Pattern Recognition Letters 44 (2014) 98–112

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Scene invariant multi camera crowd counting

David Ryan [*], Simon Denman, Clinton Fookes, Sridha Sridharan

Image and Video Laboratory, S1101, Queensland University of Technology, 2 George St, Brisbane 4000, Australia

A R T I C L E I N F O

Article history:
Available online 12 October 2013

Communicated by Simone Calderara

Keywords:
Crowd counting
Multi camera
Scene invariant

A B S T R A C T

Automated crowd counting has become an active field of computer vision research in recent years. Existing approaches are scene-specific, as they are designed to operate in the single camera viewpoint that was used to train the system. Real world camera networks often span multiple viewpoints within a facility, including many regions of overlap.

This paper proposes a novel scene invariant crowd counting algorithm that is designed to operate across multiple cameras. The approach uses camera calibration to normalise features between viewpoints and to compensate for regions of overlap. This compensation is performed by constructing an 'overlap map' which provides a measure of how much an object at one location is visible within other viewpoints. An investigation into the suitability of various feature types and regression models for scene invariant crowd counting is also conducted. The features investigated include object size, shape, edges and keypoints. The regression models evaluated include neural networks, *K*-nearest neighbours, linear and Gaussian process regression.

Our experiments demonstrate that accurate crowd counting was achieved across seven benchmark datasets, with optimal performance observed when all features were used and when Gaussian process regression was used. The combination of scene invariance and multi camera crowd counting is evaluated by training the system on footage obtained from the QUT camera network and testing it on three cameras from the PETS 2009 database. Highly accurate crowd counting was observed with a mean relative error of less than 10%.

Our approach enables a pre-trained system to be deployed on a new environment without any additional training, bringing the field one step closer toward a 'plug and play' system.

1. Introduction

Automated crowd counting has become an active field of computer vision research in recent years. Crowd size is the most common indicator of security threats such as rioting, violent protest and mass panic, and it can also indicate congestion and other abnormal events within peaceful crowds. Crowd information can also be used to provide operational analytics for business intelligence.

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. In a facility containing numerous cameras, this requires each viewpoint to be trained independently, which can be an arduous and time consuming task. It is not practical to supply hundreds of frames of ground truth for every viewpoint. In this paper, a novel algorithm is proposed which 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, including those captured at different locations.

In practice, this means that a system can be trained on a bank of 'reference viewpoints', and then deployed on any number of *new* viewpoints without any additional ground truth annotations being required, greatly reducing the time and cost of configuring a crowd counting system. To facilitate the development of this technique, an investigation into various features and regression models for scene invariant crowd counting is conducted to determine the best combination in practice.

Another limitation of existing methods is that they are designed to count crowds within a *single camera viewpoint*, whereas real-world camera networks span multiple viewpoints within a facility, including some regions of overlap. Since some individuals will be detected across multiple cameras, it is necessary to compensate for this overlap to avoid over-estimation of the total number of people. This paper extends crowd counting across multiple cameras by utilising camera calibration parameters. An 'overlap map' is calculated which provides a measure of how much an object at one location is visible within other viewpoints, and this is used to modify crowd density on a pixelwise basis.

* Corresponding author. Tel.: +61 7 3138 9329.
E-mail addresses: david.ryan@qut.edu.au, d23.ryan@student.qut.edu.au (D. Ryan), s.denman@qut.edu.au (S. Denman), c.fookes@qut.edu.au (C. Fookes), s.sridharan@qut.edu.au (S. Sridharan).

0167-8655/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.patrec.2013.10.002

The proposed algorithm is tested on seven datasets which utilise camera calibration: PETS 2009, Views 1 and 2 (PETS, 2009); PETS 2006, Views 3 and 4 (PETS, 2006); and QUT, Cameras 3, 5 and 8 (Section 5.2). These datasets feature crowds of size 1 to 43 people in various lighting conditions and differing camera angles. The system is demonstrated to be scene invariant and capable of supporting multiple cameras, with accurate crowd counting results.

The details of the scene invariant crowd counting algorithm have been previously published in Ryan et al. (2012). This paper makes a number of additional contributions:

1. A comprehensive investigation into optimal feature sets and regression models for scene invariant crowd counting.
2. An extension to multi camera environments, allowing the total number of people across a scene to be estimated.
3. A combination of scene invariance and multi camera crowd counting algorithms, which is evaluated on a three-camera setup.

The remainder of this paper is structured as follows: Section 2 reviews the existing crowd counting literature; Section 3 describes the proposed scene invariant crowd counting algorithm; Section 4 extends this algorithm to operate across multiple cameras; Section 5 presents the evaluation protocol and benchmark datasets; Section 6 presents the experimental results of our algorithm; and Section 7 presents conclusions and directions for future work.

2. Background

Current approaches to crowd counting generally employ supervised machine learning techniques to map between the image feature space and the crowd size estimate. Regression is performed at either the holistic level of an image (Davies et al., 1995; Kong et al., 2006; Chan et al., 2009) or at a local scale (Kilambi et al., 2008; Ryan et al., 2009; Lempitsky and Zisserman, 2010).

Holistic image features include textural statistics (Marana et al., 1997), Minkowski fractal dimension (Marana et al., 1999) and translation invariant orthonormal Chebyshev moments (Rahmalan et al., 2006). Holistic textural features such as these are sensitive to external changes, and for outdoor environments the natural fluctuations in lighting between morning and afternoon have been shown to reduce system performance (Rahmalan et al., 2006). A number of algorithms use background modelling techniques (Stauffer and Grimson, 1999; Denman, 2009) in order to identify pedestrians in the foreground. Davies et al. (1995) modelled the relationship between foreground pixels and crowd size using linear regression, and subsequent approaches have attempted to deal with perspective and occlusion. Paragios and Ramesh (2001) introduced the use of a density estimator to account for perspective and Ma et al. (2004) computed a density map which weighted each pixel by the area it represented on the ground plane. The sum of *weighted* foreground pixels is used as a measure of crowding.

Kong et al. (2006) proposed the use of histogram based features to capture the various levels of occlusion present in a scene. Foreground 'blob' segments were aggregated into size-based histograms, and an edge orientation histogram was constructed based on the gradient directions. The edge orientation histogram is used to help distinguish between humans and other structures in the scene (Kong et al., 2006). Similar features have been used in other visual surveillance research, such as the histogram of oriented gradients employed by Dalal and Triggs (2005) for the explicit purpose of human detection.

A unique segmentation technique was used by Chan et al. (2008) to identify foreground motion in two directions, based on the mixture of dynamic textures. A large number of holistic image features were extracted including foreground area, perimeter pixel count, edge orientation histogram and textural features. In total, 29 features were extracted and Gaussian process regression was used to predict the number of pedestrians walking in each direction.

Local approaches to crowd counting utilise detectors or features which are specific to individuals or groups of people within an image. Lin et al. (2001) has proposed the use of head detection for crowd counting. The Haar wavelet transform was used in conjunction with the support vector machine to classify head-like contours as human.

Celik et al. (2006) proposed a person-counting algorithm which did 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 et al. (2008) models a group of pedestrians as an elliptical cylinder, assuming a constant spacing between people within the group. Lempitsky and Zisserman (2010) proposed an object counting algorithm which sought to estimate a density function of the pixels 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 is represented by a feature vector containing foreground and gradient information, and a linear model is used to obtain the density at each pixel.

These approaches rely on scene-specific training data which requires a system to be trained and tested on the same viewpoint, using potentially hundreds (Kong et al., 2006) or thousands (Chan et al., 2009) of annotated training frames. 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.

3. Scene invariant crowd counting

This section describes a scene invariant crowd counting algorithm which can be trained and tested on different cameras. The system is trained on a bank of 'reference viewpoints' before being deployed on any number of unseen viewpoints, without any additional training requirements.

The approach is based on camera calibration, which is used to normalise features across viewpoints. This algorithm was originally published in Ryan et al. (2012, 2011) and the details of the approach are discussed in this section. Furthermore, this paper extends the approach by utilising additional image features, and Section 6.1 investigates various combinations of these features to determine the best approach in practice. The algorithm is extended to multi camera environments in Section 4.

This section is structured as follows: Section 3.1 describes how scene invariance is achieved through the use of a 'density map' to normalise features; Section 3.2 describes the feature extraction process; and Section 3.3 outlines the procedure used to train the system.

3.1. Scene invariance

Scene invariance is achieved by scaling the features extracted from each pixel to normalise for camera position and orientation. A density map, S, is constructed based on camera calibration parameters. The density map supplies a weight, $S(i,j)$, to each pixel, which is used to scale the features extracted from that pixel. This approach has been used previously to compensate for the effects of perspective within a single image (Section 2), however in this paper we use the density map to achieve normalisation across different scenes.

The density map is calculated from camera calibration. A number of camera calibration methods have been described in the literature (Abdel-Aziz and Karara, 1971; Tsai, 1986; Zhang, 2000), although the most popular of these is Tsai's model (Tsai, 1986, 1987), which is frequently used on visual surveillance databases such as PETS 2006 and PETS 2009 (PETS, 2006, 2009). 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 (i,j). A number of automated procedures also exist for estimating camera calibration based on human or object tracking (Bose and Grimson, 2004; Lv et al., 2002; Krahnstoever and Mendonca, 2005). These approaches could readily be incorporated into the proposed framework. However, as Tsai calibration parameters are already available for public visual surveillance datasets, and the method is widely used and well understood, Tsai's model has been used in this research.

Quasi-calibration methods have also been used previously to achieve perspective normalisation within a single image (Paragios and Ramesh, 2001; Ma et al., 2004; Chan et al., 2008). Typically, a reference pixel near the bottom of the image is assigned the weight 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. Quasi-calibration methods use *relative* object sizes, such as pedestrians (Chan et al., 2008) or roadway width (Ma et al., 2004), in terms of pixels, without taking into account real world measurements. Although these approaches are suitable for compensating for perspective within a single image, the scene invariance proposed in this paper requires a real world basis. For this reason Tsai's model is selected.

In the proposed system, a 3D cylinder model of a fixed size is used to approximate the size of a human. The cylinder has a radius of $r = 0.25$ m and a height of $h = 1.7$ m. As depicted in Fig. 1, this cylinder may be projected into a scene centred around any pixel (i,j).

The cylinder is projected into the scene at every location using camera calibration as follows. Firstly each image coordinate (i,j) is taken to correspond to the centre of a hypothetical person (or a cylinder model representing them). This is converted to the real world position (p_x, p_y, p_z) with $p_z = \frac{h}{2}$ at the centre of the model. The top and bottom circles are positioned at a height of $z = h$ and $z = 0$, respectively. Each circle is approximated using a 20-sided polygon, whose vertices are projected back into the image plane and joined together using a number of straight lines, imitating curvature. The sides of the cylinder model are also drawn using straight lines in the image plane. The notation $R_{i,j}$ is used to represent the cylinder model centred around pixel (i,j), and $|R_{i,j}|$ represents the *area* of this template in the image plane.

The density map is constructed using the inverse of the cylinder model's projected area, at every location within the image:

$$S(i,j) = \frac{1}{|R_{i,j}|} \tag{1}$$

Therefore the value of the density map at any location is based on the size of a *fixed* real world object, centred at that location. This density map provides a weight to each pixel so that an object occupying a set of pixels, B, has a weighted area of $\sum_{(i,j)\in B} S(i,j)$. Consequently distant objects occupying fewer pixels are compensated by their larger weights in the density map.

The use of camera calibration in constructing the density map, rather than an arbitrary 'reference' pixel or shape, is advantageous because it is defined in terms of a fixed real-world object; 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 features across viewpoints.

The extraction of these features and the usage of the density map S are described in the subsequent section.

3.2. Feature extraction

Background modelling is a fundamental step in many surveillance systems, and it forms the backbone of the proposed algorithm. The background model allows subsequent tasks to be performed, such as foreground detection, group segmentation and feature extraction. We use the approach described in Denman (2009) and Denman et al. (2007, 2009) to generate the foreground binary mask.

In the proposed crowd counting algorithm, a crowd estimate is obtained for each foreground 'blob' segment 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 by explicitly labelling the number of people represented by each blob in an image, therefore each frame provides several instances of ground truth. The details of this annotation process is discussed in Ryan et al. (2012).

Feature extraction is performed as a subsequent step to group segmentation, and the density map (Section 3.1) is used to weight the features extracted from each pixel in order to achieve normalisation across scenes (and within an image to compensate for perspective).

In general, the features used for crowd counting can be categorised under the following headings:

1. *Size* refers to the magnitude of any interesting segments extracted from an image which are deemed to be relevant, such as the foreground pixel count (Section 3.2.1).
2. *Shape* pertains to the orientation and shape descriptors of these segments detected in an image (Section 3.2.2).
3. *Edges* refer to the relative change in pixel intensities across an image, measured using a binary edge detector (Section 3.2.3).
4. *Keypoints* are any other points of interest, such as corners, that are detected in an image (Section 3.2.4).

3.2.1. Size

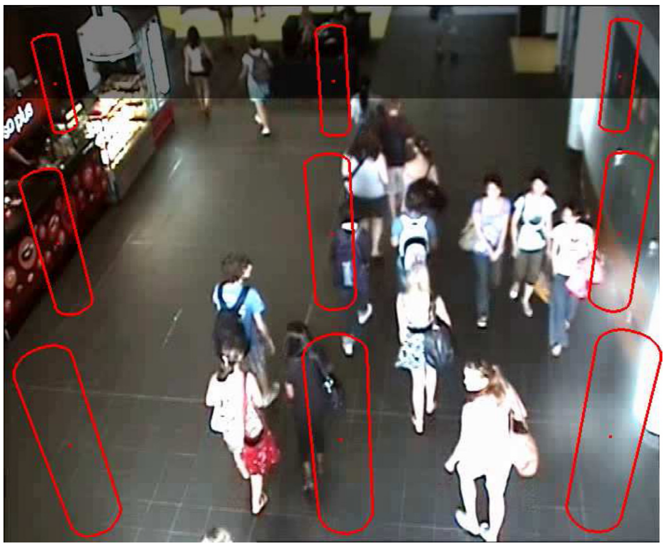
Size refers to the magnitude of any detected regions, such as foreground motion segments, in an image. Davies et al. (1995) proposed the use of the foreground pixel count as a measure of the holistic crowd size, while Ma et al. (2004) introduced the density map to weight each foreground pixel to compensate for perspective.

The set of foreground pixels within the region of interest (ROI) is denoted B. The foreground is segmented into a set of connected components, which are individually labelled, and enumerated by n. The notation B_n is used to represent the set of pixels which belong to the nth blob. The collection of blobs $\{B_n\}$ is a *partition* of the set B.

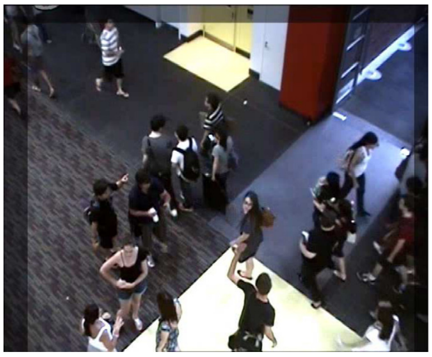
The *weighted area* of each blob is denoted A_n. This is calculated using the density map, S, to account for perspective:

$$A_n = \sum_{(i,j)\in B_n} S(i,j) \tag{2}$$

Another size feature is perimeter length. The set of perimeter pixels P_n is obtained by tracing along the boundary of the nth blob. Perimeter pixels are a one-dimensional feature, and are thus weighted using the square root of the density map S as in Chan et al. (2008). The weighted perimeter of the nth blob segment is therefore calculated as follows:



(a) Camera 3. A camera calibration technique [23] is used to project a prototypical human-sized cylindrical object into the scene at any location. This hypothetical cylinder model is used to construct the density map $S(i,j)$ at every pixel (Equation 1). The density map is used to normalise features to achieve scene invariance. Only a sparse subset of the cylinder models are shown here; however, the system constructs them around every pixel (during initialisation of the system only).



(b) Camera 5.

(c) Camera 8.

Fig. 1. Images from the QUT camera network.

$$L_n = \sum_{(i,j)\in P_n} \sqrt{S(i,j)} \tag{3}$$

The perimeter length may provide valuable size information when the foreground segment erroneously contains ‘holes’. It also supplements the area feature to provide a more complete description of group size.

3.2.2. Shape

Perimeter pixels provide valuable shape information about an object. Aside from the perimeter length, which measures the object *size*, the orientation of the perimeter pixels also contain important shape information. Such features have been used previously in Chan et al. (2008) and Denman et al. (2007).

Perimeter pixels are easily detected by tracing around the boundary of an object. It is intuitive and computationally efficient to use an orientation histogram with 4 bins, each corresponding to the direction of an adjacent pixel $(0^\circ, 45^\circ, 90^\circ, 135^\circ)$. When tracing the perimeter from one boundary pixel to the next, the direction of movement determines which histogram bin receives the pixel’s vote.

The vote weight is the square root of the density map, $\sqrt{S(i,j)}$, as perimeter pixels are a one dimensional feature. Vertical edges are more likely to indicate individuals in a scene, whereas a combination of many perimeter pixels at all orientations may indicate larger crowds.

The four shape features stored in the perimeter orientation histogram are denoted $V_n(h)$, for $h \in [0,3]$.

3.2.3. Edges

Edges have been commonly used in crowd counting systems. For example, Kong et al. (2006) introduced the use of an edge angle histogram on a holistic scale, while Davies et al. (1995), Chan and Vasconcelos (2008) and many others have used the total number of edge pixels on a holistic level, regardless of orientation.

In the proposed algorithm, an edge orientation histogram is constructed for each foreground segment in an image using the following procedure. The set of edge pixels belonging to the nth blob segment is denoted E_n, and a histogram of edge orientations H_n is constructed by allocating each edge pixel to a histogram channel, based on the pixel’s orientation $\angle G(i,j)$. The orientation bins are evenly divided over the range $[0,\pi]$, and a total of 6 bins are used.

Each edge pixel within the blob contributes a weighted vote to a histogram bin. This contribution (or vote) is equal to the square root of the density map, $\sqrt{S(i,j)}$, to normalise for perspective. If the value of the hth histogram bin is denoted $H_n(h)$, and the orientation angle for that bin is lower-bounded by θ_h:

$$H_n(h) = \sum_{(i,j)\in E_n} \begin{cases} \sqrt{S(i,j)} & \text{if } \theta_h \leqslant \angle G(i,j) < \theta_{h+1} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The edge orientation histogram "can distinguish edges caused by pedestrians, which are usually vertical, with other scene structures such as noise, shadows and cars" (Kong et al., 2006). Vertical edges are considered to be most indicative of human crowding, because they are "really important for detecting the bodies (i.e. legs and arms)" (Regazzoni et al., 1993). Edges also help to identify occlusions when multiple pedestrians partially block one another from view. Although the blob's *size* features are reduced by occlusions, the edge features become stronger due to the overlapping body parts, differing skin tones and conflicting clothing.

Canny edge detection is used due to its use of non-maximum suppression and hysteresis thresholding which results in a cleaner output.

3.2.4. Keypoints

Keypoints refer to specific pixels of interest, such as corners, which are detected in an image. Keypoints are useful for detecting salient points of interest in a scene, and these are often indicative of human crowding. For example, Conte et al. (2010) used speeded-up robust features (SURF), as introduced by Bay et al. (2008), to detect keypoints within an image. These points were masked by optical flow so that stationary points were ignored. The number of moving keypoints was used to predict crowding. Similarly, Albiol (Albiol and Silla, 2009) utilised Harris corners (Harris, 1988) to estimate crowd size on a holistic level.

Two types of feature detectors are considered for the proposed algorithm. Firstly, corners are detected using the 'FAST' algorithm recently proposed by Rosten et al. (2010), and the set of keypoints detected within the foreground blob segment n is denoted κ_n^{FAST}. Secondly, SURF keypoints (Bay et al., 2008) are extracted and this set of keypoints is denoted κ_n^{SURF}.

The two keypoint features are then calculated as follows:

$$K_n^{FAST} = \sum_{(i,j)\in\kappa_n^{FAST}} \sqrt{S(i,j)} \tag{5}$$

$$K_n^{SURF} = \sum_{(i,j)\in\kappa_n^{FAST}} \sqrt{S(i,j)} \tag{6}$$

Note that the notation κ_n^{FAST} is used to refer to a *set* of keypoints, while K_n^{FAST} represents the scalar keypoint feature that is calculated from this set.

The keypoints are masked by the foreground detection result (the summation only takes place across each foreground segment), so that keypoints belonging to background objects and surrounding structures are not included in the feature vector.

3.2.5. Full feature vector

The full feature vector used to represent blob n is denoted $\mathbf{x}_n$ and is comprised of size, shape, edge and keypoint features:

$$\mathbf{x}_n = [A_n, L_n, V_n(0), \ldots, V_n(3), H_n(0), \ldots, H_n(5), K_n^{SURF}, K_n^{FAST}] \tag{7}$$

Various subsets of these features are evaluated in Section 6.1.1 to determine the optimal feature set.

3.3. System training and testing

Training is performed on the local level by annotating each blob with the number of pedestrians contained therein, as described in Ryan et al. (2012). The number of people represented by the nth blob in the training dataset is denoted f_n, and the feature vector extracted from this segment is denoted $\mathbf{x}_n$. In order to train the proposed system, a regression function must be learned using the set of training examples, $\{\mathbf{x}_n, f_n\}_{n=1}^N$, to count the number of people present in each group. In this case we use n to enumerate all of the blobs observed in the entire training dataset, rather than just one frame.

Existing approaches use linear regression (Davies et al., 1995; Kong et al., 2006; Ryan et al., 2010), neural networks (Marana et al., 1997; Kong et al., 2006; Ryan et al., 2009) and Gaussian process regression (Chan et al., 2008). Although the linear model has demonstrated acceptable 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.

The Gaussian Process may be thought of as an infinite collection of random variables, any finite subset of which have a joint Gaussian distribution (Rasmussen, 2004). The targets $\mathbf{f} = \{f_n\}$ are envisioned as a sample from this distribution:

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}) \tag{8}$$

where the covariance matrix $\mathbf{K} \in \mathbb{R}^{N\times N}$ is defined by a function $k(\mathbf{x}_n, \mathbf{x}_m)$ which relates the covariance of outputs as a function of inputs:

$$\mathbf{K}_{nm} = k(\mathbf{x}_n, \mathbf{x}_m) \tag{9}$$

The subscripts n and m represent the indices of two samples. A typical covariance function is the squared exponential (Rasmussen, 2004) which enforces a high covariance when the samples are close within the input space. This captures the intuition that similar inputs produce similar outputs. Samples from the test data are also envisioned as belonging to this joint distribution,

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K}^* \\ \mathbf{K}^{*T} & \mathbf{K}^{**} \end{bmatrix}\right) \tag{10}$$

where $\mathbf{f}^* = \{f_n^*\}$ denotes the unseen targets in the test dataset, and the covariance submatrices are calculated as follows:

$$\mathbf{K}_{nm}^* = k(\mathbf{x}_n, \mathbf{x}_m^*) \tag{11}$$

$$\mathbf{K}_{nm}^{**} = k(\mathbf{x}_n^*, \mathbf{x}_m^*) \tag{12}$$

where $\mathbf{x}_n^*$ and $\mathbf{x}_m^*$ denote feature vectors in the test set. Following from Eq. (10), prediction using GPR is performed by conditioning the outputs on the training data:

$$\mathbf{f}^*|\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{13}$$

where:

$$\boldsymbol{\mu} = \mathbf{K}^{*T}\mathbf{K}^{-1}\mathbf{f} \tag{14}$$

$$\boldsymbol{\Sigma} = \mathbf{K}^{**} - \mathbf{K}^{*T}\mathbf{K}^{-1}\mathbf{K}^* \tag{15}$$

The point estimates for $\mathbf{f}^*$ are represented by $\boldsymbol{\mu}$ and the variance for each estimate is represented by the diagonal elements of $\boldsymbol{\Sigma}$ which can be used to generate confidence intervals.

The covariance function used in our system is adopted from previous crowd counting applications (Chan et al., 2008; Ryan et al., 2012):

$$k(\mathbf{x}_n, \mathbf{x}_m) = \alpha_1^2 \exp\left(\frac{|\mathbf{x}_n - \mathbf{x}_m|^2}{-2\ell^2}\right) + \alpha_2^2(1 + \mathbf{x}_n^T \mathbf{x}_m) + \alpha_3^2 \delta_{n,m} \tag{16}$$

This function is comprised of a squared exponential term which captures short-range trends in the data; a linear term which captures long-range trends; and a noise term which contributes only to the diagonals of the covariance matrix. The hyperparameters $\{\alpha_1, \ell, \alpha_2, \alpha_3\}$ are selected automatically during training to optimise the log-likelihood of the observed training data (following from Eq. (8)):

$$\log p(\mathbf{f}) = -\frac{1}{2}\mathbf{f}^T\mathbf{K}^{-1}\mathbf{f} - \frac{1}{2}\log|\mathbf{K}| - \frac{N}{2}\log 2\pi \tag{17}$$

This term is maximised using an optimisation algorithm such as conjugate gradients. Once optimised, prediction is then performed using Eqs. (13)–(15). The reader is referred to Rasmussen (2004) and Rasmussen and Williams (2006) for additional details regarding GPR.

The holistic estimate for a test frame is the sum of the group size estimates. If the blobs in a test frame are enumerated by n, and the group estimates are denoted $\{\mu_n\}$, then the holistic crowd estimate for that frame is:

$$e = \sum_n \mu_n \tag{18}$$

An additional group tracking module is used to smooth the estimate as outlined in Ryan et al. (2012).

4. Multi camera crowd counting

The algorithm proposed in Section 3 is scene invariant, and therefore lends itself naturally to crowd counting across multiple cameras in a single environment. In this scenario, the same area is monitored using two or more cameras, with some potential overlap between the views. It is this overlap which presents a challenge in reconciling the crowd counts across all viewpoints.

A naive approach to multi camera crowd counting would be to take the sum of the crowd counts from each viewpoint. However, some pedestrians will appear in two or more cameras and will therefore be counted multiple times. One approach to deal with this scenario is to attempt to match groups between viewpoints and perhaps to identify individuals within groups. A complication with this approach is that groups segmented in one viewpoint will not necessarily correspond to those groups segmented in another. This is observed in Fig. 2; the 'groups' that will be segmented from one angle are significantly different from another.

We seek to avoid the difficult problem of detecting individuals or matching blobs of various configurations. Two modifications to the existing algorithm of Section 3 are proposed which take into account the *overlap* between the viewpoints:

1. *Density map modification:* This approach modifies the density map, S, in regions of overlap so as to effectively compensate for the 'double-up' that occurs when a person is visible in more than one camera. This is done by reducing the density assigned to pixels in overlap regions. Counting is then performed as a subsequent step using this modified density map.
2. *Pixel density assignment:* This approach leaves the density map unaltered, so that the system described in Section 3 operates unchanged. Instead, crowd densities are modified after counting has been performed, on a pixelwise basis.

Both approaches rely on the construction of an *overlap map*, which provides a measure of how much of an object centred around a pixel is visible within other viewpoints.

The section is structured as follows: Section 4.1 describes the overlap map; Section 4.2 discusses the first approach, density map modification; Section 4.3 describes the second approach, pixel density assignment; and Section 4.4 discusses some alternative baseline approaches.

4.1. The overlap map

In a multi camera network there will be several video streams corresponding to the various cameras in the facility. Let the cameras monitoring a space be enumerated by v. This notation will be used as a superscript on the existing notation.

First we consider the pixel coordinates (i,j) in the image plane of camera v, around which a hypothetical object is centred. The cylinder model described in Section 3.1 is used for this purpose, with radius $r = 0.25$ m and height $h = 1.7$ m in real world measurements.

Using camera calibration, let $(i,j)^{v\to u}$ denote this object's centre in the image plane of camera u. That is, $(i,j)^{v\to u}$ denotes the conversion of an object's centre in image plane v to its centre in image plane u.

Let us also define $R_{i,j}^v$ as the region occupied by a cylinder model centred at (i,j) in the image plane of camera v. $|R_{i,j}^v|$ represents the *area* of the cylinder model when projected into image plane v. We also use M^v to denote the region of interest mask for viewpoint v. The intersection of these regions in the image plane is $R_{i,j}^v \cap M^v$, and this represents the portion of the model which lies within the region of interest. The *fraction* of the model within the ROI is therefore:

$$\frac{|R_{i,j}^v \cap M^v|}{|R_{i,j}^v|} \tag{19}$$

The cylinder model $R_{i,j}^v$ can be projected into another viewpoint, u, and this is denoted $R^u_{(i,j)^{v\to u}}$. The *overlap map* provides a measure for how much of this cylinder, $R_{i,j}^v$, is projected into other views. The overlap map for viewpoint v is therefore:

$$O_{i,j}^v = \sum_{\forall u \neq v} \frac{|R^u_{(i,j)^{v\to u}} \cap M^u|}{|R^u_{(i,j)^{v\to u}}|} \tag{20}$$

This is the value of the overlap map at pixel (i,j) in view v. The summation is taken across all *other* viewpoints, enumerated by u, and the summand is the fraction of the cylinder model (centred around pixel (i,j) in view v) projected into the other viewpoints, u.

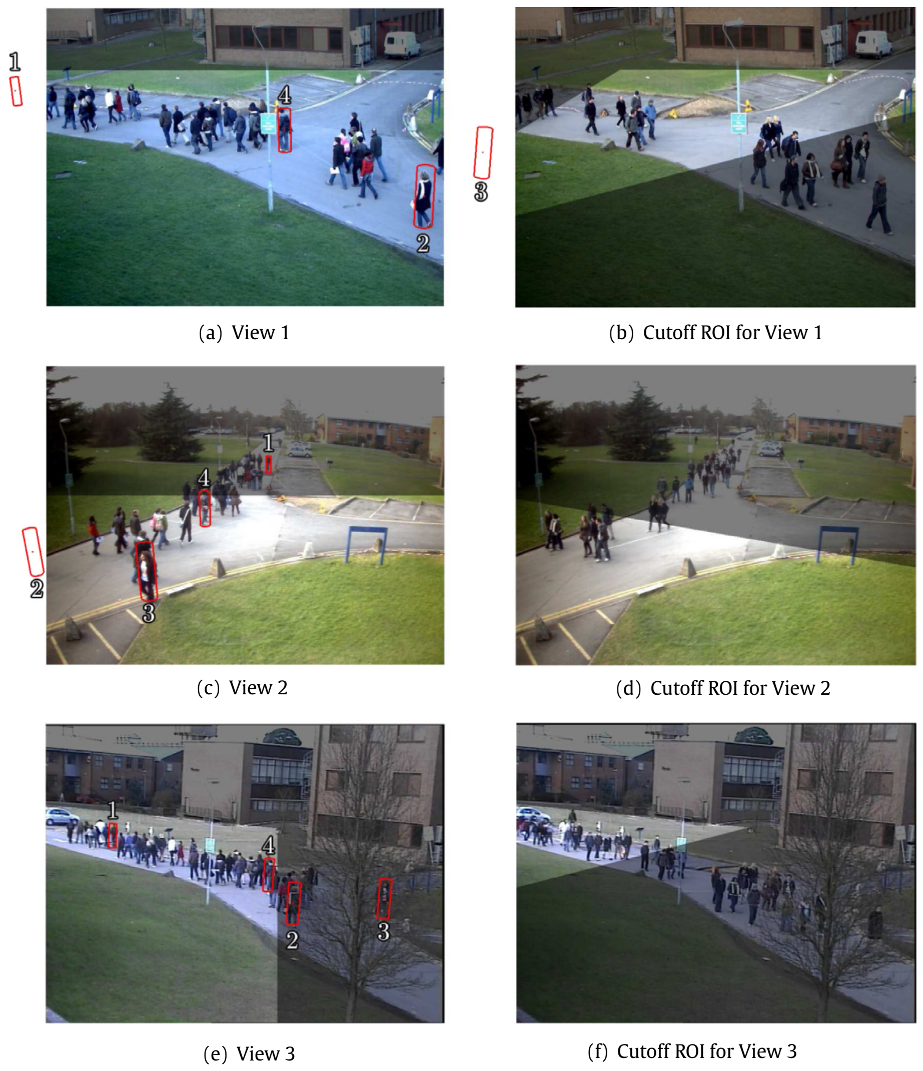
For example, if an object centred at (i,j) in viewpoint v is not projected into any other viewpoints, there is no overlap and hence $O_{i,j}^v = 0$. However, if 50% of the object is projected into another viewpoint, then $O_{i,j}^v = 0.5$.

In some cases there may be more than two viewpoints. Eq. (20) is taken as a summation across all other viewpoints, so that the total number of projections of the cylinder model into other viewpoints is included in the calculation. Fig. 2 illustrates this. The pedestrian labelled '4' in View 2 (Fig. 2(b)) is projected 100% into both other views (Figs. 2(a) and 2(c)); therefore the value of the overlap map in View 2 is 2.0 at this person's centroid.

The overlap map provides a measure of how much of an object centred at a particular pixel is visible in the other viewpoints. This enables us to compensate for over-estimation of the crowd in regions of overlap, as described in Sections 4.2 and 4.3.

4.2. Density map modification

Given a fully trained crowd counting system, as described in Section 3, we seek to perform counting across multiple viewpoints. The method proposed in this section is called density map modification because it modifies the density map, S, in regions of overlap.



(a) View 1 (b) Cutoff ROI for View 1 (c) View 2 (d) Cutoff ROI for View 2 (e) View 3 (f) Cutoff ROI for View 3

Fig. 2. The PETS 2009 database with ROIs highlighted and a sample of manually-labelled pedestrian annotations. Pedestrians 1–3 indicate unique coverage areas, while pedestrian 4 indicates an overlap region.

(The density map was originally described in Section 3.1.) Let Δ^v to denote the *modified* version of the original density map, S^v, as follows:

$$\Delta^v(i,j) = \frac{S^v(i,j)}{1 + O^v_{i,j}} \tag{21}$$

The denominator contains two terms, $O^v_{i,j}$, which represents the projection of an object into other viewpoints, and '1', which represents the object's presence in the current viewpoint, v.

If an object centred at (i,j) in viewpoint v is 100% visible in another viewpoint u, then the value of the overlap map will be $O^v_{i,j} = 1$, so that the denominator of Eq. (21) is exactly 2, resulting in a halving of the original density map's value. This compensates for the 'double-up' which would otherwise have occurred by counting the object twice. A smaller compensation will be applied when an object is only partially visible in another viewpoint, and no compensation is applied when the overlap is zero. This is due to the way in which the overlap map is calculated (Eq. (20)), which may be fractional when a cylinder model is only partially projected into another viewpoint.

The density map modification described in Eq. (21) is only applied *after* the system has been fully trained on a number of individual viewpoints, but prior to the system being deployed across a multi camera environment. Once the density map has been modified, the crowd estimate for viewpoint v is denoted $\hat{e}^v$ so that the holistic estimate across all viewpoints can be summed directly:

$$e = \sum_v \hat{e}^v \tag{22}$$

This is the global count for the number of pedestrians in the scene. The value of $\hat{e}^v$ no longer provides a meaningful representation of the number of people in viewpoint v due to the modification of the density map.

4.3. Pixel density assignment

This method does not alter the density map, and instead allows the system to operate as intended, performing overlap compensation as a subsequent step. The 'crowding density' at each pixel is utilised to modify the crowd count in a pixelwise manner.

The crowding density applied to a pixel is calculated as follows. Let μ^v_n denote the crowd estimate for blob n in viewpoint v. The set of pixels belonging to this blob is denoted B^v_n, and $C^v(i,j)$ denotes the identity of the blob to which pixel (i,j) belongs. Finally, the 'crowding density' at pixel (i,j) is calculated by:

$$d^v(i,j) = \frac{F^v(i,j)S^v(i,j)}{\sum_{(i',j')\in B^v_{C^v(i,j)}} S^v(i',j')} \mu^v_{C^v(i,j)} \tag{23}$$

where F^v denotes the foreground detection result (with 0 and 1 representing the background and foreground pixels, respectively). In this way, the crowd count for blob $C^v(i,j)$ is split between its constituent pixels, weighted by the density map S^v to assign higher crowd density to more distant points in the scene to account for perspective.

Given this pixelwise crowd density, the overlap map is subsequently used to *modify* the crowd density in order to compensate for overlap between cameras and to avoid counting people multiple times. Let us use $\delta^v(i,j)$ to denote the *modified* version of the original crowding density, $d^v(i,j)$, as follows:

$$\delta^v(i,j) = \frac{d^v(i,j)}{1+O^v_{i,j}} \tag{24}$$

The denominator in Eq. (24) serves the same purpose as in Eq. (21). The holistic count across all viewpoints is therefore the summation of all of the modified crowding densities across all viewpoints:

$$e = \sum_v \sum_{(i,j)} \delta^v(i,j) \tag{25}$$

This is the global count for the number of pedestrians in the scene.

4.4. Baseline approaches

This section describes two additional approaches for crowd counting across multiple viewpoints, against which the proposed methods in Sections 4.2 and 4.3 will be compared.

The first approach is referred to as the ‘naive’ method, in which the crowd counting algorithm of Section 3 operates unchanged, and the total crowd count is defined as the sum of the crowd counts e^v from each individual camera:

$$e = \sum_v e^v \tag{26}$$

It is expected that this approach will overestimate the true crowd size due to camera overlap. An example of camera overlap in a three-camera setup is shown in Fig. 2. The labelled annotations of four pedestrians indicate the regions which are unique to each viewpoint, as well as an overlap region.

The second approach is referred to as the ‘cutoff’ method, in which the ROI for each viewpoint is intentionally cut off to avoid any overlap between the viewpoints. This is based on a hypothetical straight line along the ground plane which is used to separate the views, as shown in Fig. 2.

Although the cutoff method is straightforward to construct for our experiments, it may not always be as easy to segment an arbitrary number of viewpoints from multiple angles using this technique.

5. Evaluation protocol and data

The predictive performance of the crowd counting algorithm is evaluated using three criteria: the *mean absolute error* (MAE), the *mean square error* (MSE) and *mean relative error* (MRE). These metrics are commonly used within the field for evaluating system performance (Davies et al., 1995; Lempitsky and Zisserman, 2010; Chan et al., 2008; Conte et al., 2010).

These metrics compare the crowd estimate to the ground truth. The definition of ground truth within a multi camera environment is discussed in Section 5.1. The datasets used in this evaluation are described in Section 5.2.

5.1. Ground truth annotation

In a single camera environment, the definition of ground truth is relatively straightforward: the number of pedestrians within the region of interest is taken as the ground truth, with fractional counts being assigned to pedestrians who are partially within the ROI (Ryan et al., 2012). In the case of multiple cameras, a pedestrian may be partially visible in more than one camera, complicating the definition of ground truth. This section defines holistic ground truth in a multi camera environment to be the *maximum* fraction of person to be visible within any camera in the environment.

Each person, enumerated by p, is annotated with a real world coordinate, (x,y,z) with $z=\frac{h}{2}$, located at approximately the centre of their body. A cylindrical pedestrian template is projected around this point into each of the camera viewpoints, occupying a region in image plane v which we denote R^v_p.[1]

The boundary of the pedestrian template, R^v_p, roughly covers the person in each viewpoint (Fig. 2). We calculate the ‘quantity’ of person p within each region of interest mask M^v:

$$Q^v_p = \frac{|M^v \cap R^v_p|}{|R^v_p|} \tag{27}$$

We can then determine the maximum quantity of person p within the *scene* (across all viewpoints):

$$Q_p = \max_v Q^v_p \tag{28}$$

Thus the holistic ground truth for the scene is taken to be:

$$Q = \sum_p Q_p \tag{29}$$

This definition of ground truth is used for evaluating the performance of the proposed algorithm in Section 6.2.

5.2. Datasets

Seven viewpoints were used to evaluate the scene invariant capabilities of the proposed algorithm.

1. View 1 from the *PETS 2009* dataset introduced at the Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS, 2009). This dataset was also used subsequently at the PETS 2012 workshop (PETS, 2012).
2. View 2 from the *PETS 2009* dataset.
3. View 3 from the *PETS 2006* dataset introduced at the Ninth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS, 2006).
4. View 4 from the *PETS 2006* dataset.
5. Camera 3 from the *QUT* camera network.
6. Camera 5 from the *QUT* camera network.
7. Camera 8 from the *QUT* camera network.

These datasets are summarised in Table 1.

The PETS 2009 database includes sequences for evaluating crowd counting systems, and the video is provided as a resolution of 768×576 pixels and $\sim$7 fps in RGB colour format. Example frames from Cameras 1 and 2 are shown in Fig. 2. The PETS 2006 database is designed to evaluate object tracking algorithms and abandoned luggage detection. As the sequences show pedestrians passing through the scene it is suitable for person counting. The

[1] The notation R^v_p in this section is used to represent the template of person p projected into image plane v, whereas the use of $R^v_{i,j}$ in Section 3.1 was used to represent a template centred around pixel (i,j) in image plane v.

video is provided at a resolution of 720 × 576 pixels and 25 fps in RGB colour format.

To supplement the existing public datasets, a new database has been developed containing footage obtained from the Queensland University of Technology's campus. This database is referred to as 'QUT' and contains data selected from a camera network installed on a single floor of a building.

This database contains three challenging viewpoints, which are referred to as Camera 3 (Fig. 1(a)), Camera 5 (Fig. 1(b)) and Camera 8 (Fig. 1(c)). The sequences contain reflections, shadows and difficult lighting fluctuations, which makes crowd counting difficult.

Previous crowd counting datasets have been substantially shorter in length than those included in the QUT database. For example, PETS 2009 contains crowd counting sequences just over 200 frames in length. 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 this may lead to biased results that do not adequately describe a system's true performance capabilities.

In order to combat this problem, the QUT datasets are annotated at sparse intervals: typically 100–400 frames apart depending on crowd variation and sequence length. 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.

The following cross validation procedure is followed to evaluate system performance across various configurations:

1. Seven datasets are annotated with ground truth data.
2. One dataset is selected and set aside for testing. The proposed crowd counting system is trained using ground truth from the *other* six datasets. Testing is then performed on the selected test dataset. Error metrics are calculated across all frames: mean absolute error (MAE), mean square error (MSE) and mean relative error (MRE) are reported.
3. This procedure is repeated for all datasets, by rotating the training and testing sets. Error metrics are then combined across all datasets using average rank and average error, with equal weighting given to each dataset.

Due to the variation in error rates on different viewpoints, those datasets with larger fluctuations tend to dominate the mean error calculation. For this reason the various system configurations are *ranked* in order of performance, and the average ranking across all datasets is reported. For completeness the mean error rates are also reported. These results are presented in Section 6.1.

Five viewpoints were used to evaluate the proposed multi camera crowd counting algorithm: Views 1–3 from the PETS 2009 database (PETS, 2009); and Views 3 and 4 from the PETS 2006 database (PETS, 2006). These datasets feature crowds of size 1 to 43 people in various lighting conditions and differing camera angles. The PETS datasets are used because the sequences are captured at the same location simultaneously and are therefore suitable for multi camera crowd counting. These results are presented in Sections 6.2 and 6.3.

6. Evaluation

6.1. Scene invariant evaluation

In this section, K-fold cross validation is used to evaluate a number of feature sets and regression models for scene invariant crowd counting. Section 6.1.1 evaluates the feature sets and Section 6.1.2 evaluates the regression models. Section 6.1.3 compares the proposed algorithm to existing scene-specific methods.

6.1.1. Feature evaluation

The proposed scene invariant algorithm is trained on multiple viewpoints, which serve as a bank of examples, so that future crowd counting can be performed on an unseen viewpoint without any additional training data being provided for that viewpoint.

Aside from size based features, it is unclear how well other types of features (shape, edges and keypoints) will scale between viewpoints. This section assesses various feature combinations in order to determine the optimal feature set for scene invariant crowd counting.

As described in Section 3.2, features are divided into the following categories: *size*, *shape*, *edges* and *keypoints*. Various combinations of features are assessed using the evaluation procedure of Section 5.2 in order to determine the most accurate feature sets. Fifteen different combinations of features are listed in Table 2.

Error metrics were calculated for each dataset, and the feature sets were ranked from 1 (lowest error rate) to 15 (highest). The *average rank* for each feature set across all datasets is reported in Table 2. For example, 'keypoint' features taken alone attained an average rank of 13.0 (out of 15) in terms of MAE, indicating consistently poor performance across all datasets. This is also seen for 'shape' features. A better ranking is observed when *multiple*

Table 1
Summary of the various conditions in the benchmark datasets (PETS, 2006, 2009).

	PETS-09, View 1	PETS-09, View 2	PETS-06, View 3	PETS-06, View 4	QUT, Camera 3	QUT, Camera 5	QUT, Camera 8
Frames	1565	221	6422	6422	3100	10,000	6100
Sequences	5	1	2	2	1	1	2
Train frames	20	11	51	50	7	28	7
Test frames	1565	22	51	50	30	57	40
Frame rate	~7	~7	25	25	25	25	25
Resolution	768 × 576	768 × 576	720 × 576	720 × 576	704 × 576	352 × 288	704 × 576
Colour	RGB	RGB	RGB	RGB	RGB	RGB	RGB
Location	Outdoor	Outdoor	Indoor	Indoor	Indoor	Indoor	Indoor
Shadows	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Reflections	No	No	No	No	No	No	Yes
Loitering	No	No	Yes	Yes	Yes	Yes	Yes
Crowd size	0 to 42	0 to 43	0 to 5	0 to 7	4 to 21	3 to 23	2 to 20

Table 2
Average rank across *all seven datasets*, when features are ranked from 1 to 15 on each dataset. Values shown are not actual error rates, but rather an average ranking.

Features	Average rank		
	MAE	MSE	MRE
Size	8.14	8.43	6.86
Shape	12.43	12.86	12.14
Edges	9.57	10.71	9.43
Keypoints	13.00	13.29	13.86
Size, Shape	8.57	8.14	8.29
Size, Edges	6.14	5.86	5.71
Size, Keypoints	5.71	5.14	5.57
Shape, Edges	8.00	8.14	7.57
Shape, Keypoints	10.71	10.43	11.00
Edges, Keypoints	10.57	10.71	11.14
Size, Shape, Edges	4.86	4.86	5.14
Size, Shape, Keypoints	5.57	5.43	6.43
Size, Edges, Keypoints	5.14	5.00	5.57
Shape, Edges, Keypoints	7.71	7.43	7.29
Size, Shape, Edges, Keypoints	**3.86**	**3.57**	**4.00**

features are used (especially when size is included). Best performance is seen when all features are used.

Table 3 lists the average error rates across all datasets, weighted equally, when all features are used. The mean absolute error is 1.351 and the mean relative error is 15.92%. This falls within the 20% threshold of acceptability suggested by Regazzoni et al. (1993). The PETS 2006 datasets do exceed this threshold, although this is due primarily to their small crowds of only 1–7 people, rather than inaccurate counting results; the mean absolute error rate is less than 0.5 for these datasets.

Fig. 3 plots the estimate of the proposed algorithm (using all features) against the ground truth for a number of sequences from the benchmark datasets.

These results provide strong support for the proposed scene invariant algorithm. The evaluation has demonstrated that the use of all features consistently outperformed any subset thereof.

6.1.2. Regression model evaluation

In this section, K-fold cross validation is used to evaluate a number of regression models used to perform the crowd estimation. For comparison we use Gaussian process regression (GPR) as proposed in Section 3.3; ordinary least squares linear regression (linear); K-nearest neighbours (KNN) with $K = 1, 2, 4, 8, 16, 32$; and a neural network (NN) with a Sigmoid activation function and one hidden input layer (containing 4, 8, 16 or 32 neurons). In total there are 12 regression models with various parameters. These regression models have been used previously in various crowd counting applications such as Kong et al. (2006), Davies et al. (1995), and Chan et al. (2008).

Error metrics were calculated for each dataset, and the regression models were ranked from 1 to 12. The *average rank* for each regression model across all seven datasets is reported in Table 4.

Table 3
Error rates for all datasets when *all features* are used. The *average* error rate across all datasets is also shown. Each dataset is weighted equally.

Dataset	MAE	MSE	MRE
PETS 2009, View 1	1.321	4.250	10.32%
PETS 2009, View 2	3.365	17.514	9.55%
PETS 2006, View 3	0.405	0.495	24.98%
PETS 2006, View 4	0.487	0.441	22.20%
QUT, Camera 3	1.574	3.506	14.03%
QUT, Camera 5	0.886	1.524	12.17%
QUT, Camera 8	1.448	3.625	18.20%
Average	1.351	4.479	15.92%

Gaussian process regression consistently ranks highest, with an average rank of 2.43 (out of 12) in terms of MAE. This is followed by linear regression (average rank 4.57 in terms of MAE) and K-nearest neighbours.

These results provide very strong support for the use of Gaussian process regression in the proposed algorithm, compared to linear, KNN or neural network regression.

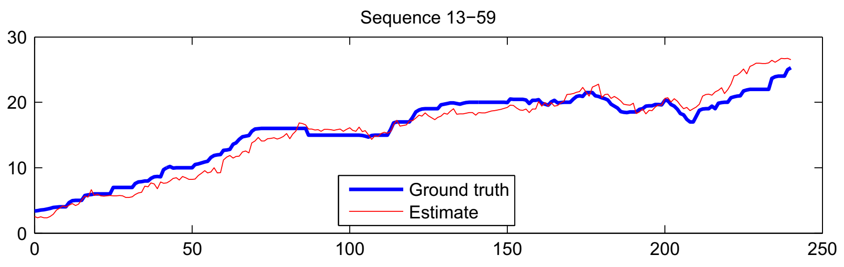
6.1.3. Comparison to other algorithms

In this section the performance of the proposed system is assessed in comparison to other algorithms in the literature. Note that these algorithms are *scene-specific*, and therefore training and testing take place on the same viewpoint (with different sequences used for training and testing). By contrast, the proposed algorithm is trained on a bank of viewpoints and tested on an unseen one. The following algorithms are evaluated:

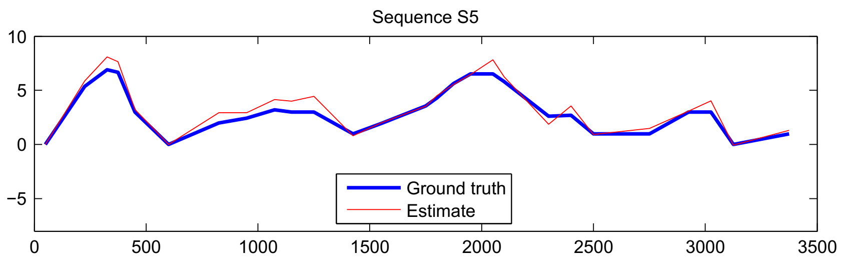
1. *Chan* (Chan et al., 2008; Chan and Vasconcelos, 2009). Bidirectional segmentation is performed using dynamic textures. Holistic features are used to count the number of pedestrians moving in each direction, from which a holistic count is obtained.
2. *Albiol* (Albiol and Silla, 2009; Conte et al., 2010). Moving corner points are extracted and linear regression is applied on a holistic level.
3. *Conte* (Conte et al., 2010). This is an extension of Albiol's work: moving SURF points are clustered and regression is applied to these clusters on a local level.
4. *Kong* (Kong et al., 2006). Blob size histograms and edge angle histograms are accumulated on a holistic level.
5. *Local Features*. This refers to the scene-specific version of the proposed algorithm; training and testing takes place on the same viewpoint. The full feature vector is used (size, shape, edges, keypoints).
6. *Holistic Features*. This refers to the *equivalent holistic* version of 'Local Features'. In this approach, feature extraction and regression takes place on the holistic level. For example, the *area* feature is computed as $A = \sum_n A_n$, where A_n is the area of blob n as defined in Eq. (2). All holistic features are computed in this manner. The full feature vector is used (size, shape, edges, keypoints).
7. *Scene invariant*. This refers to the proposed algorithm.

The PETS 2009 database (PETS, 2009) is used to compare these algorithms, as it has been widely used by various authors for evaluating crowd counting performance. Chan's results were reported in Chan and Vasconcelos (2009), and Albiol and Conte's results were reported in Conte et al. (2010).

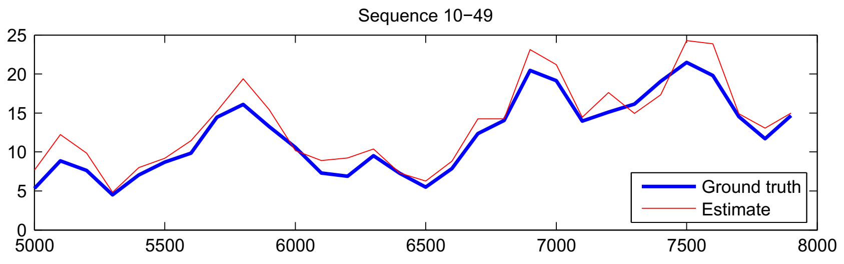
For this evaluation, Kong's algorithm was implemented as faithfully as possible to Kong et al. (2006), however some assumptions were necessary. Although Kong used a bin width of 500 for the blob size histogram, this value is not be suitable for all datasets due to differences in image resolution and camera distance with respect to the scene. Instead, the bin width is set to roughly $\frac{2}{3}$ of the size of a person in the scene, so that smaller blobs (noise) are assigned to the first histogram bin and larger groups occupy the other bins. This provides good separation between different blob sizes, as is the intent of the algorithm. The bin width is calculated by positioning a pedestrian template at the centre of an image and taking the sum of weighted pixels belonging to the template, the result of which is multiplied by $\frac{2}{3}$. Six bins are used for the blob size histogram, and eight blobs are used for the edge angle histogram, as proposed by the author in Kong et al. (2006). Kong proposed the use of neural networks and linear regression. These regression models are evaluated in addition to GPR for completeness.



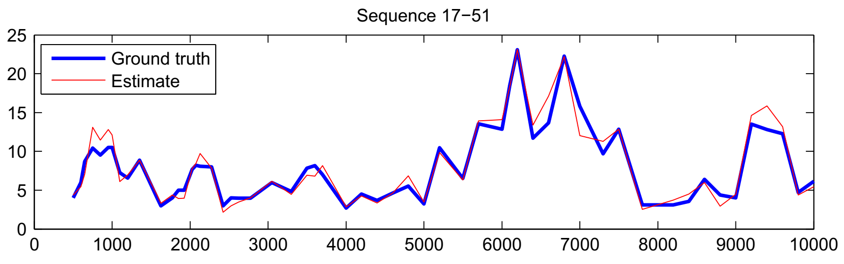
(a) PETS 2009, View 1.



(b) PETS 2006, View 4.



(c) QUT, Camera 3.



(d) QUT, Camera 5.

Fig. 3. Scene invariant crowd counting results on a number of sequences from the benchmark datasets described in Section 5.2.

Table 4
Average rank across *all datasets*, when regression models are ranked from 1 to 12 on each dataset. Values shown are not actual error rates, but rather an average ranking.

Regression model	MAE	MSE	MRE
GPR	**2.43**	**2.71**	**2.43**
Linear	4.57	4.43	4.71
KNN ($K = 1$)	6.14	7.00	6.14
KNN ($K = 2$)	5.43	5.43	5.00
KNN ($K = 4$)	4.57	5.00	4.57
KNN ($K = 8$)	5.29	4.86	5.14
KNN ($K = 16$)	5.14	5.00	5.43
KNN ($K = 32$)	4.71	4.86	4.57
NN (4)	10.29	9.86	10.29
NN (8)	9.71	9.71	9.86
NN (16)	9.14	8.71	9.43
NN (32)	10.57	10.43	10.43

The proposed algorithm, ‘Scene Invariant’, was evaluated by training the system on the six other viewpoints listed in Table 1, and then testing on PETS 2009, View 1. The scene-specific approaches, ‘Local Features’ and ‘Holistic Features’, were trained and tested on the same viewpoint (with different sequences being used for training and testing).

Results are presented in Table 5. For sequence 13-57, the best performance is observed with the use of ‘Local Features’, followed by Chan’s algorithm and Conte’s algorithm. These are all scene-specific approaches. The proposed scene invariant algorithm attains a MRE of 11.30% and outperforms the remaining approaches (Albiol, Kong and ‘Holistic Features’). These results indicate that the proposed algorithm provides similar performance to existing methods on this sequence.

For the remaining sequences (13-59 and 14-06), the proposed scene invariant algorithm outperforms the other methods in terms of all error metrics. This is also observed when results are averaged across all frames from all sequences: the proposed algorithm outperforms the other methods, with a MAE of 2.328 and MRE of 11.49%.

Table 5
Results of various systems on the PETS 2009 dataset (sequences 13-57, 13-59 and 14-06). The average performance across all frames of these sequences is also reported. See text for a description of each algorithm. (Note that the results reported by Chan et al. (2009) for sequence 14-06 used a smaller ROI than the other sequences.)

Algorithm	13-57			13-59			14-06			All frames		
	MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE
Chan	2.308	8.362	–	1.647	4.087	–	4.328	44.159	–	2.680	17.661	–
Albiol	2.800	–	12.60%	3.860	–	24.90%	5.140	–	26.10%	3.895	–	21.16%
Conte	1.920	–	8.70%	2.240	–	17.30%	4.660	–	20.50%	2.867	–	15.40%
Kong, Linear	4.003	24.344	17.21%	1.746	5.223	12.68%	4.873	41.032	22.90%	3.446	22.453	17.29%
Kong, NN (8)	2.750	10.520	14.09%	2.088	7.665	14.49%	4.120	24.559	22.27%	2.925	13.738	16.72%
Kong, NN (16)	2.655	12.603	11.60%	3.055	12.806	21.74%	6.235	65.358	23.51%	3.886	28.671	18.89%
Kong, GPR	2.600	10.846	12.47%	2.515	10.078	16.58%	6.751	73.729	24.89%	3.828	29.631	17.73%
Local Features	**1.327**	**3.081**	**6.26%**	1.684	4.591	12.19%	4.963	41.449	20.89%	2.559	15.262	12.85%
Holistic Features	4.080	27.168	15.52%	1.678	4.599	11.76%	6.696	71.943	27.16%	4.000	32.539	17.68%
Scene Invariant	2.576	10.115	11.30%	**1.269**	**2.317**	**9.14%**	**3.326**	**15.776**	**14.50%**	**2.328**	**8.996**	**11.49%**

These results indicate that the proposed methodology performs competitively with existing methods when assessed on individual sequences of the PETS 2009 dataset, and outperforms other methods on two of these sequences (13-59, 14-06), despite being trained on different viewpoints. This is most likely due to the greater quantity and wider variety of the training data which is available from the other viewpoints, leading to improved generalisation.

6.2. Multi camera evaluation

In this section the multi camera crowd counting algorithm is evaluated on two-camera environments using the PETS 2006 and PETS 2009 datasets. The performance of these algorithms is evaluated against the definition of ground truth described in Section 5.1. The density map modification of Section 4.2 is referred to as the 'map' method, and the pixel density assignment of Section 4.3 is referred to as the 'pixel' method. We also compare to the 'naive' and 'cutoff' methods of Section 4.4.

Comparison to other algorithms is not possible because existing methods are scene-specific and do not operate across multi camera networks.

6.2.1. PETS 2006 results

The PETS 2006 database provides camera calibration for each of its viewpoints, and Views 3 and 4 were selected because they provide the best coverage of the scene with some overlap. Two sequences from these cameras, S1 and S5, were chosen because they contain the largest crowds.

A twofold cross validation procedure was used to assess performance on this database. The system was first trained on sequence S5 and tested on sequence S1, and then vice versa. Average results across all frames in both sequences are then reported. Results are displayed in Table 6 for all four algorithms.

The proposed algorithms operate with a mean absolute error of 0.446 and 0.388 for the map and pixel methods, respectively. The mean relative error does not exceed 20% for either algorithm. By comparison, the naive and cutoff methods operate with a mean absolute error of 0.678 and 0.722, respectively, and the MRE of these methods exceeds 20%.

The results are also plotted in Fig. 4(a). As expected, the naive method overestimates the crowd size when pedestrians pass through the overlap region. By contrast the cutoff methods tends to underestimate the crowd during this time. Both the map and pixel methods give suitable results.

6.2.2. PETS 2009 results

Camera calibration is provided for multiple viewpoints of the PETS 2009 database, and Views 1 and 2 were selected due to the wide coverage and overlap that they provide. Sequences 13-57 and 13-59 were selected for evaluation because these were specifically captured for the purpose of evaluating crowd counting algorithms. These datasets contain much larger crowds than the PETS 2006 database.

Cross validation was performed using the sequences 13-57 and 13-59, and the results of this analysis are shown in Table 6. The proposed algorithms achieve a mean absolute error of 1.173 and 1.697 for the map and pixel methods, respectively. By comparison the naive method reports a MAE of 6.635 due to severe overestimation of the crowd size due to overlap regions. The cutoff method achieves a MAE of 1.631 which is comparable to the pixel method. Any errors which may occur within the cutoff region (as pedestrians cross from the ROI of one viewpoint to another) are minimal compared to the overall crowd size. Hence the cutoff method performs relatively well on the PETS 2009 database compared to the PETS 2006 database. The best performance across all three error metrics is observed for the map method.

The results are plotted in Fig. 4(b). The naive method overestimates the crowd size severely, while good performance ($\text{MRE} < 10\%$) is observed for the other methods.

6.3. Scene invariant and multi camera evaluation

In this section, scene invariance is combined with multi camera crowd counting, and the proposed algorithm is evaluated on a three-camera environment using the PETS 2009 database.

Training is performed using the QUT datasets described in Section 5.2, obtained from Cameras 3, 5 and 8 of the QUT camera network. Feature normalisation is performed as described in Section 3.1 to achieve scene invariance. Testing is then performed on Views 1–3 of the PETS 2009 database, for sequences 13-57 and 13-59. The ROIs used for this evaluation are shown in Fig. 2.

Table 7 presents the results of this evaluation. The proposed map and pixel methods attain mean absolute error rates of 1.756 and 2.665, respectively. Both approaches achieved $\text{MRE} < 10\%$ indicating highly accurate performance. These approaches both outperform the cutoff method which achieved an MRE of 14.34%. This is due to the use of multiple ROI cutoffs within a three-camera environment, which provide imperfect separation between cameras: humans are 3D objects and therefore it is difficult to achieve true separation using hypothetical lines on the ground plane. The compensation provided by the overlap map (Section 4.1) appears to provide superior crowd counting performance in a complicated three-camera environment such as this.

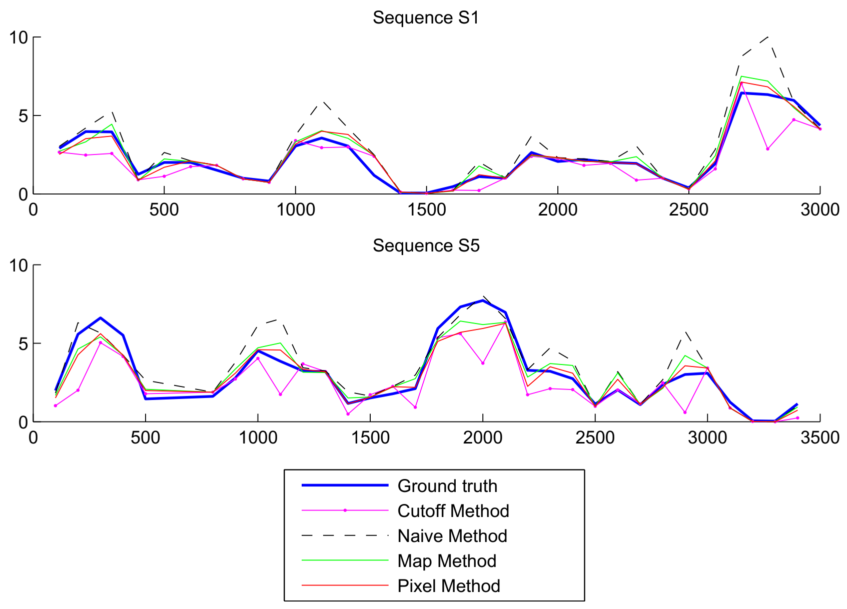
Results for this experiment are plotted in Fig. 5. These results provide strong support for both the scene invariant and the multi camera crowd counting algorithms proposed in this paper.

6.4. Processing speed

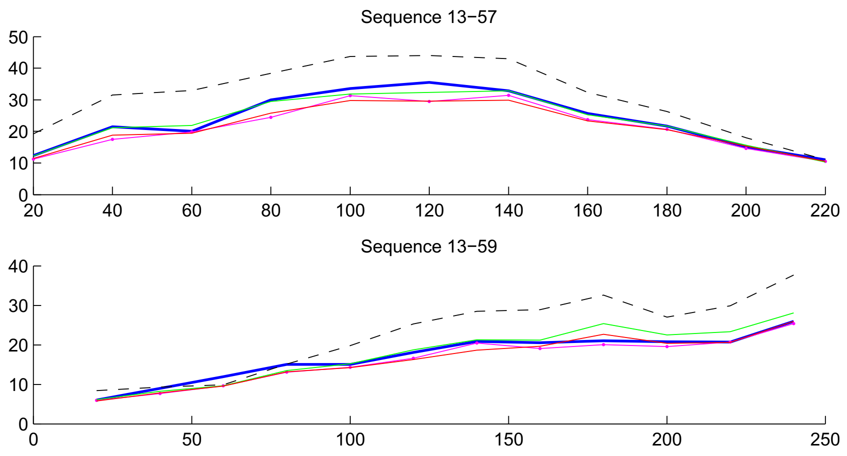
The processing speed of the proposed algorithm is reported in Table 8. These algorithms are implemented in C++ and the code

Table 6
Results of multi camera crowd counting using *Views 3 and 4* of the PETS 2006 dataset and *Views 1 and 2* of the PETS 2009 dataset.

Training	Testing	Map method			Pixel method			Naive method			Cutoff method		
		MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE
PETS 2006													
S5	S1	0.343	0.217	18.74%	0.269	0.145	14.34%	0.663	1.166	31.40%	0.545	0.780	26.23%
S1	S5	0.542	0.472	19.00%	0.500	0.472	16.03%	0.691	0.963	26.47%	0.888	1.739	28.82%
	All frames	0.446	0.349	18.88%	**0.388**	**0.314**	**15.21%**	0.678	1.061	28.86%	0.722	1.275	27.57%
PETS 2009													
13-59	13-57	0.879	1.715	3.65%	2.313	8.438	8.60%	7.378	66.919	31.51%	2.262	8.959	8.81%
13-57	13-59	1.443	3.526	8.27%	1.132	1.940	7.53%	5.953	50.568	32.00%	1.054	1.573	6.94%
	All frames	**1.173**	**2.660**	**6.06%**	1.697	5.048	8.04%	6.635	58.388	31.77%	1.631	5.105	7.84%



(a) PETS 2006



(b) PETS 2009

Fig. 4. Results of multi camera crowd counting using *Views 3 and 4* of the PETS 2006 dataset and *Views 1 and 2* of the PETS 2009 dataset.

has not been optimised for speed. In a single camera environment, all methods are equivalent and operate at 2.56 frames per second (fps). When extended to 2 cameras, the processing speed is approximately halved: the naive and cutoff methods operate at 1.28 fps, and the map method operates at 1.18 fps. The pixel method operates at only 0.78 fps due to the additional overhead in calculating crowding densities at each pixel (Eqs. (23)–(25)).

When extended to 3 cameras, processing speed is reduced by a factor of approximately 3 for the map, naive and cutoff methods (0.81–0.83 fps); and the pixel method is reduced to 0.54 fps.

The main bottleneck in the algorithm is foreground segmentation, which can be improved significantly using GPU acceleration as in Pham et al. (2010) for example. Furthermore, multi-threading programming can be used to process each camera in parallel,

Table 7
Results of multi camera crowd counting using *Views 1–3* of the PETS 2009 dataset.

Training	Testing	Map method			Pixel method			Naive method			Cutoff method		
		MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE	MAE	MSE	MRE
QUT	13-57	1.977	5.533	6.95%	2.732	8.952	8.89%	21.739	532.523	74.37%	5.207	44.034	24.63%
QUT	13-59	1.554	4.717	5.81%	2.603	7.878	10.25%	16.477	335.754	56.98%	1.236	2.262	4.91%
	All frames	**1.756**	**5.107**	**6.36%**	2.665	8.392	9.60%	18.993	429.861	65.30%	3.135	22.240	14.34%

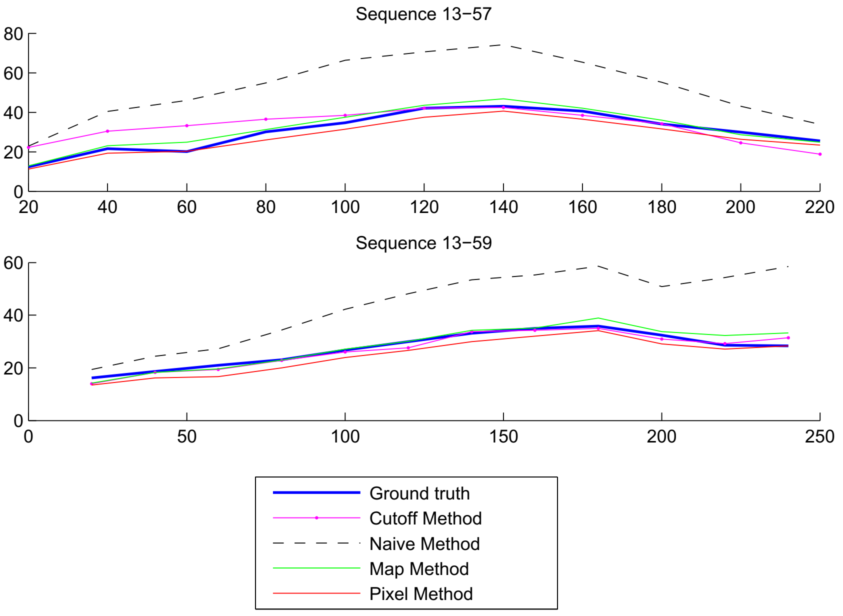


Fig. 5. Results of multi camera crowd counting using *Views 1–3* of the PETS 2009 dataset.

Table 8
Processing speed of the proposed algorithm using 1, 2 and 3 cameras from the PETS 2009 dataset.

Cameras	Frame rate (fps)			
	Map	Pixel	Naive	Cutoff
1			2.56	
2	1.18	0.78	1.28	1.28
3	0.81	0.54	0.82	0.83

eliminating much of the slow-down observed for the multi camera environments in Table 8.

7. Conclusion

This paper proposed a novel multi camera crowd counting algorithm built upon scene invariance and local image features. The proposed system utilises camera calibration to scale features between viewpoints based on a real world reference object, namely a human sized 3D cylinder model. This is used to calculate the density map, which is used to weight the features extracted at each pixel.

Scene invariance was demonstrated by training the system on a bank of multiple 'reference' cameras and then testing it on a new, unseen viewpoint. Accurate crowd counting results were obtained for the seven calibrated sequences. Additionally, this paper has contributed a detailed analysis of various feature types (size, shape, edges and keypoints) and investigated their suitability for scene invariant crowd counting. It was found that optimal performance was achieved when all features were used. A number of regression models (GPR, linear, KNN and NN) were also evaluated and it was observed that GPR provided superior predictive performance for our system.

Multi camera crowd counting was proposed by making use of camera calibration to compensate for regions of overlap. Two algorithms were proposed to count crowds in a multi camera environment. Density map modification (the map method) alters the density map in regions of overlap, which modifies the values of the image features prior to regression. By contrast, pixel density assignment (the pixel method) performs compensation on a pixel-wise basis after regression has been performed. Both approaches are based on the construction of an *overlap map* which enables a system to quantify the amount of overlap in any given region of an image.

Accurate crowd counting results were demonstrated using five calibrated cameras. The pixel method performed slightly better on the PETS 2006 database, while the map method performed better on the PETS 2009 database. Both algorithms outperform the naive approach and provide similar or better performance than the ROI 'cutoff' method (described in Section 4.4). Both approaches require some additional computational overhead compared to the naive and cutoff methods. However, the accuracy is improved by doing so.

Finally, the combination of scene invariance and multi camera crowd counting demonstrates the efficacy of the algorithms proposed in this paper. Three viewpoints from the QUT camera network were used to train the proposed system, and highly accurate crowd counting results were observed across Views 1–3 of the PETS 2009 database (with a mean relative error less than 10%).

Once trained, the proposed scene invariant crowd counting 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 'plug-and-play' system which is pre-trained on a large bank of data from a variety of cameras, and may be deployed across a multi camera environment. This technology has many potential applications, including automatic gathering of business intelligence, crowd safety monitoring and abnormality detection.

References

Abdel-Aziz, Y., Karara, H., 1971. 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.

Albiol, A., Silla, J., 2009. Statistical video analysis for crowds counting. In: 2009 16th IEEE International Conference on Image Processing (ICIP). pp. 2569–2572. http://dx.doi.org/10.1109/ICIP.2009.5414002.

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., 2008. Speeded-up robust features (SURF). Computer Vision and Image Understanding 110 (3), 346–359. http://dx.doi.org/10.1016/j.cviu.2007.09.014 http://www.sciencedirect.com/science/article/pii/S1077314207001555.

Bose, B., Grimson, E., 2004. Ground plane rectification by tracking moving objects. In: IEEE International Workshop on Visual Surveillance and PETS.

Celik, H., Hanjalic, A., Hendriks, E., 2006. Towards a robust solution to people counting. In: 2006 IEEE International Conference on Image Processing. pp. 2401–2404. http://dx.doi.org/10.1109/ICIP.2006.312946.

Chan, A., Vasconcelos, N., 2008. Modeling, clustering, and segmenting video with mixtures of dynamic textures. IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (5), 909–926. http://dx.doi.org/10.1109/TPAMI.2007.70738.

Chan, A., Vasconcelos, N., Bayesian poisson regression for crowd counting. In: 2009 IEEE 12th International Conference on Computer Vision. pp. 545–551. http://dx.doi.org/10.1109/ICCV.2009.5459191.

Chan, A., Liang, Z.-S., Vasconcelos, N., Privacy preserving crowd monitoring: counting people without people models or tracking. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2008. pp. 1–7. http://dx.doi.org/10.1109/CVPR.2008.4587569. Dataset: http://www.svcl.ucsd.edu/projects/peoplecnt/.

Chan, A.B., Morrow, M., Vasconcelos, N., 2009. Analysis of crowded scenes using holistic properties. In: Performance Evaluation of Tracking and Surveillance Workshop at CVPR 2009, Miami, Florida. pp. 101–108.

Conte, D., Foggia, P., Percannella, G., Tufano, F., Vento, M., 2010. A method for counting moving people in video surveillance videos. EURASIP Journal on Advances in Signal Processing 2010 (1), 231240. http://dx.doi.org/10.1155/2010/231240 http://asp.eurasipjournals.com/content/2010/1/231240.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, vol. 1. pp. 886–893. http://dx.doi.org/10.1109/CVPR.2005.177.

Davies, A.C., Yin, J.H., Velastin, S.A., 1995. Crowd monitoring using image processing. Electronics & Communication Engineering Journal 7 (1), 37–47. http://dx.doi.org/10.1049/ecej:19950106.

Denman, S.P., 2009. Improved detection and tracking of objects in surveillance video. Ph.D. thesis. http://eprints.qut.edu.au/29328/.

Denman, S., Chandran, V., Sridharan, S., 2007. An adaptive optical flow technique for person tracking systems. Pattern Recognition Letters 28 (10), 1232–1239. http://dx.doi.org/10.1016/j.patrec.2007.02.008 http://www.sciencedirect.com/science/article/pii/S0167865507000591.

Dong, L., Parameswaran, V., Ramesh, V., Zoghlami, I., 2007. Fast crowd segmentation using shape indexing. In: 2007 IEEE 11th International Conference on Computer Vision, ICCV 2007. pp. 1–8. http://dx.doi.org/10.1109/ICCV.2007.4409075.

Denman, S., Fookes, C., Sridharan, S., 2009. Improved simultaneous computation of motion detection and optical flow for object tracking. In: Digital Image Computing: Techniques and Applications, DICTA '09. pp. 175–182. http://dx.doi.org/10.1109/DICTA.2009.35.

Harris, C., 1988. A combined corner and edge detector. In: Proceedings of the Alvey Vision Conference. http://ci.nii.ac.jp/naid/10015313645/.

Kilambi, P., Ribnick, E., Joshi, A.J., Masoud, O., Papanikolopoulos, N., 2008. Estimating pedestrian counts in groups. Computer Vision and Image Understanding 110 (1), 43–59. http://dx.doi.org/10.1016/j.cviu.2007.02.003 http://www.sciencedirect.com/science/article/pii/S1077314207000392.

Kong, D., Gray, D., Tao, H., 2006. A viewpoint invariant approach for crowd counting. In: 18th International Conference on Pattern Recognition, 2006. ICPR 2006, vol. 3. pp. 1187–1190. http://dx.doi.org/10.1109/ICPR.2006.197.

Krahnstoever, N., Mendonca, P.R.S., 2005. Bayesian autocalibration for surveillance. In: ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision. IEEE Computer Society, Washington, DC, USA, pp. 1858–1865. http://dx.doi.org/10.1109/ICCV.2005.44.

Lempitsky, V., Zisserman, A., 2010. Learning to count objects in images. In: Advances in Neural Information Processing Systems (NIPS). http://eprints.pascal-network.org/archive/00007096/.

Lin, S.-F., Chen, J.-Y., Chao, H.-X., 2001. Estimation of number of people in crowded scenes using perspective transformation. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 31 (6), 645–654. http://dx.doi.org/10.1109/3468.983420.

Lv, F., Zhao, T., Nevatia, R., 2002. Self-calibration of a camera from video of a walking human. ICPR '02: Proceedings of the 16th International Conference on Pattern Recognition (ICPR'02), vol. 1. IEEE Computer Society, Washington, DC, USA, p. 10562.

Ma, R., Li, L., Huang, W., Tian, Q., 2004. On pixel count based crowd density estimation for visual surveillance. In: 2004 IEEE Conference on Cybernetics and Intelligent Systems, vol. 1. pp. 170–173. http://dx.doi.org/10.1109/ICCIS.2004.1460406.

Marana, A.N., Velastin, S.A., Costa, L.F., Lotufo, R.A., 1997. Estimation of crowd density using image processing. IEE Colloquium on Image Processing for Security Applications (Digest No.: 1997/074). IET, pp. 11/1–11/8. http://dx.doi.org/10.1049/ic:19970387.

Marana, A., Da Fontoura Costa, L., Lotufo, R., Velastin, S., 1999. Estimating crowd density with minkowski fractal dimension. In: Proceedings of the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 6. pp. 3521–3524. http://dx.doi.org/10.1109/ICASSP.1999.757602.

Paragios, N., Ramesh, V., 2001. A MRF-based approach for real-time subway monitoring. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001, vol. 1. IEEE, pp. I-1034–I-1040. http://dx.doi.org/10.1109/CVPR.2001.990644.

PETS, 2006. Ninth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. http://www.cvg.rdg.ac.uk/PETS2006/.

PETS, 2009. Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. http://www.cvg.rdg.ac.uk/PETS2009/.

PETS, 2012. Fourteenth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. http://www.cvg.rdg.ac.uk/PETS2012/.

Pham, V., Vo, P., Hung, V.T., Bac, L.H., 2010. GPU implementation of extended gaussian mixture model for background subtraction. In: 2010 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF). pp. 1–4. http://dx.doi.org/10.1109/RIVF.2010.5634007. http://code.google.com/p/cubgs/.

Rahmalan, H., Nixon, M., Carter, J., 2006. On crowd density estimation for surveillance. In: The Institution of Engineering and Technology Conference on Crime and Security. pp. 540–545.

Rasmussen, C.E., 2004. Gaussian processes in machine learning. In: Bousquet, O., Luxburg, U.v., Rätsch, G. (Eds.), Advanced Lectures on Machine Learning, Lecture Notes in Computer Science, vol. 3176. Springer, Berlin, Heidelberg, pp. 63–71 http://link.springer.com/chapter/10.1007/978-3-540-2865-0_94.

Rasmussen, C.E., Williams, C.K.I., 2006. Gaussian Processes for Machine Learning. MIT Press.

Regazzoni, C.S., Tesei, A., Murino, V., 1993. A real-time vision system for crowding monitoring. International Conference on Industrial Electronics, Control, and Instrumentation. Proceedings of the IECON '93, vol. 3. IEEE, pp. 1860–1864. http://dx.doi.org/10.1109/IECON.1993.339357.

Rosten, E., Porter, R., Drummond, T., 2010. Faster and better: a machine learning approach to corner detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (1), 105–119. http://dx.doi.org/10.1109/TPAMI.2008.275.

Ryan, D., Denman, S., Fookes, C., Sridharan, S., 2009. Crowd counting using multiple local features. In: Digital Image Computing: Techniques and Applications, 2009. DICTA '09. pp. 81–88. http://dx.doi.org/10.1109/DICTA.2009.22.

Ryan, D., Denman, S., Fookes, C., Sridharan, S., 2010. Crowd counting using group tracking and local features. In: 2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). pp. 218–224. http://dx.doi.org/10.1109/AVSS.2010.30.

Ryan, D., Denman, S., Sridharan, S., Fookes, C., 2011. Scene invariant crowd counting. In: 2011 International Conference on Digital Image Computing Techniques and Applications (DICTA). pp. 237–242. http://dx.doi.org/10.1109/DICTA.2011.46.

Ryan, D., Denman, S., Sridharan, S., Fookes, C., 2012. Scene invariant crowd counting and crowd occupancy analysis. In: Video Analytics for Business Intelligence. Springer-Verlag, pp. 161–198.

Stauffer, C., Grimson, W., 1999. Adaptive background mixture models for real-time tracking. In: 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2. p. 2246. http://dx.doi.org/10.1109/CVPR.1999.784637.

Tsai, R., 1986. An efficient and accurate camera calibration technique for 3D machine vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 86). pp. 364–374. http://ci.nii.ac.jp/naid/10012692972/en/.

Tsai, R., 1987. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. IEEE Journal of Robotics and Automation 3 (4), 323–344. http://dx.doi.org/10.1109/JRA.1987.1087109.

Zhang, Z., 2000. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (11), 1330–1334. http://dx.doi.org/10.1109/34.888718.