

Text Detection using Delaunay Triangulation in Video Sequence

Liang Wu¹, Palaiahnakote Shivakumara², Tong Lu^{1*} and Chew Lim Tan³

¹ National Key Lab for Novel Software Technology, Nanjing University, Nanjing, China

² Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia

³ School of Computing, National University of Singapore

wuliang0301@hotmail.com, hudempsk@yahoo.com, lutong@nju.edu.cn and tancel@comp.nus.edu.sg

Abstract—Text detection and tracking in video sequence is gaining interest due to the challenges posed by low resolution and complex background. This paper proposes a new method for text detection by estimating trajectories between the corners of texts in video sequence over time. Each trajectory is considered as one node to form a graph for all trajectories and Delaunay triangulation is used to obtain edges to connect nodes of the graph. In order to identify the edges that represent text regions, we propose four pruning criteria based on spatial proximity, motion coherence, local appearance and canny rate. This results in several sub-graphs. Then we use depth first search to collect corner points, which essentially represent text candidates. False positives are eliminated using heuristics and missing trajectories will be obtained by tracking the corners in temporal frames. We test the method on different videos and evaluate the method in terms of recall, precision, f-measure with existing results. Experimental result shows that the proposed method is superior to existing method.

Keywords—Corner point detection, Trajectories drawing, Sub-graph formation, Motion coherence, text detection

I. INTRODUCTION

Embedded text in video sequence provides valuable information for video indexing and retrieval because text of logos, sub-titles, caption and scene texts in video sequence help in addressing the problems related to semantics gap in retrieving events [1-3]. To achieve this, text detection in video sequence with good accuracy is essential. Therefore, the topic of text detection and recognition in video has attracted increasing attention from researchers recently [4]. Text in video can be seen as two types: graphics text, which is edited text like sub-titles in news channels and scene text, which is part of video and generally appears on logos, banners, billboards, trucks' bodies etc. Since graphics is edited, it is easy to process whereas scene text is part of video, which poses lots of challenges such as orientation, font, font size, contrast, color variation etc. In addition, text movements in complex background make text detection problem more complex [1-4].

The literature on text detection in video can be seen as two categories at high level, namely methods [5, 6] for static text detection (graphics/caption) and non-static text detection (scene text) [7]. The first category can be classified further as methods without temporal information (individual frames) and with temporal information [8]. Again, methods without temporal information can be classified as connected component based [9], texture based and gradient based methods. Connected components based methods are good for big font text and caption text that share uniform color. Texture based

methods [10-12] are good for detecting text from complex background images. Edge and gradient based methods [13, 14] are fast compared to texture based methods but they are sensitive to background. The main drawback of these methods is that the scope is limited to horizontal and graphics text detection but not scene text detection. There are methods [15-18] which detect both graphics and scene text of any orientation. However, these methods do not utilize the temporal information available in video rather they use individual frames for text detection. Similarly, the methods use temporal information for static text detection, based on the fact that the text appears in video at the same location for a few seconds [5, 6, 19-21]. Therefore, these methods can be used for caption or graphics text detection but not scene text detection as the latter is not static. To detect scene text, the methods [7, 8] use motion vector analysis in temporal frames with spatial information or direction. However, these methods do not perform well for graphics text because graphics text is not dynamic. Thus, there is a demand for developing a method detecting both graphics and scene text by exploring temporal information.

Hence, in this paper, we propose a hybrid method that explores both characteristics of graphics and scene text by estimating trajectories between corners of text, namely spatial proximity, motion coherence, local appearance and canny rate. We are inspired by the work presented in [22] for tracking person in crowd, where it shows that these features are effective in tracing a person moving in complex environment. To the best of our knowledge, these features are explored the first time for text detection in video. Then we propose to use Delaunay triangulation [22] to obtain graph for the trajectories over a time. The depth first search is proposed to cluster trajectories which represent text. The advantage of this method is that it works for both static (graphics) and dynamic text (scene) of any orientation and script in video.

II. PROPOSED METHOD

The proposed method is divided into three sub-sections. The first section introduces features which represent text pixels based on trajectory estimation between the corners. Cluster density and Delaunay triangles are used for identifying text candidate regions in the second section. The third section merges all candidate text regions to extract a text line. For each frame, we extract corner points in the current frame to identify location of text region. As it is noted that text generally has dense corners with close proximity, we propose to use corners for estimating trajectories in this work. Besides, it is found from [6] that corners for text detection are effective in videos. Then, we use KLT tracker to generate a set of trajectories of

corner points within a specified time window of the video sequence as in [22] where KLT is used to trace a person in the crowd. The length of time window is named τ , which means the number of frames in the video sequence. We track those corner points of the current frame respectively in the previous $\tau/2$ frames and the later $\tau/2$ frames. So we get the trajectories of every corner point in the current frame. Then the graph is constructed using a set of trajectories. Each node of the graph corresponds to a separate trajectory. Delaunay triangulation is proposed to consider trajectories as edges to form a graph. In this way, we get connected graph for all corners in a frame based on trajectories as shown in the first illustration in Fig. 1, where one can see that a graph is formed with not only corners of text but also others. This graph may contain edges which belong to non-text component and edges may connect across lines. Therefore, to eliminate such false edges, we propose four criteria to prune those edges step by step. An edge connecting two nodes, say, p and q , will be retained only if it meets all the following four criteria: (1) spatial proximity, (2) motion coherency, (3) temporal variation of local appearance and (4) canny rate. As a result, we get several small sub-graphs as shown in the second illustration in Fig. 1, where two characters have two sub-graphs correctly. We then use depth first search to combine clusters of corner points which represent text characters. However, these regions may include some non-text region also due to complex background. Therefore, we propose two rules to verify these regions to wipe out the non-text regions. At last, we merge those clusters belonging to the same text line as shown in the third illustration in Fig. 1, where text box is fixed for two characters. The same thing is illustrated on real frames in Fig. 2, where the first image from top left is the input, the second image from top right shows the initial graph given by the Delaunay triangulation based on trajectories of corners of text, the third image from bottom left shows the results of pruning, where all non text edges and connecting across lines are removed, and the fourth image from bottom right shows the final text detection result where it can be seen that all text lines are detected properly.

A. Pruning edges via four criterions

In this stage, we define four criteria to prune the edges.

Firstly, calculate the spatial proximity of two connected trajectories. We believe that if two trajectories belong to the same text line region, they will remain in close proximity. So the length of the edge connecting them will be small in every frame of the time window. We use the maximum displacement between a pair of corner trajectories within a time window to represent spatial proximity. It is defined as

$$Sp_{p,q} = \max(len_{p,q}(t)) \quad t \in [t_s, t_s + \tau - 1] \quad (1)$$

where $Sp_{p,q}$ is the value of spatial proximity between node p and q , $len_{p,q}(t)$ is the length of edge between nodes p and q at frame t , and t_s is the starting frame within the time window. If spatial proximity of the two connected trajectories is too large, then the edge connecting them will be pruned. This eliminates some of the false edges.

Secondly, calculate the motion coherency of two connected trajectories. Since if two nodes of trajectories belong to the same text line region, the motion of them will be alike, that is

to say the speed and direction of their motion will be approximate. In another word, the distance between these two nodes will always be the same within the time window. So we use the standard deviation of the distance between two trajectories over the time window to measure the dissimilarity with respect to motion coherency:

$$Cm_{p,q} = \sqrt{\frac{1}{\tau} \sum_{t_s}^{t_s+\tau-1} (len_{p,q}(t) - \overline{len_{p,q}})^2} \quad (2)$$

where $Cm_{p,q}$ is the value of motion coherency between node p and q , $\overline{len_{p,q}}$ is the mean value of length of edge between nodes p and q within the time window. If motion coherency between two connected trajectories node is too large, then this edge connecting them will be pruned. With this criterion, we are able to distinguish those two text lines, which overlaps with each other, but whose motion status is not the same as shown in Fig. 3 where gradually two text lines are separated as time goes.

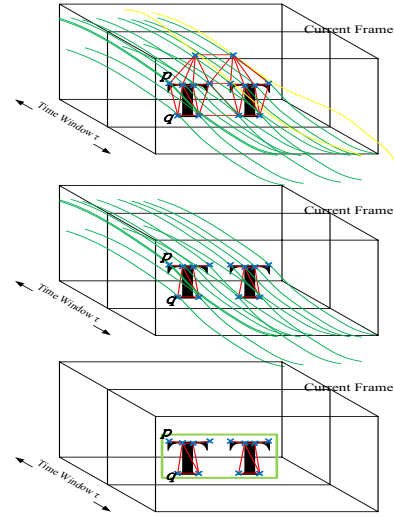


Fig. 1 The initial graph with trajectories, edges pruned graph and the text line detected.



Fig. 2 The process of text detection in a frame of image.



Fig. 3 Two text lines having overlap region but with their motion status not the same are distinguished.

Thirdly, calculate temporal variation of local appearance. As we know, if text line region in a video remains in the same shape and size, it is just like a rigid thing and its appearance remains the same during a period of time. So we utilize the temporal variation of local appearance to describe this property. In our method, the temporal variation of local appearance is measured to quantify the change in appearance of a sequence of small triangular patches bounded by three adjacent trajectories. Here we use the temporal variation of a hue-saturation color histogram of the Bhattacharyya distance between the color histogram $h(t)$, and the average color histogram \bar{h} of the patch within the time window is used to define the dissimilarity $Tv_{p,q}$ for a sequence of patches as

$$Tv_{p,q} = \sqrt{\frac{1}{\tau} \sum_{t_s}^{t_s+\tau-1} d_{hist}^2(h(t), \bar{h})} \quad (3),$$

where $Tv_{p,q}$ is the value of temporal variation of local appearance and $d_{hist}(\cdot, \cdot)$ is the Bhattacharyya distance between two histograms. Since each edge can potentially be shared by up to two adjacent patches, we chose the largest dissimilarity to represent the value of temporal variation of local appearance. If this value is larger than a threshold, we will prune this edge. The effect of this criterion can be seen in Fig. 4, where we can see the three triangles marked by red color appear the same.

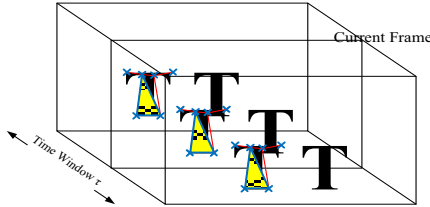


Fig. 4 Temporal variation of local appearance is measured to quantify the change in appearance of a sequence of small triangular patches bounded by three adjacent trajectories.

Fourthly, calculate canny rate for each edge between two connected nodes. Essentially, the previous three criteria used to prune edge between trajectory nodes are about the consistency of spatial proximity, motion status and local appearance within the time window. Sometimes, due to complex background, non-text region may satisfy the above three criteria. Therefore, we design one more criterion which is used to further prune some more edges in the graph and only retain those represent text regions. We calculate a canny map for the current frame. As we referred previously, there exist dense and orderly presence of corner points in characters, especially at the turning of their strokes. And these corner points always occur around a canny line. We want to retain those edges in the graph whose two endpoints are the corner points at the turning of a character's strokes and near a canny line, enabling us to store the structure of a character in the pruned graph. We then perform dilation operation on the canny map to make the canny line a little thicker to assure that the corner points near the original canny line are now in the dilated canny line. After the dilation, we calculate the canny rate over the time window for each edge in the graph as shown in Fig. 5, where the first image in the two examples shows canny of text line, the second one gives the results of dilation,

the third illustrates the results of pruning graph and the fourth one gives the results of text detection.

We define the canny rate as

$$Cr_{p,q} = \frac{\sum_{t=t_s}^{t_s+\tau-1} |E_{p,q}(t) \cap C_{p,q}(t)|}{\sum_{t=t_s}^{t_s+\tau-1} |E_{p,q}(t)|} \quad (4)$$

where $Cr_{p,q}$ is the canny rate of edge between node p and q . $E_{p,q}(t)$ and $C_{p,q}(t)$ respectively represent the set of pixels belonging to edge between nodes p and q , and the set of pixels belonging to the canny map within the time window. The notation $|S|$ means the cardinality of the set S . After calculating the canny rate, we further prune some of those edges whose canny rate value is too large.



Fig. 5 Two samples of using canny rate criterion. Four images are respectively canny map, dilated canny map, the graph which stores the structure of characters and the final text line detected.

B. Candidate text region verification

After the four steps of pruning edges in the graph, the initial connected graph becomes several groups of sub-graphs. We use deep first search to cluster these corner points. Those clusters which consist of only one corner point will be deleted directly, because a single point of pixel cannot possibly form a character. Then the remainder clusters are all small connected sub-graphs which have more than one corner points with edges not pruned connecting them. These clusters are called candidate text region. We accordingly calculate the minimum bounding box for each cluster according to its constituent corner points.

We define two rules to wipe out some clusters in which the edges connecting their corners are not that dense and arranged orderly as follows:

Rule 1: Cluster edge density is defined to represent text density in the bounding box of the cluster. Cluster edge density means the density of the remainder edge pixels in the bounding box. It is defined as

$$Cd = \frac{Nep}{C_w \times C_h} \quad (5)$$

where Cd is cluster edge density, C_w and C_h are the width and height of the bounding box, Nep is the number of the remainder edge pixels. After calculating the Cd , we compare it with a threshold CdT . If it is smaller than CdT , we will abandon this cluster as text region.

Rule 2: As we use Delaunay triangulation to construct the initial graph of the corner points, the sub-graph of the true text line

region will be a similar Delaunay triangulated graph of its containing corner points. We use the term cluster edge strength to represent this property, and it is defined as

$$Cs = \frac{Nce}{Nde} \quad (6)$$

where Cs is cluster edge strength, Nce is the number of the cluster's containing edges, and Nde is the number of edges which is formed through Delaunay triangulation according to containing corner points. If Cs is smaller than the threshold CsT , we also abandon this cluster, and consider it is not likely to be a text region.

C. Merging clusters to form text line

The objective of this section is to merge all clusters to extract text line. If the gap between two characters in a text line is small, the method will regard every character as a separate cluster of text region, because the canny rate of the edges crossing these two characters will not be high. This outputs full text line.

The process increasingly adds a cluster into the merged cluster until the set of clusters is empty. We firstly choose a cluster whose width and height is very close to be an original merged cluster. Then we select the nearest cluster to be merged with. After this step, there are at least two characters in the merged cluster. Next, we use the information of the angle and center of the merged cluster's bounding box to add the remaining clusters. If there is no such an original cluster with close width and height, we directly use the angle and center information of one of the clusters to merge other clusters. Fig. 6 shows the process of merging clusters to obtain text line based on angle information of text clusters.

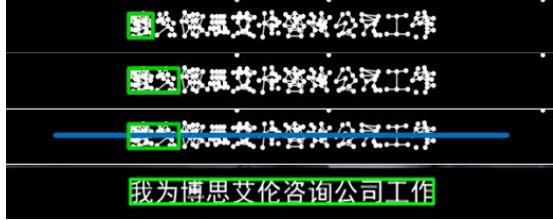


Fig. 6 The process of merging clusters

III. EXPERIMENTAL RESULTS

We collect video data from different sources for both horizontal and non-horizontal text detection, which includes 500 video of 1 to 3 seconds. This dataset consists of video of low resolution, different fonts, font sizes, scripts, orientation, background etc. The proposed method is evaluated in terms of Recall (R), Precision (P), F-measure (F) and Average Processing Time (APT). Average processing time is defined as the sum of processing time of all the frames divided by the number of frames. The definitions and the way we count the number of text blocks to compute recall, precision, f-measure can be found in [18]. To show the proposed method is effective, we compare it with the existing methods in [19, 20]. The reasons for choosing these existing algorithms are the following. The methods use temporal information for text detection as in our method and the method [19] uses motion features for multiple frame verification. However, the algorithm in [19] focuses on graphics and horizontal text but

not non-horizontal and scene texts, and the method in [20] explores edge density and similarity measures to detect text while the proposed method presents triangle formation based on trajectories of the corners of text. In addition, the proposed method detects non-horizontal text detection and scene text.

A. Experiments for Text Detection

The qualitative results of the proposed and the existing methods are shown in Fig. 8, where (a) shows an input frame having different scripts and background, graphics text, scene text with different scripts and non-horizontal text, (b) shows that the proposed method detects almost all texts in the input frames including non-horizontal text in the third frame in (a), while the existing methods do not detect text properly because of the lack of discriminative features for detecting both graphics and scene texts. It is observed from Fig. 8 that the existing methods do not detect scene text in the second frame while the proposed method detects well. The quantitative results of the proposed and the existing method are reported in Table I, where the proposed method gives high recall, precision, F-measure and low average processing time in seconds compared to existing methods. Bouaziz et al.[20] is better than Huang et al.[19] in terms of recall and f-measure because Bouaziz et al. propose quad tree technique to identify the correct text candidates based on edge density of text, while Huang et al. [19] propose some heuristics based on connected component analysis for identifying text candidates. However, since quad tree used in [20] requires more computations, it is expensive compared to the method in [19]. On the other hand, the proposed method is better than both the existing methods because of advantage of four pruning criteria proposed based on temporal information and it does not involve all 30 frames. Thus the proposed method is less expensive compared to the existing methods.

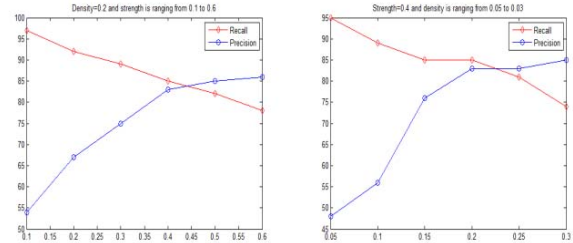


Fig. 7 The value strength and density influence recall and precision.

We chose 100 frames from our dataset and use them to set the value of edge strength threshold which is required for rule 1 and edge density threshold which is required for rule 2 presented in Section II.B to verify text candidates, respectively. Firstly, we set density threshold to be 0.2 and range strength threshold from 0.1 to 0.6 as shown on the left side in Fig. 7, where recall and precision curve influence to fix an optimal threshold. In the same way, we set strength threshold to be 0.4 and range strength threshold from 0.05 to 0.03 as shown on the right side in Fig. 7, where the experiment influences to fix an optimal threshold for edge strength. Finally, we respectively set density threshold 0.22 and strength threshold 0.45 as shown in Fig. 7(left) and Fig. 7(right).

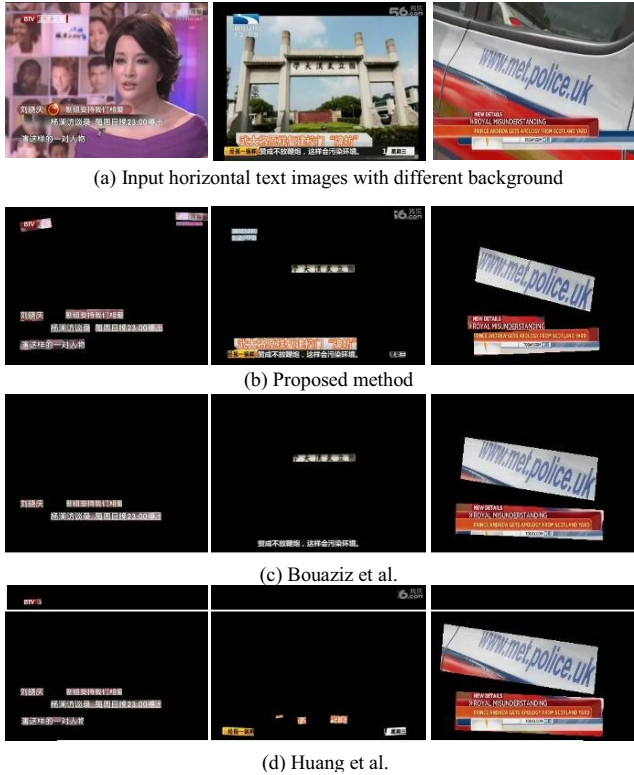


Fig. 8 Proposed and existing methods for horizontal text detection.

TABLE I. PERFORMANCE OF THE PROPOSED AND EXISTING METHODS FOR TEXT DETECTION (IN %)

Method	R	P	F	APT(seconds)
Proposed Method	85	83	84	0.98
Bouaziz et al. [20]	45	56	49	4.0
Huang et al. [19]	30	75	43	1.6

IV. CONCLUSION AND FUTURE WORK

In this paper, we propose a new method for detecting both graphics and scene texts in video sequence. This method explores temporal information by estimating trajectories for the corners of texts. Delaunay triangulation is used to obtain graphs from the trajectories between corners. The proposed pruning criteria eliminate false edges in the Delaunay triangles, which connect corners across text lines and non-text region. The four pruning criteria are derived based on spatial proximity, motion coherence, local appearance and canny rate, respectively. To verify the text region, the method proposes heuristics using the edge strength and area of text region. The identified regions are merged to extract text lines using angle information of text clusters. To the best of our knowledge, the method is the first to detect both graphics and scene texts in video sequences using temporal information. In the future, we are planning to explore a few more features using motion vector analysis to improve the accuracy of large video dataset.

ACKNOWLEDGMENTS

The work described in this paper was supported by the Natural Science Foundation of China under Grant No. 61272218 and 61321491, the 973 Program of China under Grant No. 2010CB327903, and the Program for New Century Excellent Talents under NCET-11-0232.

REFERENCES

- [1] J. Zang and R. Kasturi, "Extraction of Text Objects in Video Documents: Recent Progress", In Proc. (DAS), pp. 5-17, 2008.
- [2] K. Jung, K. Kim and A. K. Jain, "Text information extraction in images and video: a survey", PR, pp, pp 977-997, 2004.
- [3] D. Doermann, J. Liang and J. Li, "Progress in Camera-Based Document Image Analysis", In Proc. CDAR, pp 606-616, 2003.
- [4] K. Jung, "Neural network-based text location in color images", PRL, pp. 1503-1515, 2001.
- [5] A. Mosleh, N. Bouguila and A. B. Hamaz, "Automatic Inpainting Scheme for Video Text Detection and Removal", IEEE Trans on IP, pp 4460-4472, 2013.
- [6] X. Zhao, K. H. Lin, Y. Fu, Y. JU. Y. Liu and T. S. Huang. "Text From Corners: A novel Approach to Detect Text and Caption in Videos." IEEE Trans. on IP, pp 790-799, 2011.
- [7] H. Xiaodong. "A novel approach to detecting scene text in video." In Proc. CISP, pp 469-473, 2011.
- [8] R. Wang, W. Jin and L. Wu, "A Novel Video Caption Detection Approach using Multi-Frame Integration", In Proc. ICPR, 2004
- [9] K. Jung, K. I. Kim and A. K. Jain. "Text information extraction in images and video," Pattern Recognition, 2004, pp 977-997.
- [10] L. Huiping, D. Doermann, and O. Kia. "Automatic text detection and tracking in digital video." IEE Trans. on IP, pp 147-156, 2000.
- [11] K. L. Kim, K. Jung and J. H. Kim, "Texture-Based Approach for TextDetection in Images using Support Vector Machines and Continuous Adaptive Mean Shift Algorithm", IEEE Trans. on PAMI, pp. 1631-1639, 2003.
- [12] P. Shivakumara, T. Q. Phan and C. L. Tan, "New Wavelet and Color Features for Text Detection in Video", In Proc. ICPR, 2010, pp 3996-3999.
- [13] C. Liu, C. Wang, and R. Dai, "Text detection in images based on unsupervised classification of edge-based features," In: Proc. ICDAR, pp 610-614, 2005.
- [14] E. K. Wong, and M. Chen "A new robust algorithm for video text extraction," Pattern Recognition, 2003, pp. 1397-1406.
- [15] P. Shivakumara, T. Q. Phan and C. L. Tan, "A Laplacian Approach to Multi-Oriented Text Detection in Video", IEEE Trans. PAMI, pp. 412-419, 2011.
- [16] N. Sharma, P. Shivakumara, U. Pal, M. Blumenstein and C. L. Tan, "A New Method for Arbitrarily-Oriented Text Detection in Video, In Proc. DAS, pp. 74-78, 2012.
- [17] Y. Liu, Y. Song, Y. Zhang and Q. Meng, "A Novel Multi-Oriented Chinese Text Extraction Approach from Videos", In Proc. ICDAR, pp 1387-1391, 2013.
- [18] P. Shivakumara, H. T. Basavaraj, D. S. Guru and C. L. Tan, "Detection of Curved Text in Video: Quad Tree based Method", In Proc. ICDAR, , pp 594-598, 2013.
- [19] X. Huang, H. Ma and H. Yuan, "A novel video text detection and localization Approach", In Proc. PCM, 2008, pp 525-534.
- [20] B. Bouaziz, T. Zliti and W. Mahdi, "AViText: Automatic Video Text Extraction" CoRR abs/1301.2173, 2013.
- [21] D. Chen and J. M. Odobez, "Video text recognition using sequential Monte Carlo and error voting methods", PRL, pp 1386-1403, 2005.
- [22] D. Sugimura, K. Kitani, T. Okabe, Y. Sato, and A. Sugimoto, "Using Individuality to Track Individuals: Clustering Individual Trajectories in Crowds Using Local Appearance and Frequency Trait," In Proc. ICCV, pp. 1467-1474, 2009.