PEDESTRIAN COUNTING BASED ON SPATIAL AND TEMPORAL ANALYSIS

Zhongjie Yu[1], *Chen Gong*[1], *Jie Yang*[1], *Li Bai*[2]

[1]Institution of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, China
{yzjyzj, goodgongchen, jieyang}@sjtu.edu.cn
[2]University of Nottingham, UK
bai@cs.nott.ac.uk

ABSTRACT

Pedestrian counting is an important component of video processing. Existing works with overhead cameras are mainly based on visual tracking, the robustness of which is rather limited. By proposing the novel spatial-temporal matrix, this paper aims to count pedestrians without tracking. As a result, a more robust and efficient pedestrian counting algorithm can be developed. Extensive experiment reveal that our system achieves satisfying performances in terms of both accuracy and efficiency.

Index Terms— Pedestrian Counting, Video Surveillance, Spatial and Temporal Analysis, Support Vector Machine, Mean-shift Clustering

1. INTRODUCTION

Pedestrian counting is a basic requirement for intelligent transportation design. For instance, in order to optimize the schedule of a subway, the number of people entering and leaving a specific station or platform is required.

In early works, pedestrian counting is mainly based on visual tracking. To avoid occlusions, many of the tracking methods are based on overhead cameras. Velipasalar *et al.* [1] use a fast blob tracking method and the mean-shift tracking algorithm to deal with interactions among people. Antic *et al.* [2] develop a system that uses k-means clustering to enable the segmentation of single pedestrian in the scene, and establish the number of people as the maximal number of clusters. Chen *et al.* [3] use the modified overlap tracker to help on target tracking in occlusion states of merging and splitting. Although these methods perform well on videos, when applied in real-time systems, tracking is always computational expensive and lack of robustness.

In crowded situations, tracking a single person becomes difficult and it is necessary to segment different people. Zhao *et al.* [4] use human shape models to interpret the foreground in a Bayesian framework. Chan *et al.* [5] apply a Gaussian regression process to learn the correspondence between features and the number of people. Cong *et al.* [6] introduce a flow mosaicking method and an offline learning algorithm to estimate flow velocity on the detection line. Conte *et al.* [7] cluster the SURF points and estimate the number of persons per cluster. Morerio *et al.* [8] analyze small crowds relying on accurate camera calibration and the area of projection. Some works use the idea of spatial and temporal domains [9], such as clustering the trajectory of pedestrians [10, 11], or using morphological tools [12]. However, to extract the features of pedestrians from every frame in the video and process the extracted features takes tremendous amount of time.

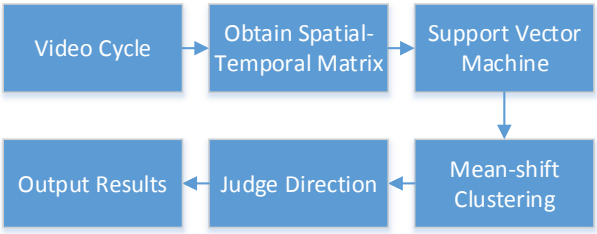

Fig. 1. The framework of our system.

Fig. 1 shows the framework of our system. First, we take a video clip and create the spatial-temporal matrix from it. Second, we apply support vector machine to classify pedestrians into two categories, crowd and individual. Then we segment the people in the crowd using the mean-shift clustering [13]. Finally we determine the direction of each pedestrian. We evaluate the performance of our system on two datasets, the SJTU dataset which is created by us and the UCSD dataset [14]. The results show that the proposed method is robust and highly accurate.

2. SYSTEM ARCHITECTURE

2.1. Spatial-temporal matrix

The proposed method creates a spatial-temporal matrix which contains the position of pedestrians and the time the pedestrians appear based on [12]. Background subtraction is applied to detect moving objects in order to create the spatial-

Corresponding author: Jie Yang, jieyang@sjtu.edu.cn;
This research is partly supported by NSFC, China (No: 61273258, 61105001).

Fig. 2. The main line (solid), the associate line (dash), and the spatial-temporal matrix.

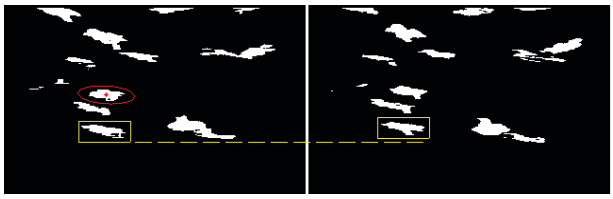

Fig. 3. Spatial-temporal matrices. left: main line, right: associate line. The frame increases from the top to the bottom of the matrix. The red ellipse is an example of covering ellipse in Sec. 2.2.

temporal matrix. Fig. 2 shows the line we draw to obtain the spatial-temporal matrix, which should be perpendicular to the flow. The solid line is the main line to collect the information of foreground and the dash one, named associated line, is utilized to help the system judge the direction of pedestrians.

At each frame, our system marks the foreground as white and the background as black. A spatial-temporal matrix consists of lines which are arranged in chronological order. Suppose w denotes width, t for the number of frame, $Frame_t$ for the t-th frame, and h for the height of the line, then the specific value in spatial-temporal matrix is

$$\boldsymbol{ST}(w,t) = \begin{cases} 0 & \text{if } \boldsymbol{Frame}_t(w,h) = 0 \\ 1 & \text{if } \boldsymbol{Frame}_t(w,h) = 255 \end{cases}. \quad (1)$$

Fig. 3 shows the spatial-temporal matrices obtained from the main line and the associate line during the first frame to the 200-th frame. In our system, the spatial-temporal matrices are obtained from every video cycle.

2.2. Support vector machine

When the pedestrians are in a crowd, it is difficult to identify each person from the spatial-temporal matrix. Regression cannot be applied because the centers of pedestrians need to be located. The proposed method classifies the connected regions from the spatial-temporal matrix into two classes, containing only one person and containing more than one person. Each pixel in the connected region has its coordinate (w,t). Let $\boldsymbol{C}$ be the covariance matrix of the connected region and $\boldsymbol{C} = \boldsymbol{\Phi\Lambda\Phi}^{-1}$, here $\boldsymbol{\Phi} = [\boldsymbol{v}_1, \boldsymbol{v}_2]$ in which $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are the eigenvectors and $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2)$ $(\lambda_1 < \lambda_2)$ is the matrix of eigenvalues.

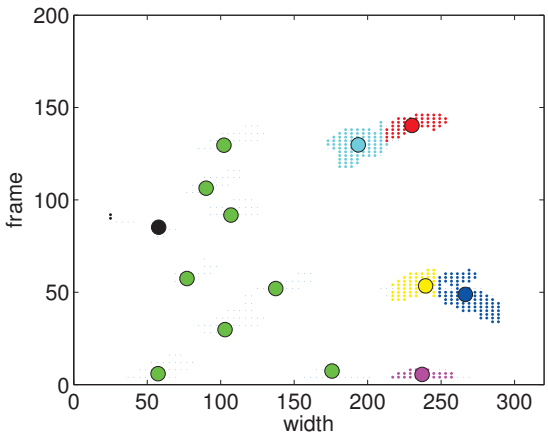

Fig. 4. Result of spatial and temporal analysis. The green dots represent the single person classified by the SVM and the dots with other colors are centers of mean-shift clustering.

Then from each connected region, 10 features are extracted and processed by the SVM. They are: (1) the area size of the connected region S; (2) the length of long axis of the covering ellipse $a = \sqrt{S \times \lambda_2/\lambda_1}$; (3) the length of minor axis of the covering ellipse $b = \sqrt{S \times \lambda_1/\lambda_2}$; (4) the length of long axis over the length of minor axis a/b; (5) the difference of the time axis; (6) the difference of the width axis; (7) the perimeter of the connected region; (8) the variance of the time axis; (9) the variance of the width axis; and (10) the perimeter over area size. One example of the covering ellipse is shown in Fig. 3.

Support vector machine classifiers are widely used in two-class classification problems. We use the LIBSVM [15] in our system. According to the above features, the connected regions are classified into two classes, individual and crowd. As shown in Fig. 4, according to the result of support vector machine, the green dots represent the centers of regions that contain only one person and the regions with other colors have more than one person in them.

2.3. Mean-shift clustering

We utilize a clustering algorithm to estimate the number of pedestrians in the crowd class. However, methods like k-means or spectral clustering cannot be applied because the number of clusters is unknown. Mean-shift clustering, which is a hierarchical method, is suitable for such situations. The mean-shift clustering method has only two inputs: the pending data and the bandwidth. In this case, the pending data are the connected regions in spatial-temporal matrix which contain more than one person. The value of bandwidth in the system can be estimated by the following procedures:

1. Collecting several spatial-temporal matrices from the training videos and labeling the pedestrians manually.
2. Increasing the bandwidth from a small value while doing the mean-shift clustering, and recording the

bandwidth b_1 when the result matches the actual value.

3. Increasing the bandwidth from b_1 until the result is above actual value by one, recording the bandwidth b_2.
4. Obtaining the proposed bandwidth of the specific spatial-temporal matrix by the mean of b_1 and b_2.
5. Obtaining the proposed bandwidth of the system bw by the mean bandwidth from all the training matrices.

The initial point $\boldsymbol{p}_s = [w_s, t_s]^T$ for clustering is chosen randomly. A new cluster S_k contains k points $\boldsymbol{p}_i$ in the connected regions which meet the equation

$$(\boldsymbol{p}_s - \boldsymbol{p}_i)^T(\boldsymbol{p}_s - \boldsymbol{p}_i) < bw^2. \tag{2}$$

The mean-shift vector is

$$\boldsymbol{M} = \frac{1}{k}\sum_{\boldsymbol{p}_i \in S_k}(\boldsymbol{p}_i - \boldsymbol{p}_s). \tag{3}$$

The procedure is iterated after the center shifts as the mean-shift vector $\boldsymbol{M}$ guides.

Fig. 4 also shows the result of mean-shift clustering on the spatial-temporal matrix from Fig. 3.

2.4. Direction determination

Our system applies a novel method to judge the direction of pedestrians by comparing the centers of them from two lines. However mean-shift clustering is an unsupervised learning algorithm, the number of pedestrians calculated from the two lines may not match. Since the centers of people in two matrices are close, we can temporarily regard the centers from the main line as that from the associate line first, then look for the foreground pixels that nearest to the centers. After the new clusters obtained and the new centers calculated, the directions of each pedestrian can be decided by the position of the corresponding centers.

When one person moves, the center of him passes two lines at different times. By analyzing the centers from two spatial-temporal matrices, we can judge the direction of the moving people. Let $\boldsymbol{ST}(w_m, t_m)$ be the central position of a pedestrian on the main line and $\boldsymbol{ST}(w_a, t_a)$ of the associate one. Then the direction of the pedestrian can be decided by

$$direction = \begin{cases} down & \text{if } t_m - t_a > 0 \\ up & \text{if } t_m - t_a < 0 \end{cases}. \tag{4}$$

Fig. 3 shows an example of direction judging. Comparing the centers of the pedestrian in the boxes, we can find that $t_m - t_a > 0$, so the man in the box is going downside.

3. EXPERIMENTS AND RESULTS

The proposed method is based on an overhead zenithal camera system. We test our system on two datasets: the SJTU dataset[1] and the UCSD dataset, shown in Fig. 5.

[1] http://www.pami.sjtu.edu.cn/people/yuzong/Video/sjtu_pami227.rar

Fig. 5. Representative frames of two adopted datasets. top: the SJTU dataset, bottom: the UCSD dataset.

Fig. 6. Illustrations of matrices to decide the bandwidth for mean-shift clustering.

3.1. The SJTU dataset

We set the camera above a corridor in a teaching building in SJTU campus, and collected 8 video clips, each lasting about 1 minute. Pedestrians walk in and out through the corridor, sometimes alone and sometimes in crowds. Our dataset contains more than 8600 frames, and a cycle is set to 200 frames during which the pedestrians are able to pass the line.

First a support vector machine classifier is trained. 24 different locations of lines are chosen to obtain spatial-temporal matrices for training, because postures of pedestrians vary on different lines. RBF kernel is chosen and cross validation is applied in each cycle to set the optimized kernel bandwidth. In order to decide the bandwidth for mean-shift clustering, four 200-frame videos are recorded for estimating. According to the spatial-temporal matrices from the 4 videos, shown in Fig. 6, the bandwidth can be estimated. The ground truth of people and the bandwidth estimated are demonstrated in Table 1.

Applying the estimated bandwidth to our system, the results of the 8 videos are shown in Table 2. Our system gives a satisfactory result and the errors are mainly caused by two reasons:

1. The proposed algorithm is not based on tracking, so when someone stays on the line for a second, the clusters obtained from mean-shift clustering will increase, so as the result of pedestrians.
2. The result of background subtraction is affected if the color of pedestrian is similar to the ground.

3.2. The UCSD dataset

The performance of our system is also evaluated on the 'vidd scene' from the UCSD dataset. The camera of the UCSD dataset is not a zenithal one, but has a tilt angle of approximately 65 degree. We still take 200 frames as a cycle. The training procedures of support vector machine classifier and the bandwidth for mean-shift clustering are omitted for

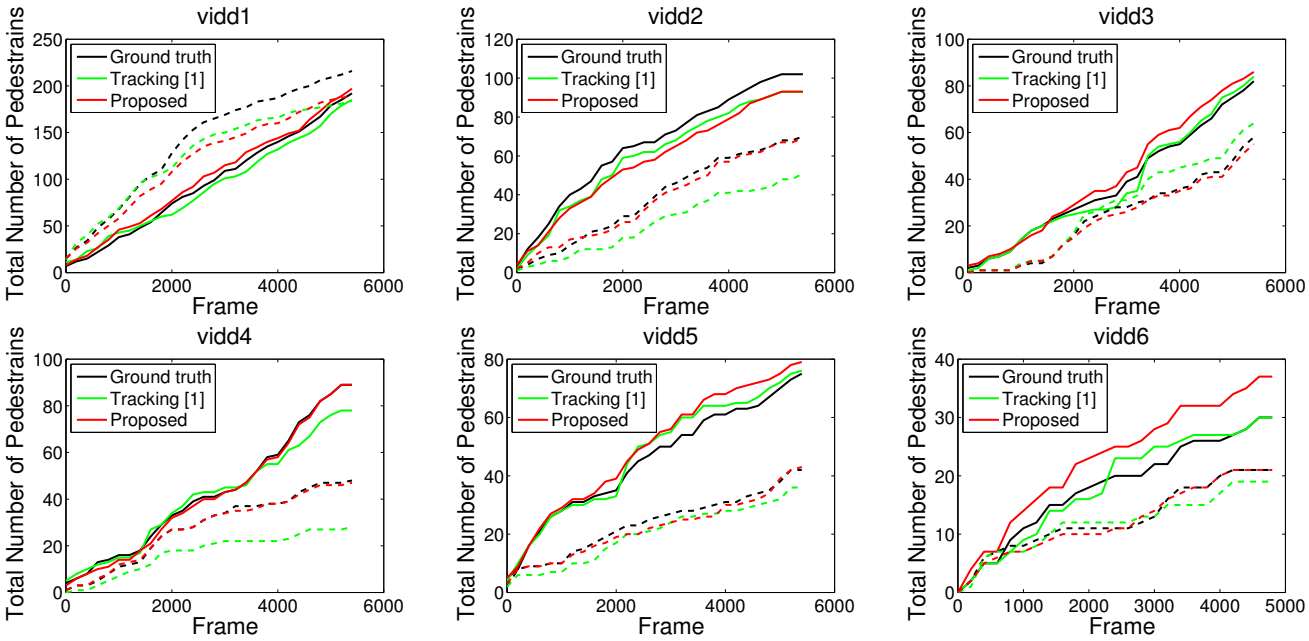

Fig. 7. Performance of our system. The solid lines represent the left direction while the dash lines represent the right direction.

Period	1	2	3	4	Mean
# of Pede.	5	9	10	3	N/A
b_1	16	20	15	14	N/A
b_2	22	25	16	29	N/A
bw	19	22.5	15.5	21.5	19.6

Table 1. Procedure to estimate the bandwidth.

Video No.	#of up	#of down	up	down	err up	err down	err total
1	15	3	15	3	0	0	0
2	12	7	13	8	+1	+1	+2
3	16	11	16	12	0	+1	+1
4	28	8	29	7	+1	-1	0
5	37	11	35	11	-2	0	-2
6	16	7	17	9	+1	+2	+3
7	13	10	13	10	0	0	0
8	8	5	11	5	+3	0	+3

Table 2. Result of the SJTU dataset. The second and third columns are the actual number of pedestrians up and down. The fourth and fifth columns are the result of our system. We show three kinds of errors in the last three columns.

brevity. Fig. 7 compares the performances of the tracking based counting [1] and the proposed algorithm.

By comparing the performance of our system with the ground truth, we conclude that the accuracy of our system is acceptable. The errors mainly come from occlusions and large moving objects such as bikes and cars.

Task	Feature Extraction	SVM	Mean-shift Clustering	Direction Judging	Total
Cost	0.15s	3.66s	0.01s	0.22s	4.04s

Table 3. Processing time of each step.

Related works test their algorithms on their own videos [1, 2, 6, 12] which are not available for us. We compare the accuracy of the proposed method on our datasets, which has a similar scene with theirs. In the scene with the tilt angle of 90 degree, the mean accuracy of our system is 96.20%, compared with the accuracy of 95% [1] and 95.45% [2]. At the degree of 65, our system achieves 95.48%, while [6] has the accuracy of 83.87% to 90.48% with the similar tilt angle.

Our algorithm is implemented in Matlab and tested on a machine with Pentium dual core 3.20GHz, 2G memory. Table 3 shows the processing time of each step. The frame rate of the test video is 18fps, so the cycle is 11.11 seconds, which is much longer than 4.04 seconds. The counting procedure can be completed before the next cycle of spatial-temporal matrix is obtained, so our system can operate in real-time.

4. DISCUSSIONS AND CONCLUSIONS

A novel spatial-temporal matrix is proposed to record the locations and time of passing pedestrians. With 10 features extracted containing the basic information of each connected region of foreground, it is fast to classify the connected regions. The direction of pedestrians is determined by the centers of pedestrians from the associate line. The proposed method only processes the data from two lines in the video, so all the steps can be conducted in one cycle and the system can operate in real-time.

5. REFERENCES

[1] S. Velipasalar, Y. Tian, and A. Hampapur, "Automatic counting of interacting people by using a single uncalibrated camera," in *Multimedia and Expo (ICME), IEEE International Conference on*. IEEE, 2006, pp. 1265–1268.

[2] B. Antic, D. Letic, D. Culibrk, and V. Crnojevic, "K-means based segmentation for real-time zenithal people counting," in *Image Processing (ICIP), 16th IEEE International Conference on*. IEEE, 2009, pp. 2565–2568.

[3] C. Chen, H. Lin, and O. Chen, "Tracking and counting people in visual surveillance systems," in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*. IEEE, 2011, pp. 1425–1428.

[4] T. Zhao and R. Nevatia, "Bayesian human segmentation in crowded situations," in *Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on*. IEEE, 2003, vol. 2, pp. II–459.

[5] A. B. Chan, Z-SJ Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on*. IEEE, 2008, pp. 1–7.

[6] Y. Cong, H. Gong, S.-C. Zhu, and Y. Tang, "Flow mosaicking: Real-time pedestrian counting without scene-specific learning," in *Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on*. IEEE, 2009, pp. 1093–1100.

[7] D. Conte, P. Foggia, G. Percannella, F. Tufano, and M. Vento, "A method for counting people in crowded scenes," in *Advanced Video and Signal Based Surveillance (AVSS), IEEE International Conference on*. IEEE, 2010, pp. 225–232.

[8] P. Morerio, L. Marcenaro, and C. Regazzoni, "People count estimation in small crowds," in *Advanced Video and Signal-Based Surveillance (AVSS), IEEE International Conference on*. IEEE, 2012, pp. 476–480.

[9] Ross Cutler and Larry S. Davis, "Robust real-time periodic motion detection, analysis, and applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 781–796, 2000.

[10] J. Xing, H. Ai, L. Liu, and S. Lao, "Robust crowd counting using detection flow," in *Image Processing (ICIP), 18th IEEE International Conference on*. IEEE, 2011, pp. 2061–2064.

[11] V. Rabaud and S. Belongie, "Counting crowded moving objects," in *Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society Conference on*. IEEE, 2006, vol. 1, pp. 705–711.

[12] A. Albiol, I. Mora, and V. Naranjo, "Real-time high density people counter using morphological tools," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 2, no. 4, pp. 204–218, 2001.

[13] Y. Cheng, "Mean shift, mode seeking, and clustering," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 17, no. 8, pp. 790–799, 1995.

[14] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 5, pp. 909–926, 2008.

[15] C. Chang and C. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, pp. 27, 2011.