Large Scale Monitoring of Crowds and Building Utilisation: A New Database and Distributed Approach

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

Abstract

Public buildings and large infrastructure are typically monitored by tens or hundreds of cameras, all capturing different physical spaces and observing different types of interactions and behaviours. However to date, in large part due to limited data availability, crowd monitoring and operational surveillance research has focused on single camera scenarios which are not representative of real-world applications. In this paper we present a new, publicly available database for large scale crowd surveillance. Footage from 12 cameras for a full work day covering the main floor of a busy university campus building, including an internal and external foyer, elevator foyers, and the main external approach are provided; alongside annotation for crowd counting (single or multi-camera) and pedestrian flow analysis for 10 and 6 sites respectively. We describe how this large dataset can be used to perform distributed monitoring of building utilisation, and demonstrate the potential of this dataset to understand and learn the relationship between different areas of a building.

1. Introduction

A significant amount of recent research has focussed on monitoring crowds, including tasks such as measuring crowd size [5, 12] and monitoring pedestrian flow [4, 6, 7] (i.e. how many people pass through a doorway). To date this research has largely focussed on single camera scenarios. However, large buildings such as those on a university campus or complex infrastructure such as airports are covered by tens or even hundreds of cameras, and often even individual areas within those buildings (i.e. a foyer) are too large to adequately cover with a single camera. Furthermore, when considering the operations and crowd levels within an entire building the relationship between the different elements of the building needs to be considered. A crowd entering through the main door will subsequently disperse throughout the building, and by observing such movements across multiple cameras over time it may be possible to learn how this dispersal takes place, allowing estimates of crowd movements to be improved. However the lack of a large multi-camera dataset that captures the crowd interactions in and around a building has, to date, limited researchers ability to investigate these problems.

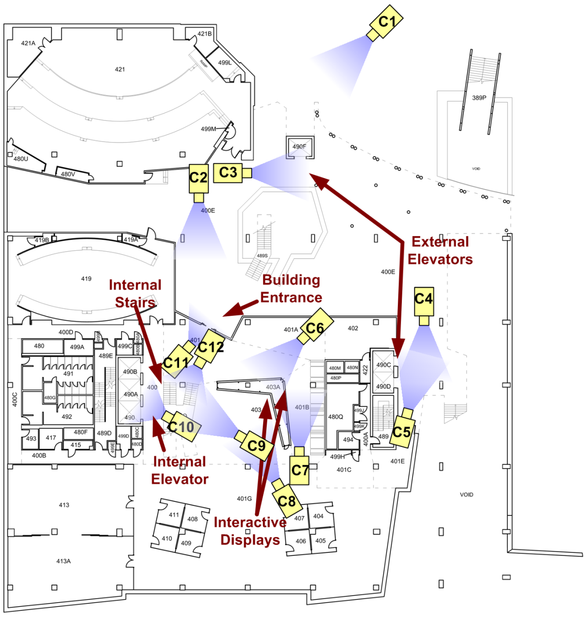
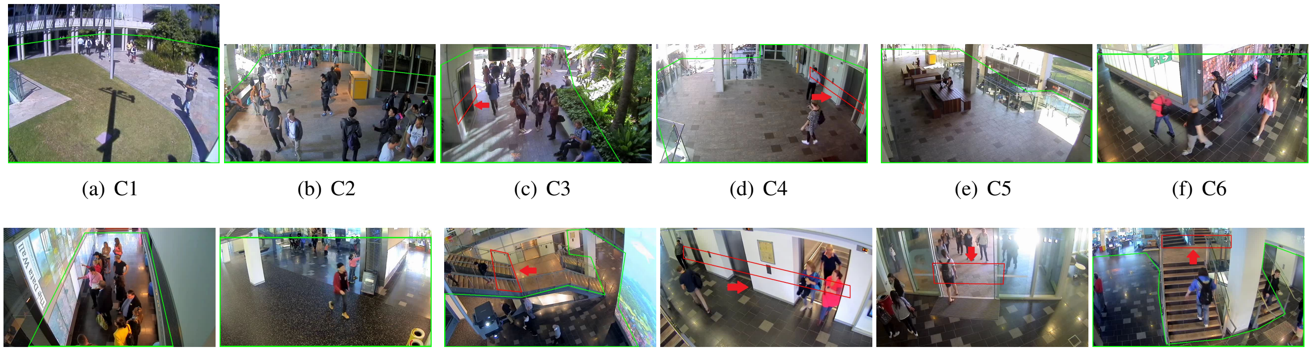


Figure 1. Floor plan of the monitored area, showing approximate camera locations and landmarks. A number of areas are covered including the approach to the building (C1), covered areas outside the main entrance (C2-C5) including two external elevators (C3 and C4), the main entrance (C11), an interactive display area (C6-C9), internal elevators (C10) and internal stairs (C12). Images captured from each camera are shown in Figure 2.

To facilitate research into the large scale monitoring of crowds and building utilisation, we present a publicly available 12 camera database collected from cameras positioned



(a) C1 (b) C2 (c) C3 (d) C4 (e) C5 (f) C6

(g) C7 (h) C8 (i) C9 (j) C10 (k) C11 (l) C12

Figure 2. Images captured from the 12 cameras. The regions of interest for throughput estimation are shown in red, with the red arrow indicating the primary direction of movement. Regions of interest for crowd counting are shown in green.

in and around a building on a university campus[1]. These cameras capture the major entrances and approach to the building, as well as large indoor and outdoor spaces, and elevator foyers (see Figures 1 and 2). 8 hours of data for each camera captured on the same work day from 9am to 5pm, of which 2 hours is annotated with ground truth (11am to 1pm) for 10 crowd counting and 6 pedestrian flow areas. We describe a distributed architecture that is being used to monitor this environment in real-time, and allows for scalable large scale monitoring of buildings and complex infrastructure.

The remainder of this paper is organised as follows: Section 2 presents the proposed database; Section 3 describes a distributed analytics system that is used to monitor this environment; Section 4 evaluates of the analytics used and demonstrates how this data may be used to monitor utilisation within a building; and Section 5 concludes the paper.

2. Building Monitoring Database

Data is captured from 12 cameras, all of which cover aspects of a single floor of a busy university campus building. Figure 1 shows a floor plan with approximate camera locations, and Figure 2 shows images from each camera. Cameras cover a mix of indoor and outdoor environments, with overlap present between many of the cameras. The mix of cameras covering doorways, elevators and other choke-points, alongside cameras covering large open environments allows a mix of algorithms to be deployed. In particular, we focus on two types: 1) Algorithms to estimate crowd sizes; and 2) Algorithms to estimate pedestrian throughput. However, other algorithms including those to estimate dwell times [3] or detect events (be they events of interest or abnormal events [11]) could also be applied.

Data is captured for an 8 hour period, covering 9am to 5pm on a single week day. Capture is performed by extracting data from a video management system (VMS), with the VMS responsible for ensuring videos are captured at a consistent frame rate and thus maintain synchronisation. Wide variation in crowd density (from empty scenes to in excess of 50 people in a single camera) is observed through the sequences, as people come and go throughout the day. Footage is captured at a resolution of 856 × 480 (with the exception of $C1$ which is captured at 768 × 578), and a frame rate of 25 frames per second (except for C4, C5 and C9 which are captured at 30 fps).

Ground truth annotation for all cameras is performed for a two hour block from 11am to 1pm. For pedestrian throughput estimation, a region of interest is defined and all people who pass through the area in each direction (primary direction and the opposite direction) are annotated according to when their approximate centre of mass is centred within the region of interest. Within the database, the primary direction is always defined as the 'inbound' direction (i.e. entering the building, entering the lift, etc.). For crowd counting estimation, regions are defined in cameras of interest and one frame every minute (120 frames in total) is annotated with the location (approximate centre of mass) of all people. It should be noted that this limits the crowd counting annotation to local crowd counting approaches (i.e. those that aggregate counts for foreground regions [12] or perform a per-pixel count [5]). However, as recent studies (i.e. [13]) have shown local approaches to be superior to holistic approaches, we argue that collecting the impractically large amount of annotation that is often needed for holistic approaches is of little benefit. Figure 2 shows the regions for throughput estimation and crowd counting that are annotated for each camera, and the primary direction of motion for throughput estimation.

Cameras C2-C12 are also calibrated to a common world coordinate scheme using Tsai's approach [14]. Cameras are calibrated against the building floor plans to allow for real-world measurements to be made, and for multi-camera algorithms to be used on the captured data. C1 is not calibrated as the ground is uneven in this camera view, making accu-

[1] Database available at http://www.qut.edu.au/research/saivt, or contact the authors for details

rate calibration very difficult. As C1 has minimal overlap with other cameras except in the far field, this is not considered a significant limitation. A number of person bounding boxes are also annotated for each crowd counting area to allow average person height at any location to be estimated to account for perspective distortion. Background segmentation has also been computed for all cameras using [1].

3. System Architecture

A key consideration when monitoring a large area is the need for the system to be scalable. With this in mind, a distributed system is proposed where every instance of an algorithm is run only on a single camera, potentially on it's own virtual machine, with results being aggregated across multiple cameras. This distributed architecture is visualised in Figure 3, and the approach is outlined in the following subsections.

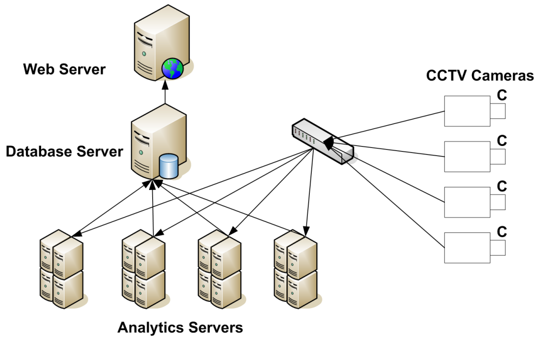


Figure 3. System Architecture: A number of analytics servers receive footage from a set of CCTV cameras. The outputs of the analytics (i.e. counts) are stored in a database, which can then be accessed by a web-server to display the outputs. In our installation, all servers are Ubuntu virtual servers.

3.1. Video Analytics

3.1.1 Crowd Counting

The size of a crowd in an intrinsically holistic measure, however local approaches that aggregate counts over motion regions [12] or individual pixels [5] have been shown to outperform their holistic counterparts [13]. Furthermore, they require less training data and are applicable to multi-camera scenarios. We adopt the local approach of [12] which extracts features from regions of motion, and estimates the number of people in each region. Features extracted from each motion region ([1] is used to perform motion segmentation) include the motion region area and perimeter (size features), a perimeter orientation histogram (shape features), an edge angle histogram (edge features), and keypoint features. Gaussian process regression [9] is used to estimate the number of people in each region, and the total count is obtained by summing the region counts.

This approach is chosen as it is a local method (based on counting motion regions) that is also able to operate in a scene independent, multi-camera manner through the use of camera calibration. We adopt the 'pixel' method for multi-camera crowd counting (see [12]), that enables multi-camera crowd counting to be performed as a secondary step. This method assigns a fractional count to each pixel based on the crowd density, weights the factional count according to how many other cameras can see the pixel, and aggregates these fractional counts across a multi-camera network. This makes the method highly scalable, as each camera can be counted independently, and a subsequent process can aggregate the results and account for overlap.

3.1.2 The Virtual Gate

Previous approaches to pedestrian flow analysis have extracted features from a line or region within an image. Kim [4] defined a virtual gate as a single line in the scene, and integrated optical flow perpendicular to the gate over time to estimate the crowd count; Lin [6] segmented entry/exit events using a 2D virtual gate rather than a single line; while Ma [7] used local HOG features and integer programming to estimate the number of people passing a line. We use a local feature approach that can be viewed as an analogue to the crowd counting approach outlined in Section 3.1.1. It is comprised of a counting line surrounded by a region of interest (ROI), and a direction of interest (DOI) in which pedestrians move. The set of pixels belonging to the ROI is denoted R, and the unit vector pointing in the DOI is denoted $\mathbf{d}$. The proposed algorithm accumulates optical flow in the direction of interest at a set of points, determined according to a feature point selection criteria. Each video sequence is divided into a set of sub-sequences, or windows, in which the optical flow is accumulated. Regression is then applied to each window independently to estimate the number of people passing through the virtual gate.

Three feature point selection criteria are used in this approach: 1) All Pixels: treats every pixel within the ROI as a feature point; 2) Edges: Canny edge detection [2] is used to detect edges in the ROI, and these pixels form the feature set; 3) Corners: FAST [10] is used to locate a set of corners to form the feature points.

The features within each window are calculated from the optical flow. The optical flow field at time t is denoted v_t, and the optical flow at a pixel $\mathbf{p}$ is denoted $v_t(\mathbf{p})$. The component of this flow which points in the direction of interest $\mathbf{d}$ is referred to as the aligned optical flow, and is computed using the dot product,

$$\hat{v}_t(\mathbf{p}) = v_t(\mathbf{p}) \cdot \mathbf{d}. \tag{1}$$

The set of feature points detected within the ROI at time t is denoted $F_{t,f}$, where the subscript f represents the type of feature under consideration (all pixels, edges or corners). At each frame in the video we calculate the total aligned flow as follows,

$$a_{t,f} = \sum_{\mathbf{p} \in F_{t,j}} \hat{v}_t(\mathbf{p}). \qquad (2)$$

The video sequence is split into a series of time windows enumerated by n. The set of frames belonging to the nth window is denoted W_n, and each window is taken to be the same length. Across each time window W_n the total aligned flow is accumulated, $\alpha_{n,f} = \sum_{t \in W_n} a_{t,f}$.

Although feature points are not explicitly tracked, this summation will be roughly proportional to the number of feature points crossing the counting line, as the summation of aligned flow for each feature point is equal to the distance it travels through the gate (i.e. the width of the ROI). This results in the total aligned flow over a time window, $\alpha_{n,f}$, being proportional to the number of points crossing the gate.

Optical flow histograms are used to separate the effects of potential noise and true motion. In practice, values of optical flow are subject to error or noise. For instance, a feature point belonging to a background object which does not move has a true optical flow of 0, but in practice may be assigned a small fractional value such as 0.02. In order to separate the effects of noise and true motion, a histogram based on flow magnitude is used. Aligned optical flow (Equations 1 and 2) is calculated within different histogram bins as follows:

1. Each pixel $\mathbf{p}$ is assigned to a histogram bin b based on the magnitude of $\hat{v}_t(\mathbf{p})$. Bin ranges of [0, 0.05), [0.05, 0.25) and [0.25,∞) are used in this paper. The set of pixels belonging to bin b is denoted H_b.

2. The total aligned flow for bin b and for feature f is denoted, $a_{t,f,b} = \sum_{\mathbf{p} \in F_{t,f} \cap H_b} \hat{v}_t(\mathbf{p})$.

3. The total aligned flow across a time window W_n for bin b and feature f is then calculated as follows, $\alpha_{n,f,b} = \sum_{t \in W_n} a_{t,f,b}$.

The set of all features and histogram bins, $\{\alpha_{n,f,b}\}_{f,b}$, are collected into a feature vector x_n which describes the time window W_n. Note that when multiple features and histogram bins are used, these features are concatenated to form a larger feature vector. A regression model (Gaussian Process Regression [9]) is then trained to learn the relationship between the feature vectors $\{x_n\}$ and the ground truth values. To allow for bi-directional counting, we split the accumulation of aligned flow according to the sign of the flow, such that positive and negative aligned flow are accumulated separately for feature points. This results in two feature vectors, $x_{n,+}$ and $x_{n,-}$, for each window, and by training two regression models we can estimate pedestrian flow is both directions.

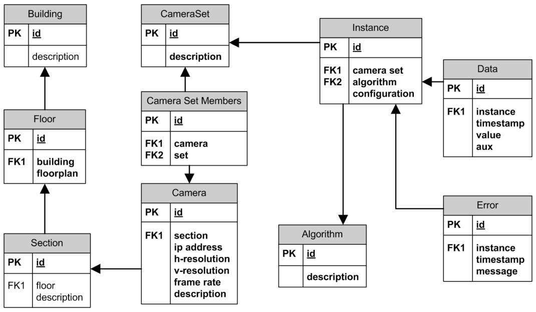


Figure 4. Database Schema: The schema contains a coarse environment definition as well as available cameras and algorithms. Data and errors are associated with specific camera set-algorithm pairs (instances) and are timestamped to allow for efficient retrieval.

3.2. Data Management

A relational database is used to store data generated by the analytics, and allow for subsequent display and analysis. The database schema is shown in Figure 4. The database schema provides a coarse definition of the environment broken down into building, floors and sections (such as inside and outside) that allows counts for related regions to be quickly aggregated. Separate tables define the cameras (and sets of cameras) and algorithms, while another table (instance) details which algorithms are run on which sets of cameras. This allows both single and multi-camera algorithms to be incorporated into the database, and for cameras to be used by multiple algorithms. This structure also has the potential to allow the automatic instantiation of virtual machines for each instance. Data extracted by the instances and errors raised are stored in further tables, both of which use timestamps to allow data to be quickly queried.

3.3. Visualisation

A web server is used to visualise collected data. We use the Django[2] toolbox as a foundation for the web application, and focus on two main pieces of functionality: the ability to quickly view current system performance using a floor plan dynamically updated with current observations, and the ability to plot data from from one or more analytics for arbitrary time periods; both of which can be easily achieved through interaction with the database server.

4. System Evaluation and Performance

We first evaluate the performance of the crowd counting and virtual gate algorithms using the annotated portion of the data to demonstrate the characteristics of and challenges present in the data (see Section 4.1); after which we explore the relationships between the counts obtained by different cameras over the complete 8 hour sequence to demonstrate how building utilisation may be explored (see Section 4.2).

[2] https://www.djangoproject.com/

Camera	MSE	MRE	Camera	MSE	MRE
C1	1.71	26.67	C2	1.62	36.28
C3	1.20	24.98	C4	0.55	31.80
C5	17.96	39.44	C6	1.75	37.51
C7	4.82	47.21	C8	6.87	44.86
C9	3.75	84.86	C12	1.58	53.57

Table 1. Crowd Counting Accuracy. Mean Squared Error (MSE) and Mean Relative Error (MRE) are reported, averaged for each frame of ground truth. Frames with a ground truth of 0 are excluded from MRE calculations as the MRE is undefined.

Camera	Inbounds			Outbounds		
	MSE	MRE	RE	MSE	MRE	RE
C3	0.91	108.6	252.7	1.28	65.6	417.0
C4	1.17	59.4	14.1	1.46	51.9	15.8
C9	0.79	29.9	3.7	0.77	27.6	8.3
C10	0.68	39.1	5.4	1.29	40.9	3.9
C11	3.93	32.2	1.1	4.34	30.5	5.5
C12	1.88	39.7	3.8	1.76	48.9	2.8

Table 2. Virtual Gate Accuracy. Mean Squared Error (MSE), Mean Relative Error (MRE) (both computed over non-overlapping 1 minute windows) and Relative Error (RE) over the entire sequence are reported for each direction. Windows with a ground truth of 0 are excluded from MRE calculations as the MRE is undefined.

4.1. Evaluation of Analytics Performance

For each camera, two hours of data (from 11am to 1pm) is annotated with ground truth which is provided with the dataset. We designate the first hour of this data for training, and the second for testing. We report accuracy for the crowd counting and virtual gate approaches in Tables 1 and 2 respectively.

As can be seen by Tables 1 and 2, in the majority of cases reasonable performance is achieved with low to moderate levels of error. However, poor performance is observed for some configurations. The crowd counting accuracy is observed to be worse than the virtual gates in almost all cases, however as can be seen in Figure 5, even the worst performing cameras still follow the ground truth and detect trends and changes. Inaccuracies are observed due to several environmental factors such as shadows and reflections (for the interior cameras) which causes false motion (and thus counts), and due to people sitting still for long periods of time causing missed counts. Errors in the virtual gates are in part due to the windowing effect, as can be seen by greatly reduced RE over the whole sequence compared to the MRE for individual windows. However particularly poor performance is observed for the virtual gate in C3, which monitors an external lift (see Figure 2 (c)). Figure 6 plots the estimated and ground truth counts for C3, and it is evident that although the crowd size estimate closely follows the ground truth, the estimate of throughput at the elevator doors is highly inaccurate. The area around the lift door is also frequently exposed to large amounts of crowding as students leave the building and lecture theatres, making accurate estimation of entries and exits to the lift very difficult. Similar problems, albeit less severe, are observed for people entering the other external lift in C4.

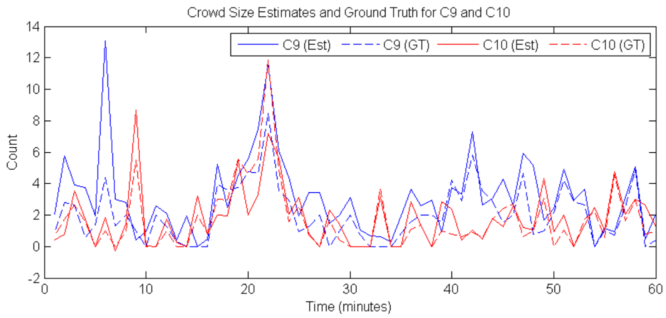


Figure 5. Crowd size estimates and ground truth for the two least accurate cameras (in terms of MRE), C9 and C10. Despite poor MRE, both estimates follow the ground truth well and high MRE is due to isolated errors, and errors made with very low crowd sizes.

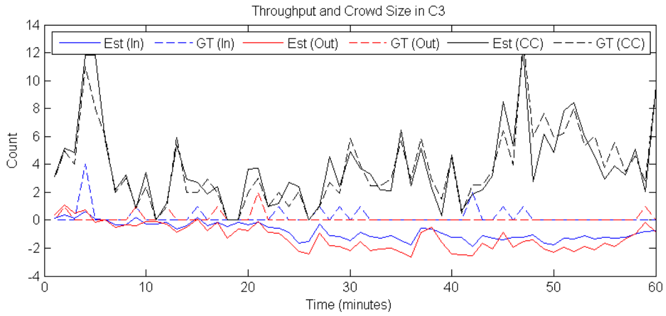
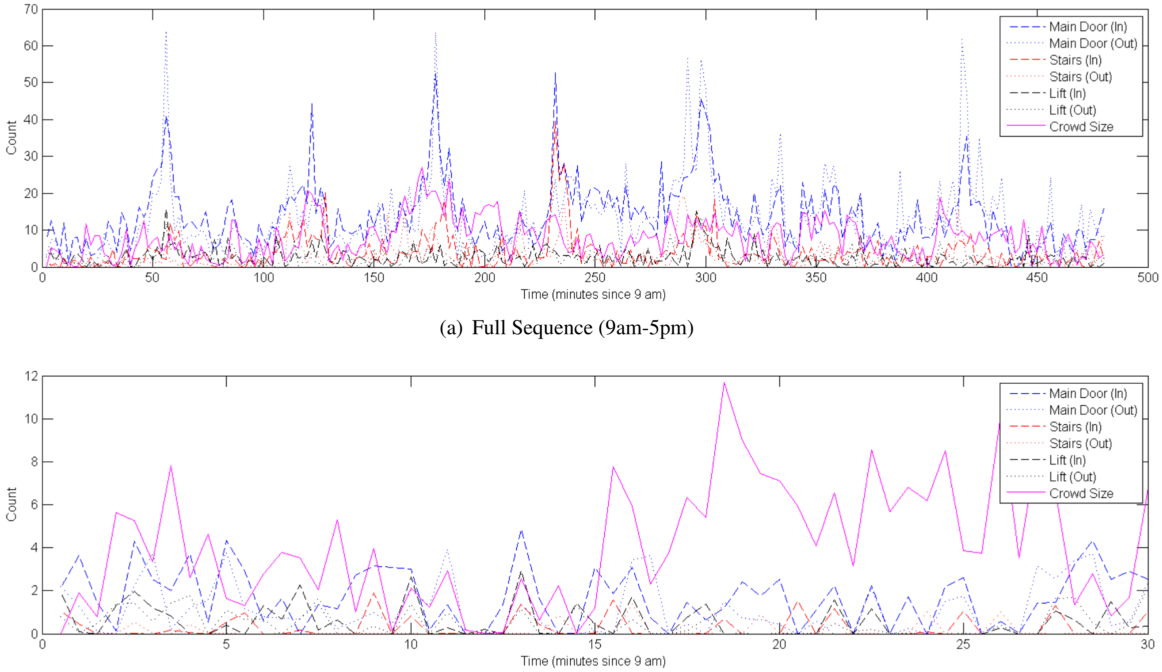


Figure 6. Crowd ('CC') and throughput ('In' and 'Out') counts over time for C3. In C3 the elevator door is frequently obscured by crowds, and high levels of crowding lead to very inaccurate flow estimation for the elevator.

4.2. Monitoring a Complete Building

The database presented in this paper allows us to investigate how a building is utilised over time. Figure 7 shows how pedestrian throughput at three points and crowd size varies over the course of a day. It is clearly visible that the crowd movements through the three cameras are related, and on closer inspection (Figure 7 (b)) it can be seen that there is a small lag between the cameras for the virtual gate counts, caused by the time it takes for people to move from one area to another. The link between crowd size and throughput can also be seen. As people enter the building (or exit from the stairs or lift) a number will remain in the building and join the crowd; and reductions in the crowd size can be traced to people leaving the area. These relationships raise the question of whether estimates for multiple virtual gates, crowd estimates, or even both, can be performed simultaneously using aggregated data. Using techniques such as Gaussian processes [8] or Bayesian



(a) Full Sequence (9am-5pm)

(b) First Half Hour (9am-9.30am)

Figure 7. Crowd movements over time. The plots show virtual gate counts covering the building entry, lift, and stairs; and the crowd count for the internal foyer (C6-9).

networks [15], it may be possible to combine multiple distributed observations to learn a much richer model that describes how people interact with the building.

5. Conclusions

In this paper we have presented a large, multi-camera dataset for distributed pedestrian monitoring. Existing crowd monitoring approaches and datasets have been predominately single camera and captured over short time spans. The proposed database contains 8 hours of footage from 12 cameras (96 hours total) located in and around a busy university campus building. This database enables new research opportunities to learn richer models of pedestrian movements and interactions by incorporating observations from multiple view points; something not possible with existing datasets. We have also described a distributed system to monitor such an environment in real-time using crowd counting and throughput estimation. Future work will concern investigation into modelling the relationships between views, incorporating additional analytics, and extending the system to cover other areas (i.e. a public food court, parking lots) or different infrastructure (i.e. airports).

References

[1] O. Barnich and M. Van Droogenbroeck. Vibe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20(6):1709–1724, 2011. 3

[2] J. Canny. A computational approach to edge detection. *T-PAMI*, (6):679–698, 1986. 3

[3] S. Denman, A. Bialkowski, C. Fookes, and S. Sridharan. Determining operational measures from multi-camera surveillance systems using soft biometrics. In *AVSS*, pages 462–467, 2011. 2

[4] B.-S. Kim, G.-G. Lee, J.-Y. Yoon, J.-J. Kim, and W.-Y. Kim. A method of counting pedestrians in crowded scenes. In *ICIC*, pages 1117–1126, Berlin, Heidelberg, 2008. Springer-Verlag. 1, 3

[5] V. Lempitsky and A. Zisserman. Learning to count objects in images. In *NIPS*, pages 1324–1332, 2010. 1, 2, 3

[6] D.-T. Lin and L.-W. Liu. Real-time detection of passing objects using virtual gate and motion vector analysis. In *UIC*, pages 710–719, Berlin, Heidelberg, 2008. Springer-Verlag. 1, 3

[7] Z. Ma and A. B. Chan. Crossing the line: Crowd counting by integer programming with local features. In *CVPR*, pages 2539–2546, 2013. 1, 3

[8] M. A. Osborne, S. J. Roberts, A. Rogers, and N. R. Jennings. Real-time information processing of environmental sensor network data using bayesian gaussian processes. *ACM Transactions on Sensor Networks*, 9(1):1, 2012. 5

[9] C. E. Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. 2004. 3, 4

[10] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *T-PAMI*, 32(1):105–119, 2010. 3

[11] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Textures of optical flow for real-time anomaly detection in crowds. In *AVSS*, pages 230–235, 2011. 2

[12] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Scene invariant multi camera crowd counting. *PRL*, 44:98–112, 2014. 1, 2, 3

[13] D. Ryan, S. Denman, S. Sridharan, and C. Fookes. An evaluation of crowd counting methods, features and regression models. *CVIU*, 130:1–17, 2015. 2, 3

[14] R. Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. *CVPR*, pages 364–374, 1986. 2

[15] P. P.-Y. Wu, J. Pitchforth, and K. Mengersen. A hybrid queue-based bayesian network framework for passenger facilitation modelling. *Transportation Research Part C: Emerging Technologies*, 46:247–260, 2014. 6