q_s, assigns fractional values to those pedestrians lying on the perimeter of the region of interest:

$$Q_s = \sum_i Q_i \tag{24}$$

This allows the holistic ground truth to be temporarily fractional as pedestrians enter or exit the scene's boundary. Both measures are considered when presenting results in Section 4.

3.5 Regression

Section 3.3 defines a vector of features to capture the number of people within a group, while Section 3.4 describes a localised ground truth annotation methodology. To train the proposed system, a regression function must be learned using a training data set to count the number of people present in each group.

Existing approaches use linear regression [12, 20, 37], neural networks [28, 20, 36] and Gaussian Process regression [7]. Although the linear model has demonstrated good performance on single datasets, it is not clear that the relationship between the image features and crowd size is indeed linear across all operating conditions and viewpoints.

We adopt Gaussian Process regression (GPR) because it does not place any prior assumptions on the functional relationship between the features and the crowd size. Instead, GPR may be thought of as defining a distribution over functions, where inference takes place in the space of functions [33, 34].

The Gaussian Process is defined as a collection of random variables, any finite subset of which have a joint Gaussian distribution. In regression problems, we observe N samples from a training set, consisting of feature vectors $\mathbf{X} = \{\mathbf{x}_n\}$ and targets $\mathbf{f} = \{f_n\}$. These terms correspond to those in Equations 20 and 23, however, in this case we use n to enumerate all of the blobs observed in the entire training dataset, rather than just one frame.

In GPR these targets are imagined as a sample from some multivariate Gaussian distribution, whose mean is typically taken to be zero:

$$\mathbf{f}|\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}) \tag{25}$$

The covariance matrix, $\mathbf{K} \in \mathbb{R}^{N \times N}$, is obtained from the covariance function $k(\mathbf{x}_n, \mathbf{x}_m)$, such that $\mathbf{K}_{nm} = k(\mathbf{x}_n, \mathbf{x}_m)$. A Gaussian process is fully specified by its covariance function, $k(\mathbf{x}_n, \mathbf{x}_m)$, and the mean function $m(\mathbf{x}) = 0$. The covariance function expresses the covariance of outputs as a function of inputs. For example, a typical covariance function is the squared exponential:

$$k_{\mathrm{SE}}(\mathbf{x}_n, \mathbf{x}_m) = \sigma_{\mathrm{SE}}^2 \exp\left(-\frac{1}{2\ell^2}|\mathbf{x}_n - \mathbf{x}_m|^2\right) \tag{26}$$

The closer the inputs, $\mathbf{x}_n$ and $\mathbf{x}_m$, to one another, the more correlated their outputs will be. The hyperparameter ℓ is a characteristic length scale which represents the distance one would expect to move in the input space to produce a significant change in the output space.

Given N^* test inputs, $\mathbf{X}^* = \{\mathbf{x}_n^*\}$, we wish to obtain the predictive outputs $\mathbf{f}^* = \{f_n^*\}$. In this case, $\mathbf{X}^*$ refers to the feature vectors of the blobs present in an image during testing. Let $\mathbf{K}^*$ denote the $N \times N^*$ train-test covariance matrix with $\mathbf{K}_{nm}^* = k(\mathbf{x}_n, \mathbf{x}_m^*)$. Similarly, let $\mathbf{K}^{**}$ denote the $N^* \times N^*$ test set covariance with $\mathbf{K}_{nm}^{**} = k(\mathbf{x}_n^*, \mathbf{x}_m^*)$.

As all variables in a Gaussian Process are normally distributed, the training and testing data sets are jointly Gaussian [34], with the following distribution:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K}^* \\ \mathbf{K}^{*T} & \mathbf{K}^{**} \end{bmatrix}\right) \tag{27}$$

Each subset of these random variables also follows a joint Gaussian distribution:

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}) \tag{28}$$

$$\mathbf{f}^* \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^{**}) \tag{29}$$

Prediction using Gaussian Process regression is performed by conditioning the predictive outputs on the training data, with the following posterior distribution obtained for $\mathbf{f}^*$:

$$\mathbf{f}^* | \mathbf{f}, \mathbf{X}, \mathbf{X}^{**} \sim \mathcal{N}(\mu, \Sigma) \tag{30}$$

$$\mu = \mathbf{K}^{*T}\mathbf{K}^{-1}\mathbf{f} \tag{31}$$

$$\Sigma = \mathbf{K}^{**} - \mathbf{K}^{*T}\mathbf{K}^{-1}\mathbf{K}^* \tag{32}$$

This method provides not only point estimates, μ, but also a matrix Σ of covariances for the test outputs. The diagonal elements of Σ can thus be used to obtain pointwise error bars.

$$\sigma_n^2 = \Sigma_{nn} \tag{33}$$

For example, setting a 95% confidence interval, the estimate for test sample n would be $\mu_n \pm 1.96\sigma_n$. One advantage of using error bars becomes clear when group tracking is employed in Section 3.6: as a group of people is tracked over time, the confidence of each estimate can be used to weight predictions accordingly.

For each group, the crowd size estimate is a predictive distribution. To obtain a holistic estimate, these distributions must be combined to get the total number of people in the scene. By calculating the sum of N^* Gaussian random variables, an overall prediction and variance is obtained for the scene:

$$\mu_{hol} = \sum_{n=1}^{N^*} \mu_n \tag{34}$$

$$\sigma_{hol}^2 = \sum_{n=1}^{N^*} \sigma_n^2 \tag{35}$$

The covariance function $k(\mathbf{x}_n, \mathbf{x}_m)$ used in our system is designed to capture both short-range and long-range trends in the data. For example, the squared exponential (equation 26) captures the intuitive notion that similar inputs should produce similar outputs. In order to extrapolate the longer trends beyond the training range, the dot product covariance function [34] is also used:

$$k_{\mathrm{DP}}(\mathbf{x}_n, \mathbf{x}_m) = \sigma_{\mathrm{DP}}^2 \left(1 + \mathbf{x}_n^T \mathbf{x}_m\right) \tag{36}$$

Combining these terms results in a regression model that preserves the nonlinearities within the training range while extrapolating outside of the training range in a predominantly linear fashion. Finally, independent Gaussian noise is modeled using the term:

$$k_{\mathrm{GN}}(\mathbf{x}_n, \mathbf{x}_m) = \sigma_{\mathrm{GN}}^2 \delta(n, m) \tag{37}$$

where δ denotes Kronecker's delta function, and contributes only to the diagonals of $\mathbf{K}$ and $\mathbf{K}^{**}$. The final covariance function is therefore:

$$k(\mathbf{x}_n, \mathbf{x}_m) = k_{\mathrm{SE}}(\mathbf{x}_n, \mathbf{x}_m) + k_{\mathrm{DP}}(\mathbf{x}_n, \mathbf{x}_m) + k_{\mathrm{GN}}(\mathbf{x}_n, \mathbf{x}_m) \tag{38}$$

$$= \sigma_{\mathrm{SE}}^2 \exp\left(-\frac{1}{2\ell^2} |\mathbf{x}_n - \mathbf{x}_m|^2\right) + \sigma_{\mathrm{DP}}^2 \left(1 + \mathbf{x}_n^T \mathbf{x}_m\right) + \sigma_{\mathrm{GN}}^2 \delta(n, m) \tag{39}$$

The GPR is 'trained' by choosing the hyperparameters, $\{\sigma_{\mathrm{SE}}, \ell, \sigma_{\mathrm{DP}}, \sigma_{\mathrm{GN}}\}$, so as to maximise the likelihood of the observed training data $p(\mathbf{f}|\mathbf{X})$. (Good predictive performance can still be achieved with reasonable estimates for these hyperparameters.) Following from equation 25,

$$\log p(\mathbf{f}|\mathbf{X}) = -\tfrac{1}{2}\mathbf{f}^T \mathbf{K}^{-1} \mathbf{f} - \tfrac{1}{2} \log |\mathbf{K}| - \tfrac{N}{2} \log 2\pi \tag{40}$$

This term is maximised using an optimisation algorithm such as conjugate-gradients, provided that $k(\mathbf{x}_n, \mathbf{x}_m)$ is differentiable with respect to each of the hyperparameters. Once optimised, prediction is then performed using equations 30-32.

3.6 Tracking Module

Crowd counting algorithms have typically analysed each frame independently of one another, estimating the crowd size based on the features extracted from that frame alone. Although a temporal smoothing may be applied to the holistic count to reduce outliers [12, 36], we propose a local method which employs blob-level tracking to improve each *group's* estimate.

When two or more groups merge to form a larger group, for example, occlusions often occur that obscure the crowd size estimate. By tracking and counting these groups before they merge, their prior estimates can be used to anticipate the size of the newly formed group. Because occlusions are usually temporary (consider two pedestrians passing by one another on a walkway), this prior information can be used to prevent the estimate from being degraded.

Blobs are tracked as they move through a scene by detecting direct correspondences, splits and merges. This is formulated as an optimisation problem by Masoud [29], however in this section we describe a simple set of heuristics based on blob overlap criteria. As we are not concerned with ensuring consistent labeling of objects throughout the sequence, as is required in object tracking, a heuristic based approach that can model the merges and splits of blobs is adequate.

Denoting the mth blob in frame t as $B_{t,m}$, we define the overlap of two blobs in consecutive frames as the number of pixels belonging to both:

$$O_t(m,n) = |B_{t,m} \cap B_{t+1,n}| \tag{41}$$

Using this notation we track groups throughout a sequence by determining direct matches, merges and splits as follows.

1. **Direct Match:** The first step in comparing consecutive frames is to detect direct matches between overlapping blobs. Any blob pair, $B_{t,m}$ and $B_{t+1,n}$, which satisfies the following conditions is deemed a match:

$$O_t(m,n) > 0 \tag{42}$$
$$O_t(i,n) = 0 \quad \forall i \neq m \tag{43}$$
$$O_t(m,j) = 0 \quad \forall j \neq n \tag{44}$$

   These criteria simply require both blobs to overlap one another exclusively.
2. **Merging:** After direct matches have been determined, the matched blobs are removed from consideration. The system then detects P:1 merges and 1:Q splits by combining the remaining blobs as follows. A set of P blobs, $\{B_{t,M_1}, B_{t,M_2}, \ldots, B_{t,M_P}\}$, are deemed to have merged to form the blob, $B_{t+1,n}$, when the following conditions are met:

$$O_t(M_p,n) > 0 \quad \forall p \in [1,P] \tag{45}$$
$$O_t(i,n) = 0 \quad \forall i \notin \{M_0, M_1, \ldots, M_P\} \tag{46}$$
$$O_t(M_p,j) = 0 \quad \forall p \in [1,P], \quad \forall j \neq n \tag{47}$$

3. **Splitting:** Similarly, a split occurs when blob $B_{t,m}$ is divided into the set of blobs: $\{B_{t+1,S_1}, B_{t+1,S_2}, \ldots, B_{t+1,S_Q}\}$. A split is determined when the following conditions are met:

$$O_t(m,S_j) > 0 \quad \forall j \in [1,Q] \tag{48}$$
$$O_t(i,S_j) = 0 \quad \forall j \in [1,Q], \quad \forall i \neq m \tag{49}$$
$$O_t(m,j) = 0 \quad \forall j \notin \{S_0, S_1, \ldots S_Q\} \tag{50}$$

The crowd counting estimates obtained for each blob can then be improved by taking advantage of the detected tracks. The splitting and merging of blobs may be

visualised using a graph structure as shown in Figure 8. As blobs enter and exit the scene, the number of persons that they represent may change while in contact with the perimeter of the scene. Once fully inside the region of interest, however, we assume that directly-matched blobs represent a constant number of people, while merged blobs represent the sum of their constituents' group sizes.

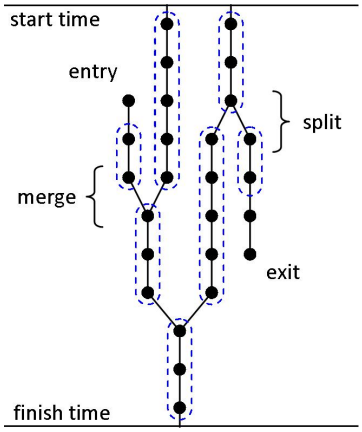


Fig. 8 Visualisation of blob tracking results. Groups of constant size are circled.

The estimate for the mth blob in frame t is denoted $\mu_{t,m}$ with variance $\sigma^2_{t,m}$. Estimates are obtained using Equations 30-33. A group which has been tracked across N frames, from time $t = t_1$ to $t = t_N$, and containing the blobs with indices $\{m_t\}_{t=t_1}^{t_N}$, has an associated set of group size predictions: $\left\{\mu_{t,m_t}, \sigma^2_{t,m_t}\right\}_{t=t_1}^{t_N}$. It is reasonable to expect that the number of people contained in this group is constant, if it is fully contained within the region of interest. We therefore seek to obtain an improved estimate for this group size, $\mu'_{t_N,m_{t_N}}$, by incorporating the tracking history.

Previous experiments [37] used the mean or median value of the group's historical list of estimates, while Kilambi [19] rounded each to an integer and then took the mode. These approaches assume each estimate to be equally valid, and therefore assign an equal weighting to each. However, in practice some frames may be less reliable due to changing environmental conditions or noise, which contribute to uncertainty in the predicted group size.

The variance provides a measurement of the system's uncertainty in the group size prediction, therefore each estimate within a track is weighted by the inverse of its variance. The improved estimate for the most recent blob in a track is thus the weighted average:

$$\mu'_{t_N,m_{t_N}} = \frac{\sum_{t=t_1}^{t_N} \mu_{t,m_t} / \sigma^2_{t,m_t}}{\sum_{i=t_1}^{t_N} 1 / \sigma^2_{t,m_t}} \tag{51}$$

When two or more groups merge to form a new group, each contains a historical list of estimates and variances. A new list is formed by summing their corresponding elements and truncating the new list's length to the shortest of those being merged. The merged group adopts this list and then appends to it any subsequent estimates while it continues to be tracked. Consequently, a tracked person who is temporarily occluded from view by another group may still be represented in the crowd size estimate due to the weight of its prior history.

The tracking procedure described in this section effectively filters the group size estimates over time as the blobs are tracked. As such, it may be expected to produce a modest improvement over the underlying 'raw' estimates obtained using the procedure in Section 3.5. Experimental results in Section 4 indicate that a 4-8% improvement in the mean absolute error are observed.

4 Results

This section presents experimental results of the proposed algorithm. Section 4.1 compares the use of local features to holistic methods using the same training and testing viewpoint, and Section 4.2 presents results for scene invariant crowd counting. Eight datasets were used to evaluate our algorithm, and these are summarised in Table 2. Images from each of these datasets is shown in Figure 1

Data set	**# Frames**	**# Annotated**	**Interval**	**Max crowd**	**Calibration**
PETS 2009, View 1 (13-57, 13-59)	220 + 240	46	10	32	Y
PETS 2009, View 2 (13-57, 13-59)	220 + 240	46	10	32	Y
PETS 2006, View 3 (S1)	3000	120	25	5	Y
PETS 2006, View 4 (S1)	3000	120	25	6	Y
QUT, Camera A	10400	50	200	8	Y
QUT, Camera B	5300	50	100	23	Y
QUT, Camera C	5300	50	100	10	Y
UCSD	2000	2000	1	45	N

Table 2 Eight datasets were used to evaluate our crowd counting algorithm. The total number of frames is listed, and a subset of these frames have been annotated at regular intervals with ground truth. The interval column indicates the spacing between annotated frames.

4.1 Comparison of Local and Holistic Features

In this section we assess the performance of a system using local features compared to holistic methods. The proposed algorithm is trained and tested on the UCSD pedestrian database [7]. This database contains 2000 annotated frames of pedestrian

traffic moving in two directions. The video has been downsampled to 238×158 pixels and 10 fps, grayscale. An example frame is shown in Figure 3.

The performance of the proposed system is assessed using three criteria:

1. **Accuracy:** Although this system is trained on the basis of individual blobs, the testing still takes place on a holistic level. The accuracy of a system is a measure of how closely the estimate follows the ground truth. For testing purposes we consider the mean absolute error and the mean squared error. In order to obtain a direct comparison with competing algorithms, we use 'hard' ground truth (Section 3.4), defined as the number of pedestrians whose manually-annotated (x,y) centroids lie within the region of interest.
2. **Scalability:** Ideally, the training data should cover a wide range of scenarios, similar to those which are expected to be found during operation. In the case of crowd counting, however, we may not have access to video footage of all possible scenarios. Excessive levels of over or under crowding may not be present in the training data because these events are abnormal, and this is the reason we wish to detect them. A system which can extrapolate outside of the ranges found in the training data is of greater practical use.
3. **Practicality:** For a crowd counting system to be practical, it must be relatively easy to deploy. For real world deployment where the algorithm may be required run on several hundred different cameras within a single installation, being able to use a reduced training set is highly desirable. When training crowd counting algorithms, each training frame requires ground truth to be supplied. If several hundred training frames are needed for each camera, then the process of training becomes very tedious and time consuming. To assess practicality, systems are evaluated using reduced training sets.

The following crowd counting techniques are evaluated:

- **Proposed:** The proposed system as described in this chapter is evaluated, in which local features are extracted for each blob and ground truth annotation is performed on a local level. Because camera calibration is not included with the UCSD dataset, we approximate the density map using the relative sizes of reference pedestrians, as described in [7]. A fully calibrated setup is tested in Section 4.2.
- **Kong:** Blobs are sorted into six blob size histograms of bin width 500, as described in [20]. An edge angle histogram is also calculated, for which we use eight histogram bins between 0° and 180°. Regression is performed using a linear model and a neural network. This is a holistic system.
- **Chan:** Segmentation is performed using a mixture of dynamic textures [8], and Gaussian Process regression is used to predict the crowd size moving in each direction (away or towards), from a bank of 29 holistic features [7]. The number of pedestrians moving in each direction is then summed to get the overall crowd count. This system is referred to as 'Chan: away+towards'. Additionally, the segmentation result for each moving class can be taken together to obtain a full foreground mask, which is then used to train the system to obtain the overall

crowd count directly. This system is referred to as 'Chan: all'. Both implementations are examples of a holistic system.

- **Lempitsky:** Each pixel is represented by a feature vector $\mathbf{x_p}$ and a density function $F(\mathbf{p})$ is learned so that integrating the density over any region will yield the number of objects in the region [22]. This is a local approach because dot annotations are used to train the system on a local level.

To assess the accuracy of these systems, the testing protocol of Chan [7] was adhered to. Following this protocol, frames 601-1400 of the data set were set aside for training, while the remaining 1200 frames were used for testing. Because the proposed system is trained using local feature vectors obtained from each group, rather than from each frame, only a subset of the 800 training frames were required to train the system. Lempitsky used two subsets of the training data, described using Matlab notation: 605:5:1400 and 640:80:1360.

In order to compare to Lempitsky, the proposed system is also trained using 10 frames. The proposed system is trained using the same subset used by Lempitsky, 640:80:1360, as well as two other subsets, 610:80:630 and 670:80:1390. We obtained results using these additional training subsets because it provides a more representative picture of how the proposed system performs on this dataset.

These results are all tabulated in Table 3. The mean absolute error of the proposed system is less than 2.0, performing competitively with the local approach proposed by Lempitsky [22]. By each measure of accuracy the proposed approach significantly outperforms the holistic systems, Kong [20] and Chan [7]. Incorporating tracking, as described in Section 3.6, provides a further improvement in performance. Mean absolute error is reduced by 4-8% by the inclusion of the tracking module, and mean square error is improved by 6-14%. The results of the proposed system are plotted in Figure 9.

System	**Training subset**	**Mean abs. Error**	**Mean Square Error**
Kong, linear	all	1.92	5.60
Kong, neural network (5 runs)	all	2.47 ± 0.41	9.53 ± 3.01
Chan, away+towards	all	1.95	5.75
Chan, all	all	1.95	6.06
Lempitsky	605:5:1400	1.70	-
Lempitsky	640:80:1360	2.02	-
Proposed, no tracking	610:80:1330	1.79	4.95
Proposed, with tracking		1.72	4.50
Proposed, no tracking	640:80:1360	1.33	2.91
Proposed, with tracking		1.28	2.74
Proposed, no tracking	670:80:1390	1.57	3.94
Proposed, with tracking		1.45	3.39

Table 3 Testing results on the UCSD data set. Frames 601-1400 were set aside for training, and frames 1-600 and 1401-2000 were used for testing. Mean and standard deviation are reported for the neural network based on five runs.

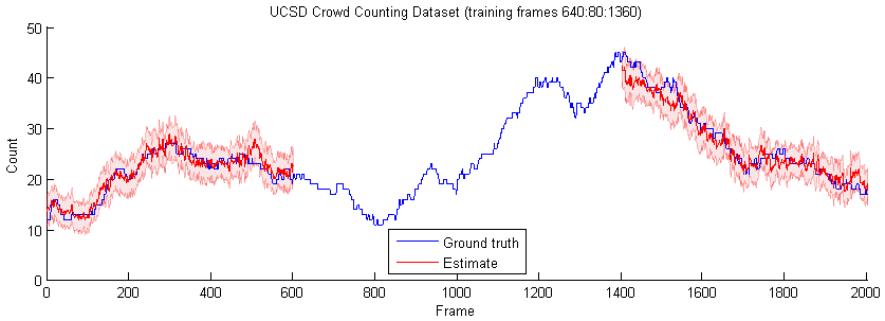


Fig. 9 Testing results on the UCSD data set. Frames 640:80:1360 were set aside for training, and frames 1-600 and 1401-2000 were used for testing. A 95% confidence interval is drawn either side of the estimate (equations 34-35).

The ability of the system to extrapolate outside the range of crowd sizes seen in the training data set (scalability) is examined by allocating five different subsets for training: {5:5:400}, {405:5:800}, {805:5:1200}, {1205:5:1600} and {1605:5:2000}. Each training set contains 80 frames. Following the experimental protocol of Lempitsky [22], the remaining frames of the dataset were reserved for testing in each case. The results of this experiment are tabulated in Table 4. The proposed system outperforms the holistic method of Kong and performs competitively with Lempitsky. The results are plotted in Figure 10, indicating the ability of the system to scale to larger or smaller crowd sizes than those encountered in the training range.

System	**Mean abs. Error**
Kong, linear	2.33 ± 0.64
Lempitsky	1.78 ± 0.39
Proposed, no tracking	1.95 ± 0.62
Proposed, with tracking	1.89 ± 0.64

Table 4 Scalability testing results on the UCSD data set. Five different training ranges were used, while the remaining frames were withheld for testing. Mean and standard deviation are reported for these five tests.

Finally, the practicality of the proposed system was evaluated by repeating the scalability experiments on more sparse training sets: {20:40:380}, {420:40:780}, {820:40:1180}, {1220:40:1580} and {1620:40:1980}. Each training set contained only 10 frames. These training frames contain insufficient data to populate all of the histograms in Kong's system, prohibiting it from being properly trained. As such the results for Kong are omitted from this experiment. The results for the proposed system are presented against Lempitsky's approach in Table 5. The proposed system achieves a mean absolute error of 2.18 without tracking and 2.14 when tracking is incorporated; Lempitsky's approach performs slightly better with a mean absolute error of 2.06.

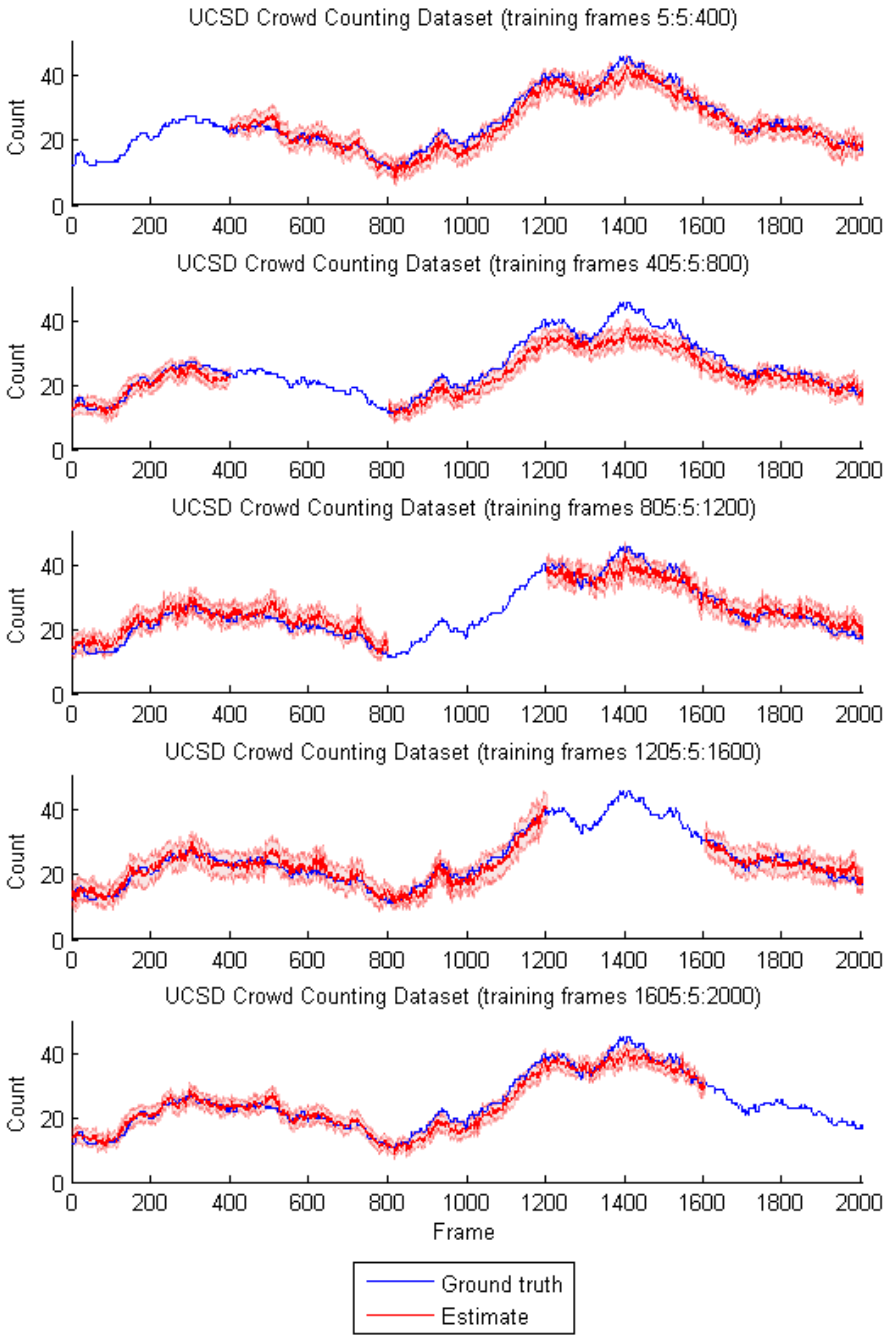


Fig. 10 Scalability testing results on the UCSD data set. Five different training ranges were used, while the remaining frames were withheld for testing.

System	Mean abs. Error
Kong	Insufficient training data
Lempitsky	2.06 ± 0.59
Proposed, no tracking	2.18 ± 0.76
Proposed, with tracking	2.14 ± 0.79

Table 5 Practicality testing results on the UCSD data set. Five sparse training ranges were used containinig only 10 frames, while the remaining frames were withheld for testing. Mean and standard deviation are reported for these five tests.

These results indicate that both local approaches are capable of accurate crowd counting, on smaller and larger crowd sizes than those encountered during training, even when a reduced training set is used (10 frames). By contrast, holistic systems appear to require a greater training set (ideally more than one hundred frames). This is most likely because local approaches have access to a wealth of data in every training frame, rather than treating each frame as a single instance.

The training requirements of a local system (Section 3.4) are slightly more involved, as they require a dot to be placed on the centre of each person, whereas a holistic system only requires the overall count. However, as a holistic count requires each individual to be located anyway, it could be argued that annotating each person with a dot involves little additional effort, resulting in a substantially greater performance benefit.

4.2 Scene Invariance

In this section we assess the performance of the proposed system when trained and tested on different viewpoints. The purpose of this experiment is to emulate the scenario in which a 'plug-and-play' or turn-key system has been pre-trained on a variety of different viewpoints before being deployed on a new scene. Section 4.2.1 introduces a new crowd counting database that has been prepared for this chapter, which will be made available to the computer vision community. Section 4.2.2 presents the scene invariant crowd counting results of the proposed system.

4.2.1 QUT Dataset

Seven data sets with camera calibration are listed in Table 2, containing a total of 27,920 frames and more than sixteen minutes of annotated crowd footage. To supplement the existing public data sets, such as PETS 2006 [1] and PETS 2009 [2], a new database has been developed containing footage obtained from our university campus. This database is referred to as 'QUT' and will be made available to the computer vision community for experimentation.[1]

[1] Please contact David Ryan (david.ryan@qut.edu.au) for details on obtaining this database.

This database contains three challenging viewpoints, which are referred to as Camera A (Figure 1(e)), Camera B (Figure 1(f)) and Camera C (Figure 1(g)). The sequences contain reflections, shadows and difficult lighting fluctuations, which makes crowd counting difficult. Furthermore, Camera C is positioned at a particularly low camera angle, leading to stronger occlusion than is present in other datasets.

Previous crowd counting datasets have been substantially shorter in length than those included in the QUT database. For example, PETS 2009 contains two crowd counting sequences of length 220 and 240 frames, while the UCSD dataset contains 2000 consecutively annotated frames. Although these resources are extremely valuable for testing crowd counting algorithms, they do not adequately capture the long-term performance of a system over varying conditions. For example, if a system performs poorly on one particular frame, it is likely that the preceding and subsequent frames will suffer from the same vulnerability. On shorter sequences such as the PETS 2009 datasets, this may lead to biased results that do not adequately describe a system's true performance capabilities.

In order to combat this potential problem, the QUT datasets are annotated at more sparse intervals: every 100 frames for cameras B and C, and every 200 frames for camera A as this is a longer sequence. Testing is then performed by comparing the crowd size estimate to the ground truth at these sparse intervals, rather than at every frame. This closely resembles the intended real-world application of this technology, where an operator may periodically 'query' the system for a crowd count. Although the human operator does not require this information from *every* frame, the system should at least provide accurate results whenever it is requested.

Due to the difficulty of the environmental conditions in these scenes, the first 400-500 frames of each sequence is set aside for learning the background model. This is a requirement for proper operation of many multi-modal algorithms such as Denman [14, 15], Stauffer-Grimson [39] and Zivkovic [46], which are used very widely in the computer vision field. Existing databases generally do not provide time to learn the background, and although PETS 2009 provides some detached background sequences, they do not immediately precede the crowd counting sequences to be tested, limiting their usefulness.

4.2.2 Results

In this section scene invariance is tested using the seven calibrated datasets from Table 2. In each experiment one viewpoint was witheld for testing, and the remaining six viewpoints were used for training. Ten frames from each training viewpoint were selected, so that a total of sixty training frames were used to train the system in each experiment. Testing was then performed on the remaining viewpoint, using all of the annotated ground truth frames to calculate the mean absolute error and the mean square error.

Because the number of people in each scene was often fractional, we use the 'soft' ground truth defined in equation 24. This makes sense when evaluating our

algorithm because it makes use of local features and has been annotated with occasionally fractional counts (Section 3.4). A blob's ground truth does not jump directly from 0 to 1 (or vice versa) when entering or exiting a scene, for example.

Results for these experiments are tabulated in Table 6. Across all experiments, weighted equally, the mean absolute error was 1.21 ± 0.58, and in most cases a modest improvement was observed by incorporating the tracking procedure of Section 3.6. The crowd counting results for each sequence are plotted in Figure 11.

	No tracking		**With tracking**	
Test Set	Mean abs. error	Mean square error	Mean abs. error	Mean square error
PETS 2009, View 1	1.70	4.12	1.65	3.91
PETS 2009, View 2	1.24	3.25	1.23	3.31
PETS 2006, View 3	0.34	0.39	0.34	0.39
PETS 2006, View 4	0.79	1.15	0.79	1.15
QUT, Camera A	0.92	1.62	0.92	1.56
QUT, Camera B	2.09	9.49	2.06	9.37
QUT, Camera C	1.36	3.20	1.22	2.42
All tests	1.21 ± 0.58	3.32 ± 3.02	1.17 ± 0.57	3.16 ± 3.00

Table 6 Scene invariant testing results on the seven calibrated data sets of Table 2. When testing each viewpoint, the system is trained on the six other viewpoints.

Screenshots from the system during its operation are shown in Figure 12. Blob perimeters are drawn in red and the group size estimates are written on the centroid of each blob, rounded to the nearest integer. In most cases the group estimate is correct within 1 of the ground truth. An advantage of the local features based approach is that the system can provide a crowding estimate not just for the holistic level, but for the regions occupied by each group within the image. This could be used by a system to detect abnormal crowd distribution patterns or local overcrowding situations, even when the holistic crowd size is within normal ranges.

Figure 12 also includes some false positives in the foreground segmentation (PETS 2009, View 1) and a missed detection (QUT, Camera B). The background subtraction on QUT Camera B is particularly challenging due to the darkness of the scene and the background. This is the main source of error in our experiments, and it accounts for the under-estimation observed for QUT Camera B, which is seen in Figure 11. Conversely, lighting fluctuations in PETS 2009 View 1 resulted in some false positives, accounting for the slight over-estimation also observed in Figure 11.

Background modeling and foreground segmentation continue to remain amongst the major challenges in visual surveillance and the state of the art is continually evolving. Nevertheless the proposed system demonstrates accurate performance on these datasets and in most cases handles noise and blob fragmentation quite well: small instances of noise are disregarded, because they were learned during training with annotations of zero, while fragmented blobs are assigned fractional counts where necessary.

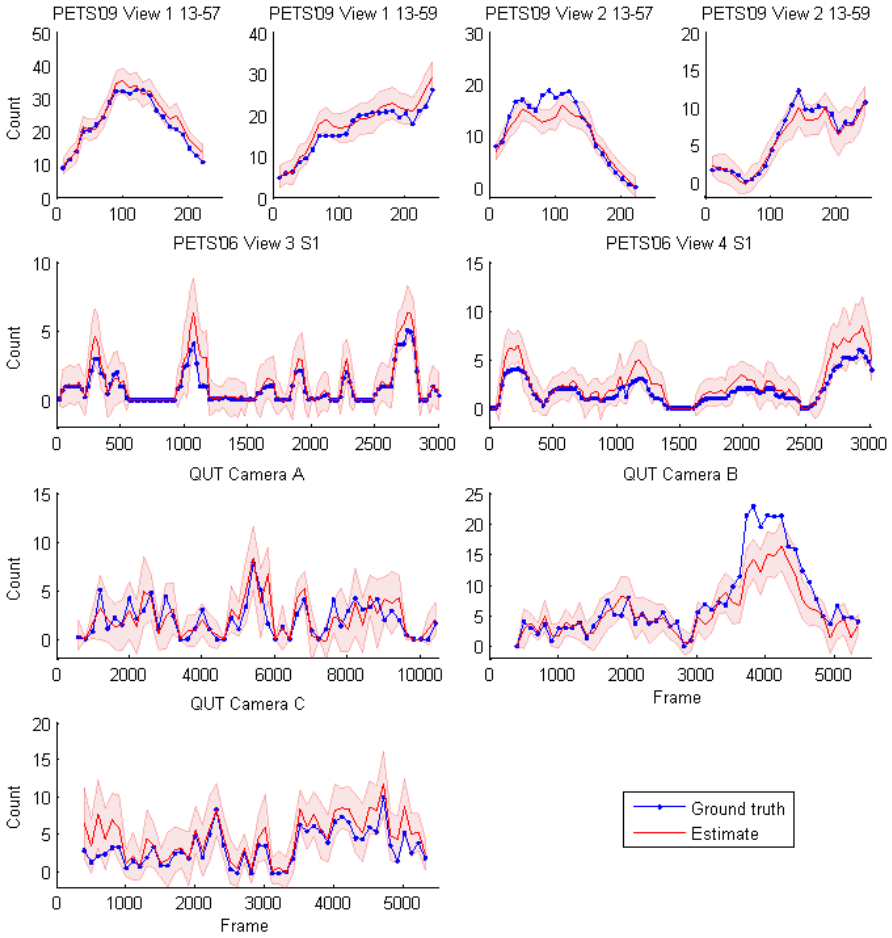
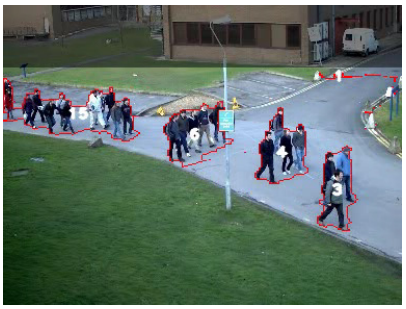


Fig. 11 Scene invariant testing results on the seven calibrated data sets of Table 2. When testing each viewpoint, the system is trained on the six other viewpoints.

These results support the ability of the proposed system to perform scene invariant crowd counting when trained and tested on different viewpoints, and provides a baseline methodology and database for future scene-invariant experiments.

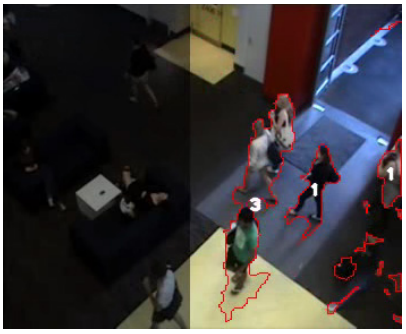
5 Conclusion

This chapter proposed a novel scene-invariant crowd counting algorithm based on local features, specific to groups and individuals in an image, to estimate the crowd size and its distribution across a scene. Unlike previous systems that have typically

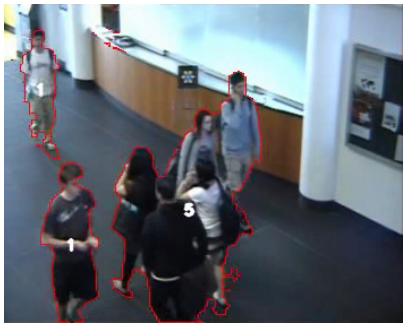


(a) PETS 2009, View 1. Groups of size 15, 6, 4 and 3 are detected.

(b) PETS 2006, View 3. Groups of size 1 and 4 are detected.



(c) QUT, Camera B. Groups of size 3, 1 and 1 are detected.



(d) QUT, Camera C. Groups of size 1, 1 and 5 are detected.

(e) UCSD dataset, smaller groups

(f) UCSD dataset, larger groups

Fig. 12 Screenshots of the proposed algorithm operating on five viewpoints. The region of interest is highlighted.

employed holistic features, the proposed approach is annotated, trained and tested at a local level. Camera calibration is incorporated into the system to scale features between viewpoints, and a tracking algorithm was described to further improve the system's performance.

The proposed approach outperforms the baseline holistic methods of Kong [20] and Chan [7], and performs competitively with the local approach of Lempitsky [22], when trained and tested on the same viewpoint. The proposed system was demonstrated to be highly accurate, scalable and practical, with very minimal training requirements. Accurate test results were obtained from as few as ten training frames of data.

Scene invariance was also demonstrated by training the system on multiple cameras and then testing it on a new, unseen viewpoint. Accurate crowd counting results were obtained for seven calibrated sequences, including a new QUT dataset designed to help evaluate the performance of crowd counting systems in difficult real-world conditions.

The proposed system does not require any additional training when deployed for crowd counting on a new camera. This brings the computer vision field one step closer toward a truly 'plug-and-play' system which is pre-trained on a large bank of data from a variety of cameras. This technology has many potential applications, including automatic gathering of business intelligence, crowd safety monitoring and abnormality detection.

Future research into scene invariant crowd counting will continue to investigate background modeling techniques, scene invariant feature extraction, autocalibration methods, and improved tracking algorithms that can be readily inorporated into framework already proposed in this chapter.

References

1. Ninth ieee international workshop on performance evaluation of tracking and surveillance (2006). URL http://www.cvg.rdg.ac.uk/PETS2006/
2. Eleventh ieee international workshop on performance evaluation of tracking and surveillance (2009). URL http://www.cvg.rdg.ac.uk/PETS2009/
3. Abdel-Aziz, Y., Karara, H.: Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. In: Proceedings of the Symposium on Close-Range Photogrammetry, pp. 1–18 (1971)
4. Bose, B., Grimson, E.: Ground plane rectification by tracking moving objects. In: IEEE International Workshop on Visual Surveillance and PETS (2004)
5. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
6. Celik, H., Hanjalic, A., Hendriks, E.: Towards a robust solution to people counting. Image Processing, 2006 IEEE International Conference on pp. 2401–2404 (2006)
7. Chan, A., Liang, Z.S., Vasconcelos, N.: Privacy preserving crowd monitoring: Counting people without people models or tracking. Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on pp. 1–7 (2008)
8. Chan, A., Vasconcelos, N.: Modeling, clustering, and segmenting video with mixtures of dynamic textures. Pattern Analysis and Machine Intelligence, IEEE Transactions on **30**(5), 909–926 (2008)

9. Chan, A.B., Morrow, M., Vasconcelos, N.: Analysis of crowded scenes using holistic properties. In: Performance Evaluation of Tracking and Surveillance workshop at CVPR 2009, pp. 101–108. Miami, Florida (2009)
10. Cho, S.Y., Chow, T., Leung, C.T.: A neural-based crowd estimation by hybrid global learning algorithm. Systems, Man, and Cybernetics, Part B, IEEE Transactions on **29**(4), 535–541 (1999)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, vol. 1, pp. 886 –893 vol. 1 (2005)
12. Davies, A., Yin, J.H., Velastin, S.: Crowd monitoring using image processing. Electronics & Communication Engineering Journal **7**(1), 37–47 (1995)
13. Denman, S.: Improved detection and tracking of objects in surveillance video. Ph.D. thesis, Queensland University of Technology (2009). URL `http://eprints.qut.edu.au/29328/`
14. Denman, S., Chandran, V., Sridharan, S.: An adaptive optical flow technique for person tracking systems. Pattern Recognition Letters **28**(10), 1232 – 1239 (2007)
15. Denman, S., Fookes, C., Sridharan, S.: Improved simultaneous computation of motion detection and optical flow for object tracking. In: Digital Image Computing: Techniques and Applications, 2009. DICTA '09., pp. 175 –182 (2009)
16. Haralick, R.: Statistical and structural approaches to texture. Proceedings of the IEEE **67**(5), 786–804 (1979)
17. Hou, Y.L., Pang, G.: People counting and human detection in a challenging situation. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on **41**(1), 24 –33 (2011)
18. Kilambi, P., Masoud, O., Papanikolopoulos, N.: Crowd analysis at mass transit sites. In: Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE, pp. 753 –758 (2006)
19. Kilambi, P., Ribnick, E., Joshi, A.J., Masoud, O., Papanikolopoulos, N.: Estimating pedestrian counts in groups. Computer Vision and Image Understanding **110**(1), 43 – 59 (2008)
20. Kong, D., Gray, D., Tao, H.: A viewpoint invariant approach for crowd counting. Pattern Recognition, 2006. ICPR 2006. 18th International Conference on **3**, 1187–1190 (2006)
21. Krahnstoever, N., Mendonca, P.R.S.: Bayesian autocalibration for surveillance. In: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV), pp. 1858–1865. IEEE Computer Society, Washington, DC, USA (2005)
22. Lempitsky, V., Zisserman, A.: Learning to count objects in images. In: Advances in Neural Information Processing Systems (NIPS) (2010)
23. Lin, S.F., Chen, J.Y., Chao, H.X.: Estimation of number of people in crowded scenes using perspective transformation. Systems, Man and Cybernetics, Part A, IEEE Transactions on **31**(6), 645–654 (2001)
24. Lv, F., Zhao, T., Nevatia, R.: Self-calibration of a camera from video of a walking human. In: ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 1, p. 10562. IEEE Computer Society, Washington, DC, USA (2002)
25. Ma, R., Li, L., Huang, W., Tian, Q.: On pixel count based crowd density estimation for visual surveillance. Cybernetics and Intelligent Systems, 2004 IEEE Conference on **1**, 170–173 vol.1 (2004)
26. Marana, A., Cavenaghi, M., Ulson, R., Drumond, F.: Real-Time Crowd Density Estimation Using Images. In: Advances in Visual Computing, *Lecture Notes in Computer Science*, vol. 3804, pp. 355–362. Springer Berlin / Heidelberg (2005)
27. Marana, A., Da Fontoura Costa, L., Lotufo, R., Velastin, S.: Estimating crowd density with minkowski fractal dimension. Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings., 1999 IEEE International Conference on **6**, 3521–3524 vol.6 (1999)
28. Marana, A., Velastin, S., Costa, L., Lotufo, R.: Estimation of crowd density using image processing. Image Processing for Security Applications (Digest No.: 1997/074), IEE Colloquium on pp. 11/1–11/8 (1997)

29. Masoud, O., Papanikolopoulos, N.: A novel method for tracking and counting pedestrians in real-time using a single camera. Vehicular Technology, IEEE Transactions on **50**(5), 1267–1278 (2001)
30. Paragios, N., Ramesh, V.: A mrf-based approach for real-time subway monitoring. In: 2001 Conference on Computer Vision and Pattern Recognition (CVPR 2001), pp. 1034–1040 (2001)
31. Rabaud, V., Belongie, S.: Counting crowded moving objects. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, vol. 1, pp. 705 – 711 (2006)
32. Rahmalan, H., Nixon, M., Carter, J.: On crowd density estimation for surveillance. Crime and Security, 2006. The Institution of Engineering and Technology Conference on pp. 540–545 (2006)
33. Rasmussen, C.: Gaussian processes in machine learning. In: O. Bousquet, U. von Luxburg, G. Rtsch (eds.) Advanced Lectures on Machine Learning, *Lecture Notes in Computer Science*, vol. 3176, pp. 63–71. Springer Berlin / Heidelberg (2004)
34. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. MIT Press (2006)
35. Regazzoni, C., Tesei, A., Murino, V.: A real-time vision system for crowding monitoring. Industrial Electronics, Control, and Instrumentation, 1993. Proceedings of the IECON '93., International Conference on pp. 1860–1864 vol.3 (1993)
36. Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using multiple local features. In: Digital Image Computing: Techniques and Applications, 2009. DICTA '09., pp. 81 –88 (2009)
37. Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using group tracking and local features. In: Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on, pp. 218 –224 (2010)
38. Shi, J., Tomasi, C.: Good features to track. In: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on, pp. 593 –600 (1994)
39. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on., vol. 2, pp. 2 vol. (xxiii+637+663) (1999)
40. Tsai, R.: An efficient and accurate camera calibration technique for 3d machine vision. Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 86) pp. 364–374 (1986)
41. Tuzel, O., Porikli, F., Meer, P.: Human detection via classification on riemannian manifolds. In: Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on, pp. 1 –8 (2007)
42. Wu, X., Liang, G., Lee, K.K., Xu, Y.: Crowd density estimation using texture analysis and learning. Robotics and Biomimetics, 2006. ROBIO '06. IEEE International Conference on pp. 214–219 (2006)
43. Xiaohua, L., Lansun, S., Huanqin, L.: Estimation of Crowd Density Based on Wavelet and Support Vector Machine. Transactions of the Institute of Measurement and Control **28**(3), 299–308 (2006)
44. Yang, T., Zhang, Y., Shao, D., Li, Y.: Clustering method for counting passengers getting in a bus with single camera. Optical Engingeering **49**(3) (2010)
45. Zhang, Z.: A flexible new technique for camera calibration. Pattern Analysis and Machine Intelligence, IEEE Transactions on **22**(11), 1330 – 1334 (2000)
46. Zivkovic, Z.: Improved adaptive gaussian mixture model for background subtraction. In: ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2, pp. 28–31. IEEE Computer Society, Washington, DC, USA (2004)