Vision-based Counting of Pedestrians and Cyclists

Mehmet Kemal Kocamaz
Robotics Institute
Carnegie Mellon University
kocamaz@cmu.edu

Jian Gong
Robotics Institute
Carnegie Mellon University
michaelgongjian@gmail.com

Bernardo R. Pires
Robotics Institute
Carnegie Mellon University
bpires@cs.cmu.edu

Abstract

This paper describes a vision-based cyclist and pedestrian counting method. It presents a data collection prototype system, as well as pedestrian and cyclist detection, tracking, and counting methodology. The prototype was used to collect approximately 50 hours of data which have been used for training and testing. Counting is done using a cascaded classifier. The first stage of the cascade detects the pedestrians or cyclists, whereas the second stage discriminates between these two classes. The system is based on a state-of-the-art pedestrian detector from the literature, which was augmented to explore the geometry and constraints of the target application. Namely, foreground detection, geometry prior information, and temporal moving direction (optical flow) are used as inputs to a multi-cue clustering algorithm. In this way, false alarms of the detector are reduced and better fitted detection windows are obtained. The presented project was the result of a partnership with the City of Pittsburgh with the objective of providing actionable data for government officials and advocates that promote bicycling and walking.

1. Introduction

Object (vehicles, people, bicyclists, animals, etc) counting is one of the essential tasks in Computer Vision systems for multiple applications and domains. This is a challenging and open problem due to several factors, including illumination variations, pose changes, appearance similarities, and occlusion between multiple objects and between the objects and the environment. There are two primary approaches to this problem: (1) Region of Interest (*ROI*) counting [7, 8, 9, 17, 20, 22] and (2) and Line of Interest (*LOI*) counting [2, 10, 18, 24]. The goal of *ROI* counting is to estimate the number of the objects in a desired region of the image at a specific time. In the *LOI* counting approach, an object is counted if it crosses a virtual line of interest in the image. This paper uses the *LOI* approach, as it better matches the statistics desired by the end users of the system (City of Pittsburgh).

Figure 1: The portable Data Collection and Pedestrian/Cyclist Counting Device

In contrast to *ROI* approaches, *LOI* methods track the objects over time to produce instantaneous total count. Multi-object tracking is one of the important steps of *LOI* counting approaches. Advances of the object detection algorithms [1, 11, 12, 23, 25] in the last decade have increased the popularity and performance of the detection based tracking algorithms [6, 16, 19, 21, 26]. These trackers first run a object detector to produce a group of candidate observations at time *t*. Then, the observations are associated with the object trajectories at time *t-1*. The association step can utilize appearance and structural spatial-temporal features.

This paper proposes a vision-based pedestrian and bicyclist counting system. The presented system consists of data collection prototype and a cascaded detect-track based *LOI* counting approach. The presented approach uses only stationary monocular uncalibrated camera input. The first step in the cascade detects the pedestrians and bicyclists in the image (i.e., detects regions where a pedestrian *or* a cyclist are present, without distinguishing between them). The second step of the cascade discriminates the bicyclists from

the pedestrians. Both steps of the cascade use the infrastructure of *Benenson's* state-of-the-art object detector [4] and extended it by incorporating multiple geometry priors, including foreground detection, object size, and temporal moving direction. The tracking step follows a data association technique which utilizes the spatial proximity and temporal flow directions between the observations and the trajectories.

This work makes several contributions in the following ways:

1: The state-of-the-art object detector [4] is extended to obtain better results for the static camera input. The modifications employed in the clustering step of [4] reduce the false alarms and provides better fitted window outputs.

2: A cascaded detection system is proposed that can accurately classify and distinguish the pedestrians and bicyclists even though these have similar appearances.

3: A new pedestrian and bicyclist dataset, called *PedBike*, is introduced. This dataset consists of 50 hours of video which has total of 5 million frames. 10 hours of the dataset have been labeled. The labeled ground truths include approximately 270,000 pedestrian samples (corresponding to 541 different pedestrians) and approximately 49,000 bicyclist samples (corresponding to 111 different cyclists.)

The details of the hardware prototype and the dataset are explained in the next section. The cascaded classifier, the extensions of *Benenson's* detector [4], and the counting algorithm are described in Section 3. Experimental results of the cascaded classifier and the pedestrian/cyclist counter are analyzed in the Section 4. The last section presents conclusions and future directions for the presented work.

.

2. System Setup and Dataset

The pedestrian and cyclist counting project was a result of a real-world need from the City of Pittsburgh to determine the usage of newly-created dedicated bike lanes throughout the city. Due to the relatively large area of bike paths for which it would be desirable to obtain information, a portable data collection system was deemed the most effective solution.

2.1. Data Collection Device

The data collection device (Figure 1) consists of a ruggedized Windows tablet (Panasonic Toughpad), an extensible pole, and a miniature bullet camera. In order to collect data, the bullet camera is mounted at the top of the pole, which is extended the pole to a suitable height. The whole system is fastened to a lamp post or other sturdy city fixture. The tabled is used to verify that the camera is pointed accurately at the bike lane and to control the data collection.

Figure 2: Samples from the recorded dataset, called as *PedBike* dataset.

The system is battery powered and allows for the collection of up to 12 hours of data on a full charge.

2.2. Dataset

The dataset was collected during a period of 11 days, usually from 9 a.m. to 4 p.m., at two locations alongside a bike lane. Data was recorded at different view-points, weather, and illumination conditions. Approximately 50 hours of video were recorded. A sample of the dataset is shown in Figure 2. The data was saved in AVI encoded files each corresponding to about 20 seconds of video. 10 hours of the recorded dataset was labeled. The labeled dataset includes 541 unique pedestrians and 111 unique cyclists. In total, ground truths were generated for $\sim$270,000 separate pedestrian observations and $\sim$49,000 separate bicyclist observations.

All the videos focus on the same bike lane but in different directions. Among the 11 days, 5 days of data face the bike lane in one direction and all the other days facing in the opposite direction. Since the system is disassembled at the end of each day, differences are present in the camera height and angle from day to day. The weather varies from clear, to cloudy to rainy, often during the same day which leads to large background brightness variations throughout the dataset.

3. Method

The proposed pedestrian and bicyclist counting method follows a detect-track-count approach. The detection step of the algorithm consists of a cascaded classifier. In the first level of the cascaded classifier, the pedestrians and the bicyclists are detected. In the second level of the classifier

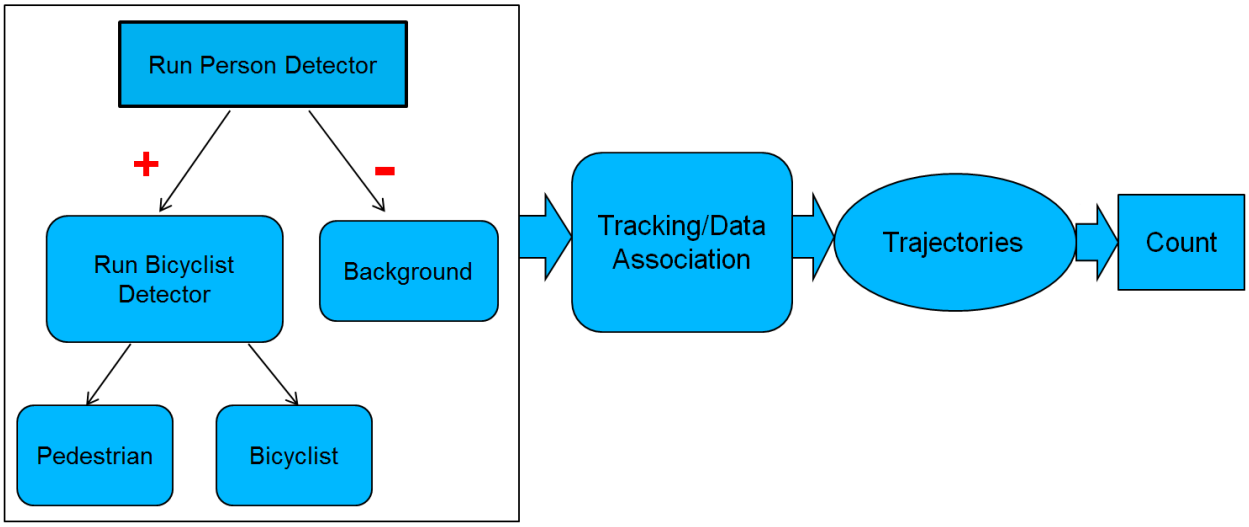

Figure 3: Overview the proposed counting approach. In the first step, a cascaded classifier detects pedestrians and bicyclists at each video frame. The subjects are tracked by associating the detections with the trajectories. In the last step, the subjects are counted if their trajectories cross a virtual line in the image.

distinguishes the bicyclists from the pedestrians. The cascaded classifier is based on the state-of-art object detector [4]. In order to reduce the false alarms of the original detector, it has been extended by incorporating additional cues. Namely, foreground segmentation, geometry prior information, and temporal moving direction are incorporated into the non-maximal suppression step of the algorithm. The final detections are obtained from by using a multi-cue clustering algorithm. The call the overall detector, including the priors and the multi-cue clustering, if referred throughout the paper as the *E*xtended Detector. A data association method employs the output of this detector to produce trajectories which are ultimately used to determine the pedestrian and cyclist counts. These cues are explained in the next sections. The overview of the proposed method can be seen in Figure 3.

Object classifiers commonly output a set of candidate windows, $S_C = \{W_1^C, ..., W_i^C, ..., W_n^C\}$, where W_i^C is a candidate window before a clustering step. W_i^C consists of the detector score, s, the top left coordinate, (x, y), width, w, and height, h, of the window in the image space. These candidate windows are clustered to output the final detections, S_D.

3.1. Foreground Detection

For a given image at time *t*, I^t, its foreground image, I_F^t is computed. The set of the number of the foreground points in the candidate windows, $S_F = \{W_1^F, ..., W_i^F, ..., W_n^F\}$, are formed by the following procedure:

Figure 4: A sample result of the foreground detection method explained in Section 3.1.

1) Using the image set from time *t* to *t-k*, optical flow is computed by Horn-Schunck's method [15].

2) A background image, I_t^B, is formed by masking out the flow points obtained in Step-1.

3) Foreground image, I_F^t, is calculated by subtracting I_t^B from I^t. Then, thresholding and region filtering operations are performed.

4) The number of the points counted in I_F^t for each candidate window in S_C to assign to S_F.

A sample computed foreground of an image can be seen in Figure 4.

3.2. Flow Direction Vector

The set of optical flow directions, $S_O = \{\hat{O}_1, ..., \hat{O}_i, ..., \hat{O}_n\}$, for each W_i^C is computed as follows:

1) For given two images, I_t and I_{t-1}, whose time stamps are t and $t-1$ respectively in the image sequences, their

Figure 5: The scale priors of the detection windows for some points are obtained from the user with the help of a software. These points are uniformly distributed in the image. The user provides the expected scale info for each green colored point in the image.

SURF [3] keypoints are computed by:

$$K_t = \mathscr{S}(I_t) \quad , \quad K_{t-1} = \mathscr{S}(I_{t-1}) \tag{1}$$

where K_t is the set of the keypoints, and $\mathscr{S}$ is the operator to obtain SURF keypoints from the image I_t.

2) The keypoints which are not in the foreground image, I_F^t, are excluded only from K_t:

$$K_t^- = \{k(x,y) \in K_t \ : \ k(x,y) \notin I_F^t\} \tag{2}$$

$$K_t = K_t - K_t^- \tag{3}$$

where $k(x,y)$ is a keypoint in the image and I_F^t is the foreground image.

3) The keypoints in K_t and K_{t-1} are matched:

$$K_{t,\,t-1} = \mathscr{M}(K_t,\ K_{t-1}) \tag{4}$$

$$\begin{aligned} K_{t,\,t-1} = \{&k_t(x_1,y_1), k_{t-1}(x_1,y_1)\}, \\ &\ldots., \\ &\{k_t(x_n,y_n), k_{t-1}(x_n,y_n)\} \end{aligned} \tag{5}$$

where $K_{t,\,t-1}$ is the set of pairs of keypoints matched from K_t to K_{t-1}, and $\mathscr{M}$ is the keypoint matching function.

5) For each pedestrian candidate window, $W^C \in S_C$, its average displacement vector at time t is calculated by:

$$\begin{aligned} K^W_{t,\,t-1} = \{&k^W_t(x_1,y_1), k_{t-1}(x_1,y_1)\}, \\ &\ldots., \\ &\{k^W_t(x_n,y_n), k_{t-1}(x_n,y_n)\} \end{aligned} \tag{6}$$

where $k_t^W(x_i,y_i) \in W^C$

$$W^C_{dis} = \frac{1}{n}\sum_{i=1}^{n} k_t^W(x_i,y_i) - k_t(x_i,y_i) \tag{7}$$

where W^C_{dis} is the displacement vector of a candidate window, W^C.

6) The unit vector of, W^C_{dis}, becomes the optical flow direction, $\hat{O}_i$, of the candidate window, W_i^C.

3.3. Geometric Priors

It is useful to have a prior information about the scales of the detection windows at different points of the scene. This prior, S_s, can be obtained through the help of a software in following way:

1) The image of the scene is divided into grids

$$I_G = \{G_1(p_1,...,p_4), G_n(p_1,...,p_4)\} \tag{8}$$

where G_i is a grid which consists of four corner points. I_G is the set of the grids in the image.

2) For each $G_i \in I_G$, expected scales at its corner points are taken from a person

$$G_i^S = \{W_1^S(w_1,h_1),...,W_4^S(w_4,h_4)\} \tag{9}$$

where G_i^S corresponds to the scales of a grid, $G_i \in I_G$. $W_i^S(w_i,h_i)$ is the scale of one corner of the grid G_i.

3) In order to compute the expected scales for the other points in a grid, bilinear interpolation, *B*, is performed using the prior scales at the corner points of the grid, G^S

$$p_i^S(w_i,h_i) = B(G^S) \tag{10}$$

where p_i^S is the scale of a point in a grid. The set of geometric prior for the candidate windows, S_s, are formed by $p_i^c(w_i,h_i)$ that is the expected width and height of a window whose center point is p_c. An illustration of this process can be seen in Figure 5.

3.4. Multi-Cue Clustering

In order to have the final detections, S_D, a non-maximal suppression based multiple cue clustering method is applied using the candidate detection windows, $S_C = \{W_1^C,...,W_i^C,...,W_n^C\}$, obtained from the detector [4]. S_C might include some false detections and outliers whose size are too big or small for the specific part of the scene. The false alarms and outliers can be removed by a filtering process. The filtering process computes the ratio between the number of the foreground points, W_i^F, in a window and its area. It removes W_i^F if the ratio is less than a threshold. In a similar way, W_i^F is filtered by calculating the ratio between its size and the prior for its center defined in S_s.

The local maximas define the detections in the non-maximal suppression algorithm. The local maximas are determined based on two cues: 1) the overlap difference, s_o

between the area of the candidate windows; and 2) the difference, s_f, between the spatial flow directions in the candidate windows. If s_o or s_f are larger than constants o_{thresh} or f_{thresh}, these candidate windows might define two different local maximas. The pseudo-code of the described method can be see in Algorithm 1.

Algorithm 1 Multi-Cue NMS

Input: * Set of candidate detections, $S_C = \{W_1^C, ..., W_i^C, ..., W_n^C\}$
Output: * Set of pedestrian detections, $S_D = \{W_1, ..., W_i, ..., W_m\}$
* Compute S_F, S_O, and S_s by the steps in Section 3.1, 3.2 and 3.3
* Find the false and outlier detections by filtering in S_F and S_s
* Remove these false and outlier detections from S_C
* Sort candidate detections, S_C, by descending order of the detector scores
for i=1, ... , length(S_C)
  for j=i+1, ... , length(S_C)
  * Compute flow direction difference score, s_f, between W_i^C and W_j^C
  * Calculate area overlap score, s_o between W_i^C and W_j^C
    if $s_o < o_{thresh}$ or $s_f < f_{thresh}$
    * Remove W_i^C from S_C
* $S_D \leftarrow S_C$

3.5. Tracking and Counting

A multiple object tracking approach is used. In order to construct the trajectories of the objects, the detection windows, S_D, at time t need to be associated to the trajectories at time *t-1*, $\mathscr{T}_{t-1} = (Wt^1, ..., Wt^i, ..., Wt^m)$, in the method where Wt^i is a detection window of this trajectory at time *i*. This *data association* process use two features: the spatial proximity, s_d, and the flow direction difference, s_f, between the last detection windows of the trajectories in $\mathscr{T}_{t-1}$ and detection windows, S_D. The window whose s_f is smaller than a threshold is associated with the trajectory whose s_d is the smallest. Since the proposed cascaded classifier already provides the subject type, the trajectories are labeled for pedestrians or bicyclists. Finally, the subjects in the image are counted if they cross a virtual line segment.

4. Experiments

Several experiments were conducted to quantify the performance of the improved person detector, cascaded classifier, proposed tracking, and counting methods.

Table 1: Comparison of the raw [4] and the extended detector which includes the modifications explained in Section 3.4. Mean Euclidean distances (in pixels) are computed between the output boxes of the detectors and the ground truth.

Metric (in px)	Raw Detector	Extended Detector
Δx	7.79	4.78
Δy	8.07	5.13
Δwidth	9.53	4.26
Δheight	11.82	5.71

4.1. Analyzing the Performance of the Extended Detector

In order to analyze the multiple cue clustering approach outlined in Algorithm 1, two sets of the experiments were performed. In the first experiment, the proposed detector was compared with a publicly available challenging dataset [5]. The experimented dataset, *TownCentre*, includes a 4500 video frames from a busy town street. It contains 230 unique pedestrians walking in two different directions and their 71,460 ground truth bounding boxes, $B(x, y, w, h)$, where x and y is the top left point, w is the width, and h the height of the box.

Pre-trained detector trained with Caltech pedestrian dataset [4], was run for *TownCentre* dataset to detect the pedestrians. The detector was set to 0.2 miss rate which produces $\sim$ 5 false positives per image. When the output of the raw detector was clustered by applying the modified algorithm explained in Section 3.4, the proposed Extended Detector obtained a false alarm rate of 1.7 per image at the same miss rate. In all experiments, o_{thresh} was set to 0.7 and f_{thresh} was set to 30 degrees. If the ratio between the area of the candidate window and the number of the foreground points is less than 0.6, the candidate window was filtered. In a similar way, if the ratios between the width and height of the candidate window and the expected window scale prior are less than 0.67 or larger than 1.33, then this candidate window is filtered.

In addition to low miss and false alarm rate, it is beneficial to have correctly well fitted bounding boxes for detect-track applications to form precise trajectories. Therefore, the output of the raw detector [4] is compared with the output of the proposed extended detector and the with the ground truth bounding boxes. The median Euclidean distance (in pixels) is computed between the center points, width, and the height of the bounding box estimated by the detectors and the ground truth. These metrics are defined as the Δx, Δy, Δwidth, and Δheight. As can be seen in Table 1, the extended detector achieved significantly better results than the original raw detector [4]. Figure 6 presents some of the detector outputs.

Figure 6: Sample results of the raw and extended detectors from *TownCentre* dataset. First column displays the output of the raw detector. Second column shows the results of the extended detector. Proposed detector produces better fitted detection windows in addition to eliminating some false alarms.

Figure 7: Some false detections of the cascaded detector. Column headings show the classification results of the detector. Low resolution of far objects, and/or partial occlusion are main reasons of the failure cases.

4.2. Performance of the Cascaded Detector

In order to train the proposed cascaded detector, the collected dataset is divided into the train, *PedBike-Train*, and the test, *PedBike-Test*, sets. *PedBike-Train* dataset consists of 40k pedestrians, 13k bicyclists, and 200k negative samples. *PedBike-Test* dataset was formed in a way that no same pedestrian or bicyclist exist in *PedBike-Train*. *PedBike-Test* includes 10k pedestrian, 5k bicyclists, and 10k negative samples. The training of the first and second levels of the cascaded detector took $\sim$ 18 hours and $\sim$ 6 hours, respectively, on a machine which has an Intel i7-3930K 6 core CPU processor and an 1536 core NVIDIA GPU.

The confusion matrix for the trained detector can be seen in Table 2. The presented cascaded detector achieved 95.1% accuracy in the classification of the samples. Total of 461 background samples were labeled as the person and 421 pedestrian samples were evaluated as the background by the detector. Only 11 pedestrians were classified as bikes in the second level of the detector which shows that it has high accuracy to distinguish between the pedestrians and bicyclists. Some false detections of the detector can be seen in Figure 7. They usually occur when the objects are far from the camera, and/or are partially occluded.

Table 2: Results of the presented cascaded detector.

		Peds	Bikes	Background
Predicted	Peds	9568	4	453
	Bikes	11	4665	8
	Background	421	331	9539

4.3. Performance of the Tracking and Counting Methods

The tracking results of the proposed method were compared with one of the state-of-the-art patch based tracker, *Fuhr's* [13, 14]. As it is suggested in these same references, the trajectories of two subjects were analyzed from *TownCentre* dataset. Median Euclidean distance in the image space between the trackers and the ground truth trajectories were measured at each frame. The results of this experiment are displayed in Table 3. The presented method significantly outperformed *Fuhr's* [13, 14] by reducing the median distance from 16.08 to 5.16 for the first subject and from 20.86 to 8.45 for the second subject. The results of the presented tracker are displayed in Figure 8.

Two experiments were conducted to analyze the perfor-

Figure 8: Results of the presented tracker, described in Section 3.5, for two subjects from *TownCentre* dataset as it is suggested in [14] [13].

Table 3: Comparison of the methods. Mean average distances (in pixels).

Test Subject	*Fuhr's* [14] [13]	Proposed Method
TownCentre-1	16.08	5.16
TownCentre-2	20.86	8.45

mance of the presented counting method. In the first experiment, *TownCentre* dataset and same pre-trained person detector, as in Section 4.1, were used to count the pedestrians. The proposed counting method achieved 94% accuracy. In the second experiment, the pedestrians and the bicyclists in ∼3 hours of the collected dataset where manually counted (which includes 149 pedestrians and 51 bikes). The same cascaded classifier trained in Section 4.2 was used. All bicyclists were counted correctly, but 9 out of 149 pedestrians were mis-counted. Figure 9 displays some counting results of *PedBike* dataset.

5. Conclusion

This paper proposed a vision based pedestrian and cyclist counting method, and an hardware prototype to collect a dataset for the training of the proposed method. This large dataset, which includes ∼ 50 hours of video, was recorded in different days, view-points, weather, and illumination conditions. Ten hours of the recorded dataset were labeled (includes ∼ 270,000 pedestrian and ∼ 49,000 bicyclist samples). The dataset was used to train the proposed algorithm, which follows the *detect-track-count* approach to count the pedestrians and cyclists if they cross a virtual line in the image. The detection step uses a cascaded classifier which extends the-state-of-the-art object detector [4] by improving its clustering stage with geometric priors (foreground detection, detection window scale information, and the temporal flow direction). The first level of the cascaded classifier detects pedestrians and cyclists, whereas its second level distinguishes cyclists. Our experiments showed that the extensions of the detector reduces the false alarms while producing better fitted windows. The presented counting method was and produced a 95% accuracy on the collected dataset.

Acknowledgement

This work was supported in part by T-SET (Technology for Safe and Efficient Transportation) Center, part of Carnegie Mellon's University Transportation Center.

References

[1] A. Bar-Hillel, D. Levi, E. Krupka, and C. Goldberg. Part-based feature synthesis for human detection. In *ECCV*, pages 127–142, 2010. 1

[2] J. Barandiaran, B. Murguia, and F. Boto. Real-time people counting using multiple lines. In *International Workshop on Image Analysis for Multimedia Interactive Services*, pages 159–162, 2008. 1

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008. 4

[4] R. Benenson, M. Omran, J. Hosang, and B. Schiele. Ten years of pedestrian detection, what have we learned? In *ECCV*, pages 613–627, 2014. 2, 3, 4, 5, 7

[5] B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *CVPR*, pages 3457–3464, 2011. 5

[6] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. *PAMI*, 33(9):1820–1833, 2011. 1

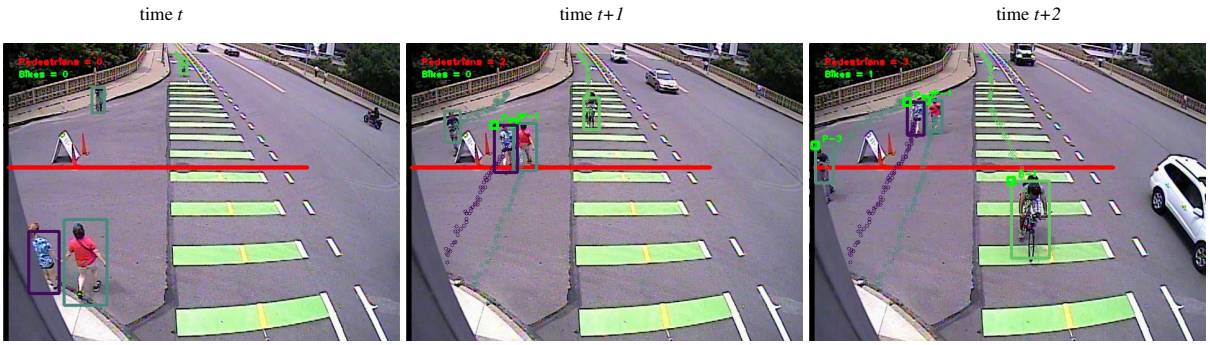

Figure 9: Sample results of the proposed cascaded counting method, explained in Section 3.5, for *PedBike* dataset.

[7] A. Chan, Z.-S. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *CVPR*, pages 1–7, 2008. 1

[8] A. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *ICCV*, pages 545–551, 2009. 1

[9] A. B. Chan and N. Vasconcelos. Counting people with low-level features and bayesian regression. *IEEE Transactions Image Processing*, 21(4):2160–2177, 2012. 1

[10] Y. Cong, H. Gong, S.-C. Zhu, and Y. Tang. Flow mosaicking: Real-time pedestrian counting without scene-specific learning. In *CVPR*, pages 1093–1100, 2009. 1

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005. 1

[12] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral Channel Features. In *Proceedings of British Machine Vision Conference (BMVC)*, 2009. 1

[13] G. Fhr and C. R. Jung. Robust patch-based pedestrian tracking using monocular calibrated cameras. In *Graphics, Patterns and Images (SIBGRAPI)*, pages 166–173, 2012. 6, 7

[14] G. Fhr and C. R. Jung. Combining patch matching and detection for robust pedestrian tracking in monocular calibrated cameras. *Pattern Recognition Letters*, 39:11–20, 2014. 6, 7

[15] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1):185 – 203, 1981. 3

[16] S. Kim, S. Kwak, J. Feyereisl, and B. Han. Online multi-target tracking by large margin structured learning. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 98–111, 2013. 1

[17] J. Li, L. Huang, and C. Liu. Robust people counting in video surveillance: Dataset and system. In *Advanced Video and Signal-Based Surveillance*, pages 54–59, 2011. 1

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

[19] A. Milan, S. Roth, and K. Schindler. Continuous energy minimization for multitarget tracking. *PAMI*, 36(1):58–72, 2014. 1

[20] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Crowd counting using multiple local features. In *Digital Image Computing: Techniques and Applications*, pages 81–88, 2009. 1

[21] V. Takala and M. Pietikainen. Multi-object tracking using color, texture and motion. In *CVPR*, pages 1–7, 2007. 1

[22] I. Topkaya, H. Erdogan, and F. Porikli. Counting people by clustering person detector outputs. In *Advanced Video and Signal Based Surveillance*, pages 313–318, 2014. 1

[23] O. Tuzel, F. Porikli, and P. Meer. Human Detection via Classification on Riemannian Manifolds. In *CVPR*, pages 1–8, 2007. 1

[24] K.-Y. Yam, W.-C. Siu, N.-F. Law, and C.-K. Chan. Effective bi-directional people flow counting for real time surveillance system. In *IEEE International Conference on Consumer Electronics*, pages 863–864, 2011. 1

[25] J. Yan, Z. Lei, L. Wen, and S. Li. The fastest deformable part model for object detection. In *CVPR*, pages 2497–2504, 2014. 1

[26] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon. Bayesian multi-object tracking using motion context from multiple objects. In *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 33–40, 2015. 1