2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance

Crowd Counting Using Group Tracking and Local Features

David Ryan, Simon Denman, Clinton Fookes, Sridha Sridharan
Image and Video Laboratory, Queensland University of Technology
GPO Box 2434, Brisbane 4001, Australia
{david.ryan, s.denman, c.fookes, s.sridharan}@qut.edu.au

Abstract

In public venues, crowd size is a key indicator of crowd safety and stability. In this paper we propose a crowd counting algorithm that uses tracking and local features to count the number of people in each group as represented by a foreground blob segment, so that the total crowd estimate is the sum of the group sizes. Tracking is employed to improve the robustness of the estimate, by analysing the history of each group, including splitting and merging events. A simplified ground truth annotation strategy results in an approach with minimal setup requirements that is highly accurate.

1. Introduction

In public places, crowd size may be an indicator of congestion, delay, abnormality or instability. Most crowd counting algorithms have utilised holistic image features to estimate crowd size [6, 7, 14]. However, due to the wide variability in crowd behaviours, distribution, density and overall size, holistic approaches require a broad set of training data and relatively complicated counting strategies. In a facility containing numerous cameras, it is not practical to hand-annotate hundreds of frames of ground truth for each viewpoint. Therefore accurate surveillance systems requiring minimal setup are highly desirable.

In this paper we utilise the local features of individuals and groups within an image, as proposed by Ryan [18]. While existing techniques have used features such as foreground pixels and edges [11, 4], 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 at the blob level, training data must also be annotated with local information. An improved annotation method compared to the baseline [18] is proposed, which greatly reduces the setup requirements.

Tracking of groups is performed enabling the system to use historic information to improve the group size estimates. Detection of splits and merges provide additional insight, as the number of pedestrians represented by a merged blob is expected to be the sum of its constituents, and the reverse is true for a split.

The proposed system is tested on a 2000 frame database featuring crowds of size 11-45 people (Figure 1). Results indicate that the proposed approach can achieve accurate results with as few as 20 frames of training data.

The remainder of the paper is structured as follows: Section 2 provides an overview of existing crowd counting techniques, Section 3 describes an overview of the proposed crowd counting system, Section 4 proposes an improved method for calculating localised ground truth, Section 5 explains how the tracking algorithm is used to improve crowd counting estimates, Section 6 presents experimental results and Section 7 discusses conclusions and directions for future work.

2. Existing Work

Crowd size is a holistic description of a scene, and therefore existing crowd size monitoring algorithms have typically utilised holistic image features. Examples of these include textural information [14], Minkowski Fractal Dimension [13], and Translation Invariant Orthonormal Chebyshev Moments [17]. Holistic features such as these are sensitive to external changes (such as lighting conditions), and it has been shown that for outdoor environments, the natural fluctuations in lighting between morning and afternoon reduce system performance [17].

Other crowd counting algorithms have utilised specific features which are indicative of crowding, such as edge and foreground pixels. While these features are local to points of interest in an image, they are considered at a holistic level. Many techniques such as [7, 16, 11] have used foreground segmentation to determine the crowd count. The relationship between the total number of foreground pixels and the number of people in the scene has been shown to be approximately linear [7]. However, local nonlinearities arise due to the effects of perspective and occlusion.

Paragios [16] proposed the use a geometric factor to weight each pixel according to its location on the ground

978-0-7695-4264-5/10 $26.00 © 2010 IEEE
DOI 10.1109/AVSS.2010.30

(a) Frame 1280.

(b) Foreground mask.

(c) Region of interest.

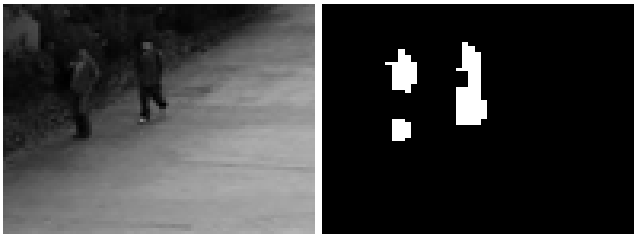
Figure 1. A frame from the UCSD pedestrian database [4].

plane, to overcome the problem of perspective. Occlusions have been addressed using blob size histograms [11] to better capture the range of blob sizes present in an image, and to enable the classifier to distinguish between groups of people and individuals. Chan [4] extract features in a greater quantity, although this also increases the quantity of training data required.

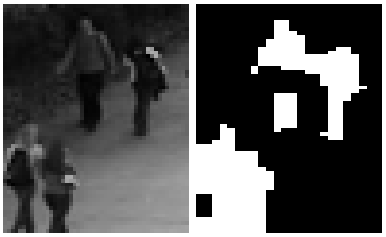
Local features are specific to an individual or small group of people within an image. For example, head detection has been proposed to estimate crowd sizes [12]. Individual pedestrian tracking [15] and blob segmentation [19] have been employed, however these approaches are best suited to situations where crowds are small. Celik [3] assumed linearity between blob size and a group's count, and Kilambi [10] used an elliptical cylinder to model groups of pedestrians as they are tracked through a scene. These assumptions may be restrictive in highly crowded and occluded environments, such as that depicted in Figure 1.

Ryan [18] utilised local blob features to estimate group sizes, however this approach requires annotating ground truth on the blob level after foreground segmentation has been performed. This process would be tedious when numerous blobs appear in an image, or when blobs are fragmented due to segmentation error (Figure 2) requiring fractional ground truth assignments.

Local features have been employed to other crowd related problems though, such as tracking [2] and analysis of



(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.

Figure 2. Typical errors in foreground extraction of low quality images.

Feature	Description
Area	The weighted pixel count for the blob, B: $B_{size} = \sum_{(i,j)\in B} W(i,j)$
Perimeter	The weighted pixel count for the blob's perimeter, P: $P_{size} = \sum_{(i,j)\in P} \sqrt{W(i,j)}$
Perimeter-Area Ratio	A measure of shape complexity [4], corresponding to crowding: $R = P_{size}/B_{size}$
Edges	The weighted pixel count for the set of Canny edges within the blob, E: $E_{size} = \sum_{(i,j)\in E} \sqrt{W(i,j)}$
Edge Angle Histogram	Canny edge angles are quantised into 6 histogram bins in the range 0°-180° [11]. Each pixel's contribution to a histogram bin is $\sqrt{W(i,j)}$.

Table 1. Features extracted from each blob [18].

crowd stability [1]. However neither of these algorithms are concerned with the overall size of the crowd.

3. System Description

In this paper we propose a number of extensions to the work of Ryan [18]. This section describes the baseline algorithm (Section 3.1) and a summary of our contributions (Section 3.2).

3.1. Baseline Algorithm

Ryan [18] proposes a crowd counting algorithm that extracts local features from each blob in an image, obtained using a foreground segmentation algorithm [8]. The features extracted from each blob are listed in Table 1. A density map is calculated as in [4], weighting each pixel (i, j) by $W(i, j)$ to compensate for perspective.

The group size estimate E_n for the nth blob is calculated from its extracted feature set $\{f_i\}$ using a least-squares linear model (as in [11]):

$$E_n = w_0 + \sum_i w_i f_i \tag{1}$$

The total crowd estimate is the sum of the individual group sizes, $E = \sum_n E_n$.

The linear model of (1) can be replaced by any regression model or classifier. Because classification is performed on the local level, ground truth is also annotated on the local level. This is necessary to train the system. It requires manual identification of the number of pedestrians in each blob, using a training data set.

3.2. Summary of Contributions

The main contributions of this paper are: (1) A simplified approach to ground truth annotation in which pedestrian counts are automatically assigned to blobs, greatly reducing system setup time (Section 4); (2) The tracking of individual blobs through splits and merges to improve the estimated count for each blob by using historic information, resulting in improved accuracy (Section 5).

4. Improved Ground Truth Annotation

The approach of Ryan [18] requires localised ground truth annotation of blobs. 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 2).

An attempt to allocate fractional values in proportion to blob sizes is described in [18], however this is still a tedious manual process with room for ambiguity, subjectivity and inconsistency. Further, the annotation is performed after the segmentation stage; if this segmentation algorithm is modified at a later date, the former annotations may no longer be applicable. Thus the manual annotations would need to be performed again with the new segmentation results in order to re-train the system.

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 (x, y) coordinates of each person in the scene. The localised blob annotation can then be performed automatically in an unsupervised and consistent manner.

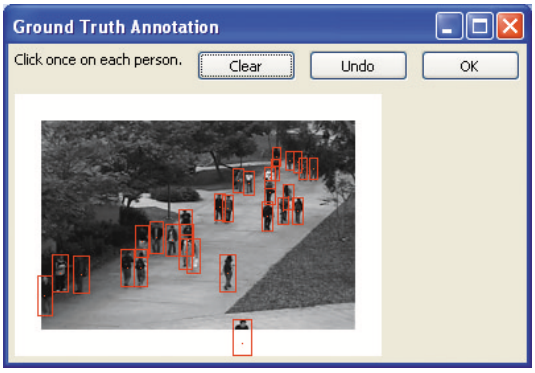


Figure 3. GUI used for the proposed ground truth annotation method. The user clicks on each person; bounding boxes are approximated using a simple calibration technique.

Notation	Description
M	Mask of scene (region of interest/ROI).
F	Foreground pixels detected using an adaptive background model [8].
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	Rectangular bounding box 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 2. Various regions in an image. Regions are treated as sets of pixels, and set notation is used.

During the proposed ground truth process, the approximate centre of each person in the scene is annotated. The size of a person is modeled as a rectangle whose width and height vary linearly according to their y coordinate in the image plane (as in [4]). This is used to approximate each person's bounding box. An example of the annotation process can be seen in Figure 3. An established tool such as ViPER [9] is also suitable for this purpose.

The localised blob annotation is then performed automatically by considering the overlap between foreground blobs and the pedestrian bounding boxes. In the case of large groups, multiple bounding boxes will overlap the same blob. On the other hand, when blob fragmentation occurs, multiple blobs will overlap a single bounding box.

Using set notation, we define a number of regions as sets of pixels in Table 2. From this we calculate the following values:

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

$$Q_i = \frac{|M \cap R_i|}{R_i} \tag{2}$$

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

$$C_{in} = \frac{|R_i \cap B_n|}{|R_i \cap B|} \times Q_i \tag{3}$$

- T_n: the total number of people represented by blob n. This is given by the sum of 'contributions' from pedestrians:

$$T_n = \sum_i C_{in} \tag{4}$$

Thus $\{T_n\}$ are the target counts for the blobs in the scene. These annotations will be similar to the hand-annotated blobs of [18], however they are computed automatically from the (x, y) coordinates. This simplifies the annotation process (as the user merely need to click on each person in a GUI); and seperates the annotation stage from the segmentation stage.

As our measure of holistic ground truth, we calculate the total quantity of pedestrians in the scene:

$$Q = \sum_i Q_i \tag{5}$$

This may be temporarily fractional as pedestrians enter or exit the scene's boundary. This reduces the ambiguity inherent in classifying a pedestrian as either 'in' or 'out' of a region, when in reality the transition is gradual.

5. Improved Counting Using Tracking

Crowd counting algorithms have typically treated each frame of video as independent of one another, estimating the crowd size based on the features extracted from that frame. Although a temporal smoothing may be applied to the holistic count to reduce outliers, in this work we propose the use of blob-level tracking to improve each group's estimate.

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 [15], however in this section we describe a 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.

5.1. Direct Match

The first step in comparing consecutive frames is to detect direct matches between overlapping blobs. Given two regions A, B, let $O(A, B)$ denote the fraction of region A that overlaps B:

$$O(A, B) = \frac{|A \cap B|}{|A|} \tag{6}$$

Let $B_{t,m}$ denote the mth blob in frame t. Between frame t and $t + 1$, blobs are compared in order to attempt the matching of blob $B_{t,m}$ to blob $B_{t+1,n}$. This is done by calculating two fractional overlaps: forward overlap $F_t(m, n)$ and reverse overlap $R_t(m, n)$.

$$F_t(m, n) = O(B_{t,m}, B_{t+1,n}) \tag{7}$$

$$R_t(m, n) = O(B_{t+1,n}, B_{t,m}) \tag{8}$$

To match blobs $B_{t,m}$ and $B_{t+1,n}$ it is necessary to ensure sufficient overlap:

$$F_t(m, n) \geq T_{min} \tag{9}$$

$$R_t(m, n) \geq T_{min} \tag{10}$$

And to distinguish a match from split or merge events, this overlap should be mostly exclusive to $B_{t,m}$ and $B_{t+1,n}$. Therefore the following requirements are also enforced:

$$F_t(i, n) < T_{max} \quad \forall i \neq m \tag{11}$$

$$R_t(i, n) < T_{max} \quad \forall i \neq m \tag{12}$$

$$F_t(m, j) < T_{max} \quad \forall j \neq n \tag{13}$$

$$R_t(m, j) < T_{max} \quad \forall j \neq n \tag{14}$$

Any blob pair (m, n) which satisfies conditions 9-14 is deemed a match. The threshold values T_{min} and T_{max} serve only to filter out erroneous matches, and their values are not particularly important. The values $T_{min} = 0.5$ and $T_{max} = 0.2$ were selected for our experiments.

5.2. Merging and Splitting

After direct matches have been determined, the matched blobs are removed from consideration.

The second step in our approach detects merges and splits by combining remaining blobs into pairs and attempting to match them collectively to a single blob in the other frame.

5.2.1 Merging

To match blobs $B_{t,p}$ and $B_{t,q}$ to the merged blob $B_{t+1,r}$, we combine them to form the joined region:

$$J_{t,p,q} = B_{t,p} \cup B_{t,q} \tag{15}$$

and attempt to match it to $B_{t+1,r}$ using the matching procedure described in Section 5.1.

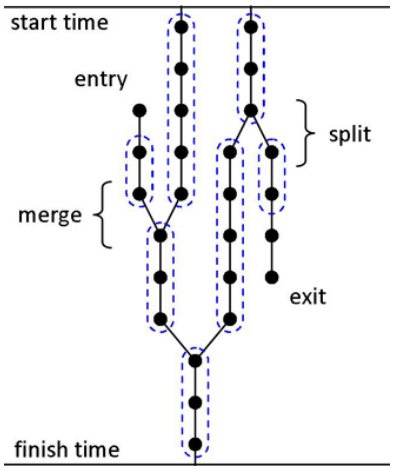


Figure 4. Visualisation of blob tracking results. Groups of constant size are circled.

5.2.2 Splitting

To match blob $B_{t,p}$ to the split blobs $B_{t+1,q}$ and $B_{t+1,r}$, we combine them to form the joined region:

$$J_{t+1,q,r} = B_{t+1,q} \cup B_{t+1,r} \tag{16}$$

and attempt to match it to $B_{t,p}$ using the matching procedure described in Section 5.1.

5.3. Improved Counting

This section describes the procedure used to improve the counting estimate by taking advantage of the tracking results. The splitting and merging of blobs may be visualised using a graph structure such as Figure 4.

In this approach we assume that directly-matched blobs represent a constant number of people, while a merged blob represents the sum of its constituent blobs' group sizes.

The exceptions to this rule are blobs entering or exiting while touching the scene's perimeter; these blobs are classified independently using the baseline method described in Section 3.1. The subsequent discussion concerns blobs fully contained within the region of interest.

Let $B_{t,n}$ denote blob n in frame t, and $E_{t,n}$ the estimated group size using the baseline method. We seek to calculate an improved estimate, $\hat{E}_{t,n}$, by incorporating the tracking results. The following three counting strategies are proposed.

5.3.1 Adaptive Updating

An adaptive learning rate is used to prevent rapid fluctuations due to outliers in a group's estimate over time. The estimate is calculated as follows:

No match When the blob $B_{t,n}$ cannot be matched to any blob in the previous frame, the direct estimate $E_{t,n}$ is used:

$$\hat{E}_{t,n} = E_{t,n} \tag{17}$$

Direct Match When the blob $B_{t,n}$ is matched to $B_{t-1,m}$, we update the estimate with a learning rate α:

$$\hat{E}_{t,n} = \alpha E_{t,n} + (1-\alpha)\hat{E}_{t-1,m} \tag{18}$$

Merging When the blob $B_{t,n}$ is the result of merging $B_{t-1,p}$ and $B_{t-1,q}$, we update:

$$\hat{E}_{t,n} = \alpha E_{t,n} + (1-\alpha)(\hat{E}_{t-1,p} + \hat{E}_{t-1,q}) \tag{19}$$

Splitting When the blobs $B_{t,m}$ and $B_{t,n}$ are the result of splitting $B_{t-1,p}$, we calculate the initial estimates:

$$I_{t,m} = \frac{E_{t,m}}{E_{t,m} + E_{t,n}} \hat{E}_{t-1,p} \tag{20}$$

$$I_{t,n} = \frac{E_{t,n}}{E_{t,m} + E_{t,n}} \hat{E}_{t-1,p} \tag{21}$$

and then update them as follows:

$$\hat{E}_{t,m} = \alpha E_{t,m} + (1-\alpha) I_{t,m} \tag{22}$$

$$\hat{E}_{t,n} = \alpha E_{t,n} + (1-\alpha) I_{t,n} \tag{23}$$

5.3.2 Mean Value

In this strategy, each blob $B_{t,n}$ retains a historical list of estimates. The improved estimate, $\hat{E}_{t,n}$, is taken to be the mean value of this history. The list is calculated as follows:

No match When the blob $B_{t,n}$ cannot be matched to any blob in the previous frame, the historical list for blob $B_{t,n}$ contains only the current estimate, $E_{t,n}$.

Direct Match When the blob $B_{t,n}$ is matched to $B_{t-1,m}$, it adopts the same historical list and appends to it the current estimate, $E_{t,n}$.

Merging When two blobs merge, each contains a historical list of estimates. A new list is formed by summing their corresponding elements. The new list's length is the shorter of the two being merged. The merged blob adopts this list and appends to it the current estimate, $E_{t,n}$.

Splitting When the blobs $B_{t,m}$ and $B_{t,n}$ are the result of a split, we calculate the initial estimates $I_{t,m}$ and $I_{t,n}$ as defined in equations 20 and 21. These are adopted as the first elements in the historical list for $B_{t,m}$ and $B_{t,n}$, respectively. The current estimates, $E_{t,m}$ and $E_{t,n}$, are then appended to their respective lists.

5.3.3 Median Value

In this strategy, each blob $B_{t,n}$ retains a historical list of estimates, as described in Section 5.3.2. The improved estimate, $\hat{E}_{t,n}$, is taken to be the median value of this history.

6. Experimental Results

Testing was performed on frames 1-2000 of the UCSD pedestrian database from Chan [4]. This footage contains pedestrian traffic moving in two directions, with crowds of size 11-45 people. The video has been downsampled to 238×158 pixels and 10 fps, grayscale. An example frame is shown in Figure 1.

Training was performed on twenty frames selected over a six minute window (frames 2200, 2400, ..., 6000) using the ground truth annotation process described in Section 4. In order to compare the proposed algorithm against the baseline, the blobs in these frames were also manually annotated using the system of Ryan [18]. These annotations were somewhat subjective, particularly in the case of partial blob fading, fragmentation, and pedestrians entering or exiting the scene's perimeter.

Three measures were used to assess system performance: mean absolute error, mean square error (MSE), and mean difference (bias) between the system's estimate and the holistic ground truth. Counting results are presented in Table 3.

The system described as "Proposed, no tracking" employs the simplified ground truth annotation method of Section 4, but not the tracking scheme of Section 5. This system achieves a high level of accuracy, demonstrating the validity of the proposed annotation strategy.

The tracking scheme of Section 5 further improves the estimate. It can be observed from Table 3 that accuracy improves with decreasing learning rate α, as this corresponds to an increasingly stronger smoothing effect on a group's estimate over time. Our experiments indicate that the median value strategy of Section 5.3.3 is most accurate, with a mean square error of 2.36 (against the baseline MSE of 2.75). This configuration is best suited to reject outlier estimates, as is evident in the 'Bias' column of Table 3.

We also present the results for this database reported by Chan [4] in Table 4. Chan counts the number of pedestrians moving in each direction using a mixture of dynamic textures segmentation algorithm [5], so the results cannot be directly compared. Nevertheless Chan reports a mean square error of 1.291 for pedestrians moving toward the camera, and 4.181 for crowds moving away from the camera.

An advantage of Chan's technique is that it can count crowds moving in either direction, due to the bidirectional segmentation algorithm. However, this approach can only segment moving pedestrians, and not those who have stopped in the middle of the scene. Many surveillance settings involve stationary subjects, and this can even be caused by excessive congestion, which is what we seek to detect.

Direction	Error	MSE
Away	1.621	4.181
Towards	0.869	1.291

Table 4. Results presented in Chan [4]. The training frames were 600-1399, and the testing frames were 1-599 and 1400-2000.

We also note that Chan utilised 600 frames of training data which were annotated with the number of pedestrians moving in each direction. This would be a burdensome task to perform for multiple viewpoints in a large facility where crowd counting was required. The results presented here were obtained using 20 frames of training data, which is a more practical setup requirement, and does not compromise accuracy.

7. Conclusions

In this paper we have proposed the use of group-level tracking and local features for crowd counting. Results presented in Section 6 indicate that counting strategies which are robust against outliers, such as the median value (Section 5.3.3), are most accurate.

An improved annotation strategy simplifies the training process, so that the proposed system can operate on a minimal training set of 20 frames. This is a highly practical setup requirement when configuring a large number of crowd counting systems in a multi-camera environment.

The system is limited by the simple model of Equation 1. This can be replaced with a more accurate classifier or regression model without altering the surrounding framework.

Future work will investigate these techniques for counting crowds in multi-camera environments, and detecting local abnormalities in crowd density across a scene.

References

[1] S. Ali and M. Shah. A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–6, 2007. 2

[2] S. Ali and M. Shah. Floor fields for tracking in high density crowd scenes. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 1–14, Berlin, Heidelberg, 2008. Springer-Verlag. 2

[3] H. Celik, A. Hanjalic, and E. Hendriks. Towards a robust solution to people counting. *Image Processing, 2006 IEEE International Conference on*, pages 2401–2404, Oct. 2006. 2

System	Error	MSE	Bias
Baseline - Ryan [18]	1.2991	2.7470	-0.3322
Proposed, no tracking	1.2586	2.5504	-0.2688
Proposed, adaptive update ($\alpha = 0.5$)	1.2408	2.4872	-0.2645
Proposed, adaptive update ($\alpha = 0.25$)	1.2307	2.4610	-0.2570
Proposed, adaptive update ($\alpha = 0.1$)	1.2224	2.4402	-0.2394
Proposed, mean value	1.2245	2.4078	-0.2491
Proposed, median value	**1.2212**	**2.3586**	**-0.1499**

Table 3. Test results for both the baseline system and the proposed counting algorithm. The training frames were 2200, 2400, ..., 6000; and the testing frames were 1-2000.

[4] A. Chan, Z.-S. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7, June 2008. 1, 2, 3, 6

[5] A. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(5):909–926, May 2008. 6

[6] A. B. Chan, M. Morrow, and N. Vasconcelos. Analysis of crowded scenes using holistic properties. In *Performance Evaluation of Tracking and Surveillance workshop at CVPR 2009*, pages 101–108, Miami, Florida, 2009. 1

[7] A. Davies, J. H. Yin, and S. Velastin. Crowd monitoring using image processing. *Electronics & Communication Engineering Journal*, 7(1):37–47, Feb 1995. 1

[8] S. Denman, V. Chandran, and S. Sridharan. An adaptive optical flow technique for person tracking systems. *Pattern Recognition Letters*, 28(10):1232 – 1239, 2007. 3

[9] D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 4, pages 167 –170 vol.4, 2000. 3

[10] P. Kilambi, E. Ribnick, A. J. Joshi, O. Masoud, and N. Papanikolopoulos. Estimating pedestrian counts in groups. *Computer Vision and Image Understanding*, 110(1):43 – 59, 2008. 2

[11] D. Kong, D. Gray, and H. Tao. A viewpoint invariant approach for crowd counting. *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, 3:1187–1190, 2006. 1, 2, 3

[12] S.-F. Lin, J.-Y. Chen, and H.-X. Chao. Estimation of number of people in crowded scenes using perspective transformation. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 31(6):645–654, Nov 2001. 2

[13] A. Marana, L. Da Fontoura Costa, R. Lotufo, and S. Velastin. 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, Mar 1999. 1

[14] A. Marana, S. Velastin, L. Costa, and R. Lotufo. Estimation of crowd density using image processing. *Image Processing for Security Applications (Digest No.: 1997/074), IEE Colloquium on*, pages 11/1–11/8, Mar 1997. 1

[15] O. Masoud and N. Papanikolopoulos. A novel method for tracking and counting pedestrians in real-time using a single camera. *Vehicular Technology, IEEE Transactions on*, 50(5):1267–1278, Sep 2001. 2, 4

[16] N. Paragios and V. Ramesh. A mrf-based approach for real-time subway monitoring. In *2001 Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, pages 1034–1040, Dec. 2001. 1

[17] H. Rahmalan, M. Nixon, and J. Carter. On crowd density estimation for surveillance. *Crime and Security, 2006. The Institution of Engineering and Technology Conference on*, pages 540–545, June 2006. 1

[18] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Crowd counting using multiple local features. In *Digital Image Computing: Techniques and Applications, 2009. DICTA '09.*, pages 81 –88, Dec. 2009. 1, 2, 3, 4, 6, 7

[19] T. Zhao and R. Nevatia. Bayesian human segmentation in crowded situations. *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, 2:II–459–66 vol.2, June 2003. 2