Analysis of Crowded Scenes using Holistic Properties

Antoni B. Chan Mulloy Morrow Nuno Vasconcelos
Department of Electrical and Computer Engineering,
University of California, San Diego
abchan@ucsd.edu, mmorrow@ucsd.edu, nuno@ece.ucsd.edu

Abstract

We present results on the PETS 2009 dataset using surveillance systems based on holistic properties of the video. In particular, we evaluate a crowd counting system, based on regression of holistic (global) features, on the PETS 2009 dataset. We also present experimental results on crowd event detection when using the dynamic texture model to represent holistic motion flow in the video.

1. Introduction

There is currently a great interest in vision technology for monitoring all types of environments. This could have many goals, *e.g*. security, resource management, or advertising. From the technological standpoint, computer vision solutions typically focus on detecting, tracking, and analyzing individuals in the scene. However, there are many problems in environment monitoring that can be solved without explicit tracking of individuals. These are problems where all the information required to perform the task can be gathered by analyzing the environment *holistically*: *e.g*. monitoring of traffic flows, detection of disturbances in public spaces, detection of speeding on highways, or estimation of the size of moving crowds. By definition, these tasks are based on either properties of 1) the "crowd" as a whole, or 2) an individual's "deviation" from the crowd. In both cases, to accomplish the task it should suffice to build good *models for the patterns of crowd behavior*. Events could then be detected as *variations in these patterns*, and abnormal individual actions could be detected as *outliers* with respect to the crowd behavior.

In this work, we demonstrate the efficacy of computer vision surveillance systems that utilize holistic representations of the scene on the PETS 2009 database. In particular, we test a crowd counting system that is based on segmenting the crowd into sub-parts of interest (*e.g*. groups of people moving in different directions) and estimating the number of people by analyzing *holistic* properties of each component [1]. We also perform event recognition by holistically modeling the crowd flow in the scene using the dynamic texture model, and training event classifiers on dynamic textures [2]. In the remainder of this paper, we review the crowd counting system from [1] in Section 2, and the dynamic texture classifiers from [2] in Section 3. We then discuss the experimental setup and results on the PETS 2009 dataset in Section 4 and 5.

2. Crowd counting using low-level features

We adopt the crowd counting system proposed in [1], which is based on regression of low-level features. Consider the example scene shown in Figure 1, where the goal is to estimate the number of people moving in each direction. Given a segmentation into the two sub-components of the crowd, the key insight is that certain *low-level global features*, extracted from the crowd segment, are indeed good indicators of the number of people in the crowd segment. Intuitively, one such features, assuming proper normalization for the perspective of the scene, is the area of the crowd segment (*i.e*., the number of pixels in the segment). In the ideal case, this relationship should be linear. However, due to a variety of confounding factors (*e.g*. occlusions and segmentation errors), this relationship may deviate from linearity. This indicates that additional features are required to better model the crowd count, along with a suitable regression function.

The counting system of [1] is outlined in Figure 2. The video is segmented into crowd regions moving in different directions, using a mixture of dynamic textures [3]. For each crowd segment, various features are extracted, while applying a perspective map to weight each image location according to its approximate size in the real scene. Finally, the number of people per segment is estimated from the feature vector with Gaussian process regression.

2.1. Crowd segmentation

The *mixture of dynamic textures* [3] is used to segment the crowds moving in different directions. The video is represented as a collection of spatio-temporal patches, which

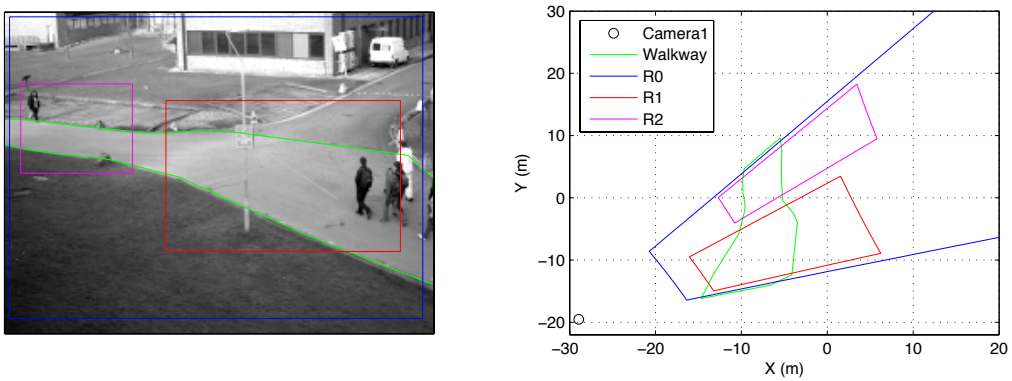

Figure 1. (left) an example of View 1 from the PETS 2009 dataset, along with the regions-of-interest (R0=blue, R1=red, R2=magenta); (right) the sidewalk and regions-of-interest projected into the 3-d scene.

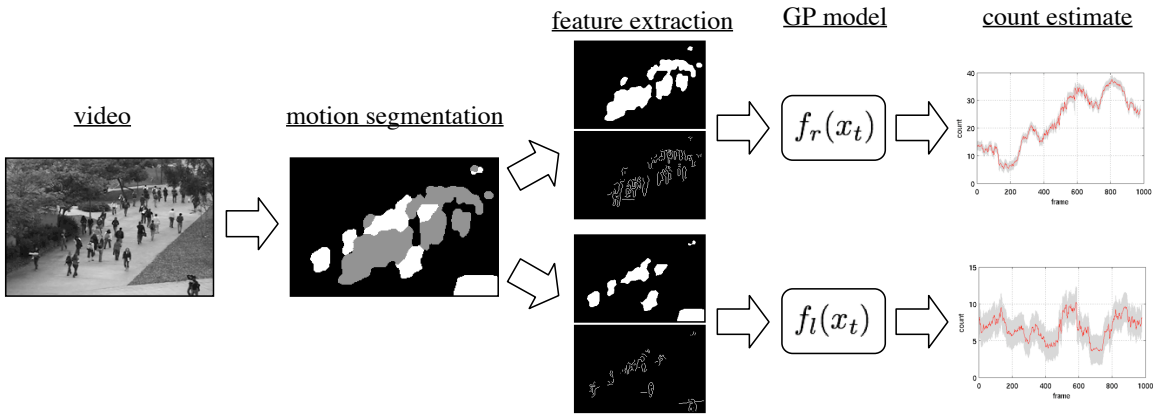

Figure 2. Crowd counting system: the scene is segmented into crowds moving in different directions. Features, which are normalized to account for perspective, are extracted from each segment, and the number of people in each segment is estimated with Gaussian process regression.

are modeled as independent samples from a mixture of dynamic textures. The mixture model is learned with the expectation-maximization (EM) algorithm [3]. Video locations are then scanned sequentially, a patch is extracted at each location, and assigned to the mixture component of largest posterior probability. The location is declared to belong to the segmentation region associated with that component. In this work, we use the `13-57` and `13-59` sequences from View 1 to train the segmentation model. The remaining video was then segmented by computing the posterior assignments as before. More implementation details are available in [3].

2.2. Perspective normalization

Before extracting features from the crowd segments, it is important to consider the effects of perspective. Because objects closer to the camera appear larger, any feature (*e.g.* the segment area) extracted from a foreground object will account for a smaller portion of the object than one extracted from an object farther away. This makes it important to normalize the features for perspective. In this work,

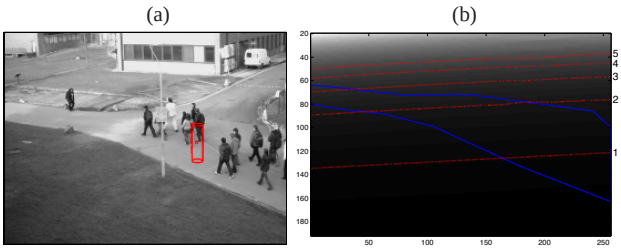

Figure 3. Perspective map for View 1: a) an example of a virtual person modeled as a cylinder; b) the perspective map for View 1, with contours (red) and sidewalk (blue).

we weight each pixel according to a perspective normalization map, which applies a pixel weight that is based on the perceived size of an object at different depths, with larger weights given to further objects.

The perspective map was constructed by moving a virtual person, approximated as a cylinder with height 1.75m and radius 0.25m, in the 3d-scene. For each pixel (x, y) in the 2-d camera view, a cylinder was positioned in the 3-d scene such that the center of the cylinder projects onto

(x, y) in the 2-d view (*e.g.* see Figure 3a). The number of total pixels used by the filled-in cylinder is denoted as $c(x, y)$. The perspective map was then computed as $M(x, y) = c(230, 123)/c(x, y)$, where the coordinates $(230, 123)$ correspond to a reference person on the right-side of the walkway. Figure 3b shows the perspective map for View 1, with contour lines showing where the weights of the pixels are $\{1, \cdots, 5\}$. Finally, $M(x, y)$ is the perspective map for extracting features based on the area or size of the object. When the features are based on edges (*e.g.* edge histograms), then the weights are the square-roots of the perspective map, $\sqrt{M(x, y)}$.

2.3. Feature extraction

Ideally, features such as segmentation area or number of edges should vary linearly with the number of people in the scene [4,5]. However, local non-linearities in the regression function arise from a variety of factors, including occlusion, segmentation errors, and pedestrian configuration (*e.g.* spacing within a segment). To model these non-linearities, we extract a total of 30 features from each crowd segment.

2.3.1 Segment features

These features capture properties of the segment, such as shape and size.

- *Area* – total number of pixels in the segment.
- *Perimeter* – total number of pixels on the segment perimeter.
- *Perimeter edge orientation* – a 6-bin orientation histogram of the segment perimeter. The orientation of each edge pixel is computed by finding the maximum response to a set of oriented Gaussian filters, with opposite orientations (180° apart) considered the same.
- *Perimeter-area ratio* – ratio between the segment perimeter and area, which measures the complexity of the segment shape.
- *"Blob" count* – the number of connected components with more than 10 pixels in the segment.

2.3.2 Internal edge features

The edges contained in a crowd segment are a strong clue about the number of people in the segment [4, 6]. A Canny edge detector [7] is applied to each frame, and the result is then masked by the crowd segmentation, forming the internal edge image. The following edge features are computed:

- *Total edge pixels* – total number of internal edge pixels contained in the segment.
- *Edge orientation* – 6-bin histogram of the edge orientations in the segment, generated in the same way as the perimeter orientation histogram.
- *Minkowski dimension* - a fractal dimension of the internal edges, which estimates the degree of "space-filling" of the edges [8].

2.3.3 Texture features

Texture features are computed using the gray-level co-occurrence matrix (GLCM), similar to [9]. First, the image is quantized into 8 gray levels (from 256), and the joint histogram of neighboring pixel values $p(i, j|\theta)$ is estimated for angles $\theta \in \{0^\circ, 45^\circ, 90^\circ, 135^\circ\}$. The following texture properties are computed for each θ, resulting in a total of 12 texture features:

- *Homogeneity*: measures the smoothness of the texture.
- *Energy*: measures the total sum-squared energy.
- *Entropy*: measures the randomness of the texture distribution.

2.4. Gaussian process regression

Feature vectors for each frame are formed by concatenating all 30 features, described in Section 2.3, into a single vector $x \in \mathbb{R}^{30}$. A Gaussian process (GP) [10] is used to regress feature vectors to the number of people per segment. The GP defines a distribution over functions, which is "pinned down" at the training points. The classes of functions that the GP can model is dependent on the kernel function used. For the task of pedestrian counting, we note that the dominant trend of many of the features is linear (*e.g.* segment area), with local non-linearities. To capture both trends, we combine the linear and the squared-exponential (RBF) kernels, *i.e.*

$$k(x_p, x_q) = \alpha_1(x_p^T x_q + 1) + \alpha_2 e^{\frac{-\|x_p - x_q\|^2}{\alpha_3}} + \alpha_4 \delta(p, q)$$

with hyperparameters $\alpha = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$. The first and second terms of the kernel are the linear and RBF components, while the third term models observation noise. The hyperparameters are learned by maximizing the marginal likelihood of the training data.

3. Event Classification with Dynamic Textures

Since most of the information required for the classification of crowd events is contained in the interaction between the many motions that it contains, a holistic representation can be used to capture the variability of the motion field without the need for segmenting or tracking individual components. We adopt the methodology of [2], where the entire motion field of each video is modeled as a dynamic texture (DT) [11]. A dynamic texture is a generative probabilistic model that treats a video as a sample from a linear dynamical system (LDS),

$$\begin{cases} y_t = Cx_t + w_t + \bar{y} \\ x_t = Ax_{t-1} + v_t \end{cases} \quad (1)$$

scenario	test video	region	training set	training frames
S1.L1	13-57	R0, R1, R2	13-59, 13-59F, 14-03, 14-03F	1308
S1.L1	13-59	R0, R1, R2	13-57, 13-57F, 14-03, 14-03F	1268
S1.L2	14-06	R1, R2	13-57, 13-57F, 13-59, 13-59F, 14-03, 14-03F	1750
S1.L3	14-17	R1	13-57, 13-57F, 13-59, 13-59F, 14-03, 14-03F	1750

Table 1. test videos and training sets for PETS 2009 counting experiments

		total			**right**			**left**			**PETS**			
scenario video	region	error	MSE	bias	error	MSE	bias	error	MSE	bias	pred.	true	error	frames
S1.L1 13-57	R0	2.308	8.362	-1.855	0.249	0.339	0.131	2.475	8.955	-2.032	4411	4838	2.46	218
	R1	1.697	5.000	-1.615	0.100	0.100	-0.045	1.643	4.720	-1.579	2301	2757	2.28	217
	R2	1.072	1.796	-0.258	0.235	0.317	0.217	0.842	1.484	-0.462	1436	1437	0.99	201
S1.L1 13-59	R0	1.647	4.087	-1.025	1.668	4.158	-1.154	0.154	0.154	0.145	3455	3628	1.41	240
	R1	0.685	1.116	0.120	0.589	0.871	0.049	0.095	0.095	0.087	1636	1539	0.69	217
	R2	1.282	2.577	-1.000	1.291	2.436	-1.025	0.066	0.066	-0.058	1313	1473	1.23	228
S1.L2 14-06	R1	4.328	44.159	-4.219	4.338	44.159	-4.219	0.005	0.005	0.005	1727	2462	5.89	131
	R2	3.139	26.035	-2.891	3.144	26.219	-2.915	0.020	0.020	0.020	1078	1629	4.48	132
S1.L3 14-17	R1	0.604	1.220	0.385	0.604	1.198	0.407	0.000	0.000	0.000	518	481	0.98	50

Table 2. Crowd Counting Results on PETS 2009: (left) per-frame average results; (right) results using PETS ground-truth.

where $y_t \in \mathbb{R}^m$ encodes the vectorized video frame at time t, and $x_t \in \mathbb{R}^n$ is a hidden state variable ($n < m$) that represents the dynamics of the video over time. The matrix $C \in \mathbb{R}^{m \times n}$ is the observation matrix, that projects from the hidden state space to the observations, $A \in \mathbb{R}^{n \times n}$ is the transition matrix that controls the evolution of the hidden state (and hence, the motion of the video) over time, and $\bar{y}$ is the video mean. Finally, w_t and v_t are normally distributed noise terms with zero mean and covariance matrices $R \in rI_m$ and $Q \in \mathbb{R}^{n \times n}$, respectively. When the DT parameters are learned as in [11], the observation matrix contains the principal components of the video, and x_t are the corresponding time-varying PCA coefficients.

An event classifier is trained as follows. Given a set of training video clips and the corresponding classes, a dynamic texture is learned for each video clip. For the nearest-neighbor classifier, we use the Kullback-Leibler (KL) divergence [2] or the Martin distance [12] to find the closest dynamic texture to that of the test clip. An SVM classifier is also trained using the KL kernel [14], which is a kernel function for probability distributions that is analogous to the RBF kernel for real vectors.

4. Counting Experiments on PETS 2009

In this section we present the experimental results on the PETS 2009 dataset using the counting system of Section 2.

4.1. Data and Setup

The counting experiments use View 1 from the 13-57, 13-59, 14-03, 14-06, and 14-17 videos from the PETS 2009 dataset. An example frame of video is shown in Figure 1, along with the three regions of interests (R0, R1, and R2). The ground-truth count was obtained by annotating the number of left-moving and right-moving people by hand in every 5th frame. The count in the remaining frames were obtained using linear interpolation.

A counting regression function was learned for each of the test videos and regions using the training set listed in Table 1. A regression function was learned for right-moving and left-moving people classes. Because of the small amount of training data, we augment the training set by flipping the classes of each of the training videos (this is denoted as a video with an F suffix). The counts for the left- and right-moving people classes were obtained by rounding the regression predictions to the nearest non-negative integer. The total count was obtained by summing the left and right counts. We report the absolute error, mean-squared error, and error bias between the ground-truth count and the estimates, averaged over the frames of each test video.

4.2. Counting Results

A summary of the counting results, with respect to our hand-annotated ground-truth, for the total, left-moving, and right-moving count predicitions are given in Table 2 (left). For the 13-57, 13-59, and 14-17 videos, the count error is reasonable (2.3, 1.6 and 0.6). However, the error is much larger for the 14-06 video (4.3 error). This is because the 14-06 video contains a very dense crowd, which is not represented in the training set. Hence, the system underestimates the counts for this video. Table 2 (right) presents the overall predicted counts, true count, and per-frame error rate using the PETS ground-truth data. The error rates are similar to those using our ground-truth data, with differences due to the number of frames evaluated and the method of hand-annotation.

A plot of the counts over time for the various regions (R0, R1, and R2) and classes (total, right, and left) are shown in Figures 4 and 5. Again, in general, the count estimates follow the ground-truth counts, except for the 14-06 video. Several example frames of each video are shown in Figures 6, 7, and 8. The region-of-interest and crowd esti-

mates are displayed in each frame. In addition, the crowd segmentation and ROI are projected into the 3-d scene, to obtain an occupancy map of the sidewalk, which appears below each example frame.

5. Event Recognition on PETS 2009

In this section, we report results on crowd event recognition on the PETS 2009 dataset.

5.1. Data and Setup

Videos from the PETS 2009 dataset were collected and the frames were annotated with one of 6 classes: walking, running, merging, splitting, evacuation, and dispersion (see Table 3). Each video was downsampled by 8 and split into chunks of 20 frames each. A dynamic texture was learned for each chunk, and the distance matrices for the Martin distance (MD), state-space KL divergence (SKL), and image-space KL divergence (IKL) were computed. The dataset was then split into training (75%) and test (25%) sets, and a nearest-neighbor (NN) or support vector machine (SVM) was learned for each of the 6 classes (*e.g.* walking vs. not-walking). We report the error for each classifier on the test set, averaged over the 4 splits of the data. The experiment was repeated for each view (Views 1, 2, 3, and 4), and for all four views combined.

5.2. Recognition Results

The event classification results are presented in Table 4, for the different classes and views. The average error rate is also shown for each view. Overall, the MD-NN classifier has the lowest average error rate for 3 out of 4 of the views.

Finally, to obtain a probability score for classification, we combine the MD-NN and SKL-NN classifier decisions for the 4 views using a voting scheme (8 votes total). If the probability is greater than 0.5 (more than 4 votes), then the event is detected. Figure 9 shows several examples of event detection on video `14-33`. In the beginning of the video, people walk towards the center of the frame, and the "walking" and "merging" events are detected. Next, the people form a group in the center of the frame, and no events are detected since the people are not moving. Finally, when the people run away from the center, the "running", "evacuation", and "dispersion" events are detected.

6. Conclusions

In this paper, we have presented results on the PETS 2009 dataset using surveillance systems based on holistic properties of the video. In particular, experimental results indicate that crowd counting using low-level global features is indeed accurate and viable. We have also presented crowd event detection results, which indicates crowd events, such as evacuation, dispersion, and running, can be detected using a global representation of motion flow in the video.

class	set:video[frames]
walking	S0:`12-34`[all]; S0:`13-06`[0-42]; S0:`13-19`[82-218]; S0:`13-19`[320-end]; S0:`13-38`[0-48]; S0:`14-03`[all]; S0:`14-06`[all]; S1.L1:`13-57`[all]; S1.L1:`13-59`[all]; S2.L3:`14-41`[all]; S3:`12-43`[all]; S3:`14-13`[all]; S3:`14-37`[all]; S3:`14-46`[all]; S3:`14-52`[all]; S3:`14-16`[0-30,108-162]; S3:`14-31`[0-42,48-130]; S3:`14-33`[265-310];
running	S3:`14-16`[36-end]; S3:`14-33`[328-377];
evacuation	S3:`14-16`[36-102,180-end];
dispersion	S3:`14-27`[96-144,270-318]; S3:`14-33`[328-377];
merging	S3:`14-33`[0-180];
splitting	S3:`14-31`[48-130]

Table 3. Videos used for crowd event detection

		MD	SKL		IKL	
	view	NN	NN	SVM	NN	SVM
walking	001	0.05	**0.02**	0.06	0.11	0.10
	002	0.12	0.08	0.13	0.05	**0.04**
	003	0.06	**0.05**	0.11	0.24	0.13
	004	0.12	**0.11**	0.17	0.28	0.20
	all	**0.13**	0.14	0.17	0.34	**0.13**
running	001	**0.03**	**0.03**	0.09	**0.03**	**0.03**
	002	0.06	0.09	0.20	**0.03**	0.06
	003	**0.00**	**0.00**	0.06	0.03	0.03
	004	0.09	**0.06**	0.19	0.10	0.10
	all	0.15	0.16	0.16	**0.12**	0.15
evacuation	001	0.10	0.10	0.07	0.10	**0.06**
	002	0.13	0.15	**0.07**	**0.07**	**0.07**
	003	0.10	0.10	0.07	**0.03**	0.07
	004	0.09	0.09	**0.06**	0.07	0.07
	all	0.12	0.12	0.08	**0.06**	**0.06**
dispersion	001	**0.15**	**0.15**	0.34	0.42	0.31
	002	**0.21**	**0.21**	**0.21**	0.37	0.26
	003	0.29	0.29	0.34	0.27	**0.10**
	004	**0.05**	**0.05**	0.28	0.36	0.32
	all	0.22	0.25	0.30	0.25	**0.20**
merging	001	0.37	**0.31**	0.39	0.39	0.45
	002	0.39	0.37	0.45	0.45	**0.27**
	003	0.33	0.45	**0.18**	0.52	0.45
	004	**0.12**	0.60	0.33	0.54	0.37
	all	0.44	0.44	0.42	0.44	**0.32**
splitting	001	0.33	0.33	**0.27**	**0.27**	**0.27**
	002	**0.06**	**0.06**	0.27	0.25	0.33
	003	**0.00**	0.06	0.20	0.39	0.27
	004	**0.27**	**0.27**	0.35	0.39	**0.27**
	all	0.30	0.30	0.28	**0.23**	0.26
average	001	0.17	**0.16**	0.20	0.22	0.20
	002	**0.16**	**0.16**	0.22	0.20	0.17
	003	**0.13**	0.16	0.16	0.25	0.18
	004	**0.12**	0.20	0.23	0.29	0.22
	all	0.23	0.24	0.24	0.24	**0.19**

Table 4. Error rates for classifying the 6 crowd events in PETS 2009

Acknowledgements

This work was partially funded by NSF awards IIS-0534985, IIS-0448609, and DGE- 0333451.

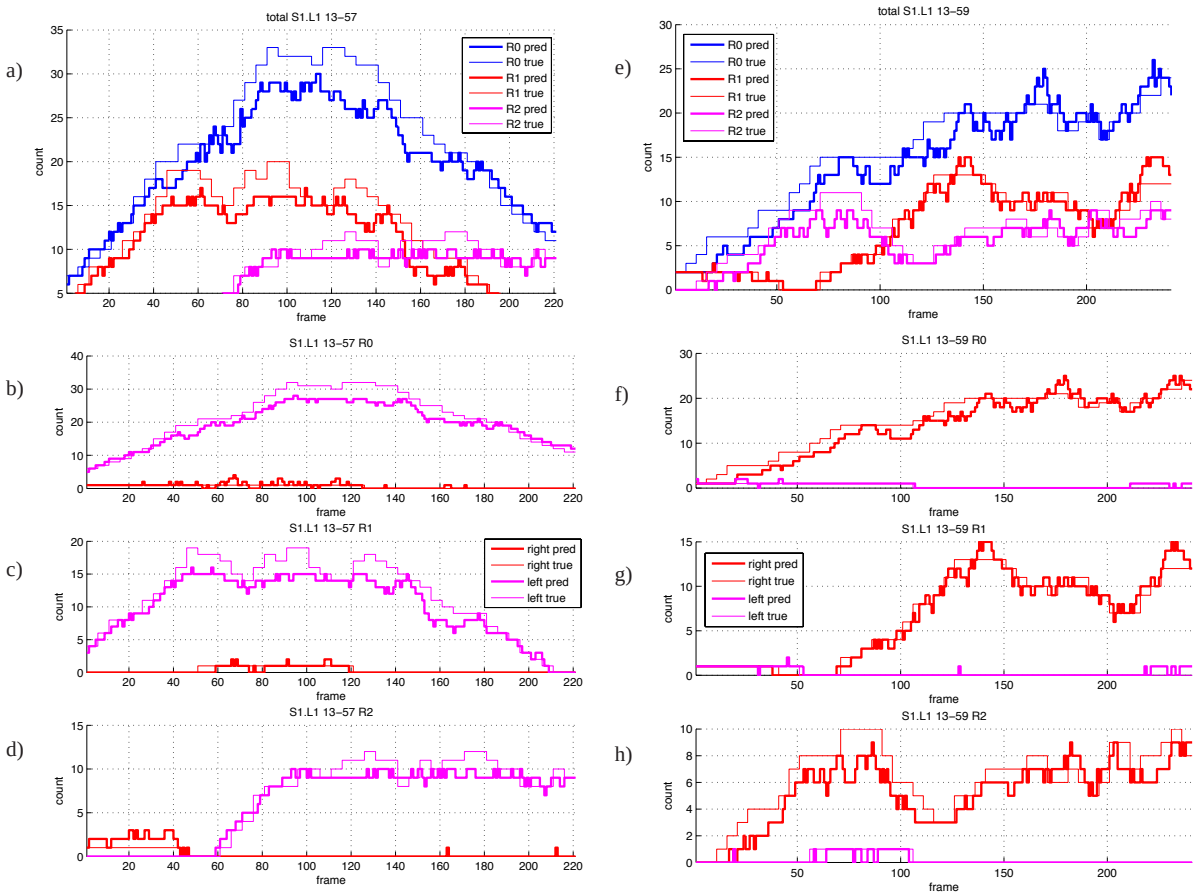

Figure 4. Count over time for the `13-57`: a) total count for R0, R1, and R2; b,c,d) right and left count for regions R0, R1, and R2; Count over time for the `13-59` video: e) total count for R0, R1, and R2; f,g,h) right and left counts for regions R0, R1, and R2.

References

[1] A. B. Chan, Z. S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[2] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 846–851.

[3] ——, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 909–926, May 2008.

[4] A. C. Davies, J. H. Yin, and S. A. Velastin, "Crowd monitoring using image processing," *Electron. Commun. Eng. J.*, vol. 7, pp. 37–47, 1995.

[5] N. Paragios and V. Ramesh, "A MRF-based approach for real-time subway monitoring," in *IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. 1034–40.

[6] D. Kong, D. Gray, and H. Tao, "Counting pedestrians in crowds using viewpoint invariant training," in *British Machine Vision Conf.*, 2005.

[7] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–714, 1986.

[8] A. N. Marana, L. F. Costa, R. A. Lotufo, and S. A. Velastin, "Estimating crowd density with minkoski fractal dimension," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, vol. 6, 1999, pp. 3521–4.

[9] ——, "On the efficacy of texture analysis for crowd monitoring," in *Proc. Computer Graphics, Image Processing, and Vision*, 1998, pp. 354–61.

[10] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[11] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *Intl. J. Computer Vision*, vol. 51, no. 2, pp. 91–109, 2003.

[12] P. Saisan, G. Doretto, Y. Wu, and S. Soatto, "Dynamic texture recognition," in *IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2001, pp. 58–63.

[13] V. N. Vapnik, *The nature of statistical learning theory*. Springer-Verlag, 1995.

[14] P. J. Moreno, P. Ho, and N. Vasconcelos, "A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, Dec 2003.

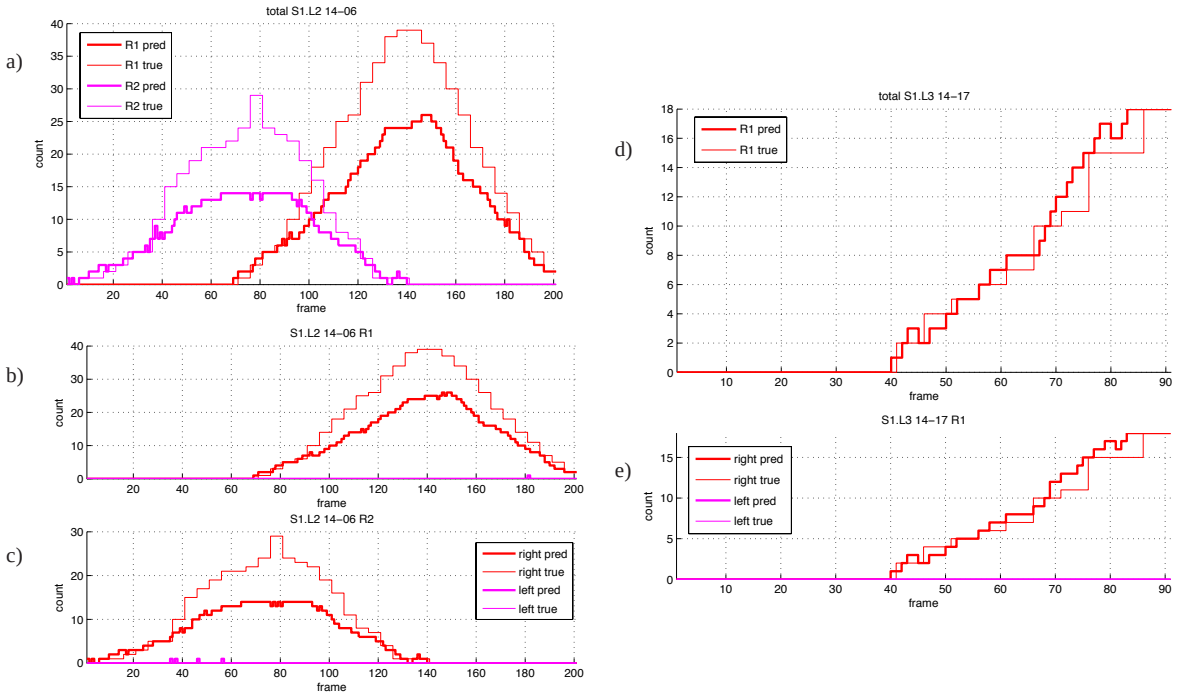

Figure 5. Count over time for the `14-06` video: a) total count for R1 and R2; b,c) right and left counts for regions R1, and R2; Count over time for the `14-17` video: d) total count for R1; e) right and left counts for region R1.

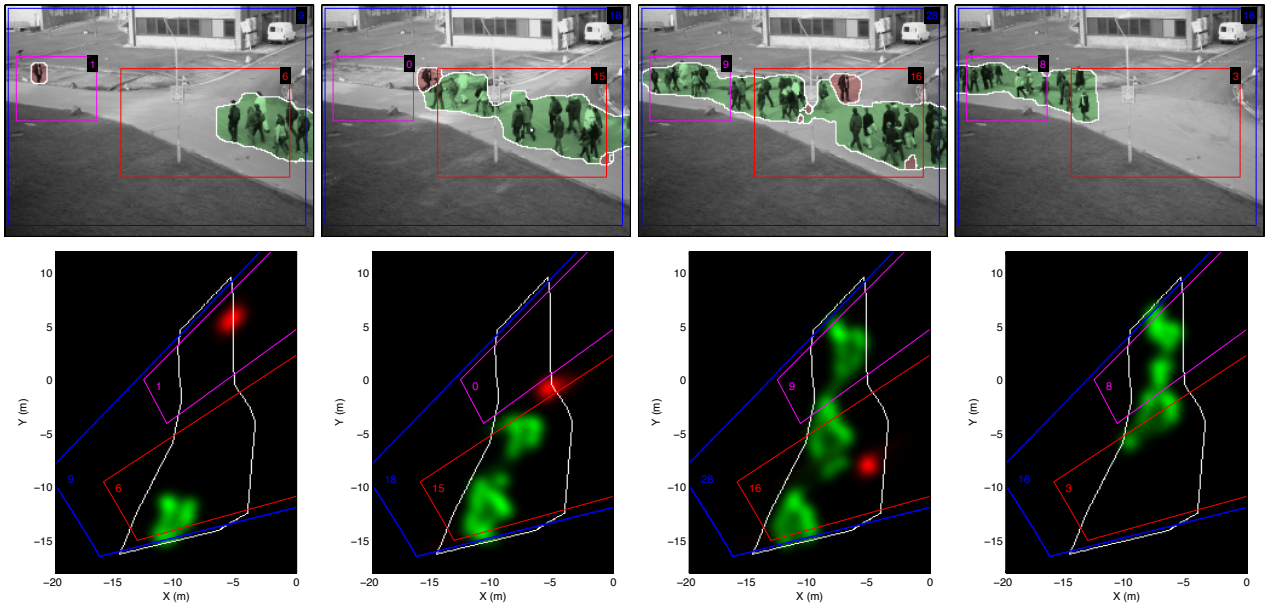

Figure 6. (top) Example frames for `13-57`, with the segmentation, regions-of-interests, and count estimates. (bottom) the occupancy maps for each frame.

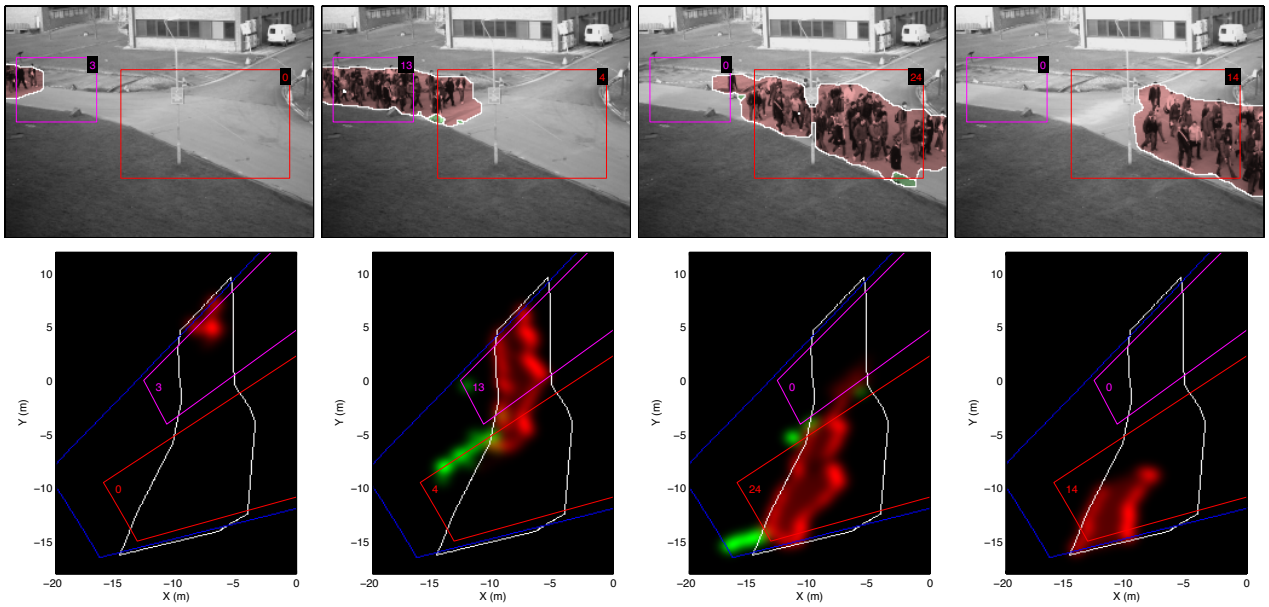

Figure 7. (top) Example frames for `14-06`, with the segmentation, regions-of-interests, and count estimates. (bottom) the occupancy maps for each frame.

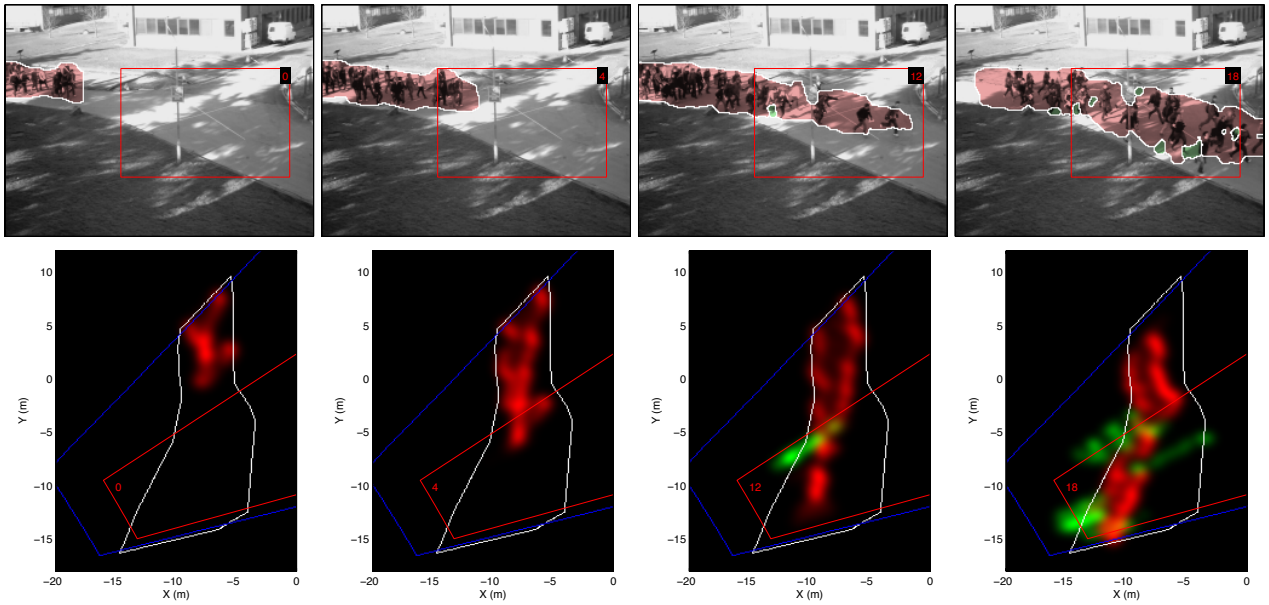

Figure 8. (top) Example frames for `14-17`, with the segmentation, regions-of-interests, and count estimates. (bottom) the occupancy maps for each frame.

Figure 9. Examples of event recognition on `14-33`. Green text indicates that the class was detected. Detection probabilities are given in parenthesis.