A TRAJECTORY CLUSTERING APPROACH TO CROWD FLOW SEGMENTATION IN VIDEOS

Rahul Sharma, Tanaya Guha

Electrical Engineering, Indian Institute of Technology Kanpur, India

ABSTRACT

This work proposes a trajectory clustering-based approach for segmenting flow patterns in high density crowd videos. The goal is to produce a pixel-wise segmentation of a video sequence (static camera), where each segment corresponds to a different motion pattern. Unlike previous studies that use only motion vectors, we extract full trajectories so as to capture the complete temporal evolution of each region (block) in a video sequence. The extracted trajectories are dense, complex and often overlapping. A novel clustering algorithm is developed to group these trajectories that takes into account the information about the trajectories' shape, location, and the density of trajectory patterns in a spatial neighborhood. Once the trajectories are clustered, final motion segments are obtained by grouping of the resulting trajectory clusters on the basis of their area of overlap, and average flow direction. The proposed method is validated on a set of crowd videos that are commonly used in this field. On comparison with several state-of-the-art techniques, our method achieves better overall accuracy.

Index Terms— Crowd Flow Segmentation,Trajectory Extraction, Trajectory Clustering, DBSCAN

1. INTRODUCTION

With the increasing security threats across the world, the need for video surveillance in crowded places is growing faster than ever. In highly populous countries, like India, even crowd mismanagement has caused loss of human lives in several occasions [1, 2]. Automatic monitoring of crowded areas is hence important for securing public safety, and for better management of events involving a large crowd, such as, rallies, festivals and sports matches.

A crucial step in such monitoring systems involves identifying the dominant motion patterns and directions of crowd flow. Although motion-based video segmentation is a well studied problem, the majority of standard object detection-based tracking methods does not perform well on high density crowd videos [3]. This is due to the complex dynamics of the crowded scenes, and the presence of a large number of very small objects, which are difficult to track simultaneously (see Fig. 1 for examples).

In the recent past, several methods have been developed to address the problem of identifying (and later, segmenting) the dominant coherent flow patterns in high density crowd videos [4, 5, 6, 7, 8, 9, 10]. A supervised machine learning method [4] has proposed to train an offline system that can identify certain classes of crowd behavior. This pretrained system is used to label each patch in a given test video, based on its motion pattern. Among the unsupervised methods, a region growing technique has been proposed to segment crowd flows based on an optical flow field [5]. Another approach has addressed the problem from the perspective of scattered motion field segmentation, where the optical flow is computed only at salient locations, and a local translational domain segmentation model is developed [6].

Fig. 1. Sample frames from high density crowd videos.

A fluid dynamics perspective has also been successfully adopted to address the problem of crowd flow segmentation [7]. This method has used finite time Lyapunov exponent field to detect the Lagrangian coherent structures (LCS) present in the underlying flow of crowd. These LCS are then used to locate the flow boundaries. More recently, researchers have attempted to solve this problem by directly using the motion vector information from the compressed video files [8, 9]. One such approach involves clustering the motion vectors using a standard expectation-maximization (EM) algorithm, and finally merging the clusters to a single flow [9]. Following the similar line of research, another work has modeled the motion vectors as conditional random fields (CRF) [8], and the motion segments are obtained by labeling the motion vectors such that the global energy of the CRF model is minimized.

In this paper, we develop a trajectory clustering-based approach for high density crowd flow segmentation. Following an unsupervised paradigm, our approach exploits the rich temporal information contained in trajectories, which is expected to capture the complex motion patterns better. Our method relies on tracking video regions (blocks) to extract *block-based* trajectories. Since object detection and tracking is not practical in the high density videos, and tracking individual points is too noisy, we consider tracking a mid-level structure (block) in a video sequence. The proposed trajectory clustering algorithm takes into account the shape, location, and the neighborhood density of trajectory patterns. Our segmentation results are compared with the state-of-the-art methods on a set of benchmark video sequences. The performance of the proposed approach is evaluated using both visual and objective measures (Jaccard similarity), and superior results are obtained.

2. THE PROPOSED APPROACH

Motion trajectories are an effective way to capture the complex temporal dynamics in a video scene. Hence, we cast the problem of crowd flow segmentation as a trajectory extraction and clustering task. Our method is divided into the following steps: (i) *Trajectory extraction*: detection of tracking blocks or regions in a video to extract the trajectories. (ii) *Trajectory clustering*: development of a clustering algorithm, particularly suited for high density trajectories, that uses shape, location and the density of the trajectory patterns in

a neighborhood. (iii) *Flow segmentation*: Labeling each pixel based on their motion pattern to generate the final motion segments. Each step is described below in detail (see Fig. 2 for result at each step).

2.1. Trajectory extraction

The videos under consideration are densely populated with the moving objects, very small in size. For such videos, standard object detection and tracking algorithms perform very poorly due to the presence of large number of small objects that often get occluded [3]. So, instead of focusing on objects, we concentrate on tracking blocks in a video frame. We divide a frame into non-overlapping blocks of size $p \times p$. For each block, a set of interest points are detected using Harris corner detector [11]. The centroid of the interest points associated with each block is tracked using the standard Kanade-Lucas-Tomasi (KLT) tracking algorithm to obtain a trajectory for each block.

In a high density crowd video, new moving objects keep appearing into the video frame over time. In order to account for these newly appearing objects, we refresh our existing set of blocks after every r number of frames. The blocks which had gone out of the frame are removed and the ones which have the newly appearing objects, are added to the feature tacker.

Only the trajectories of length greater than a chosen threshold τ are retained (see Fig. 2(b) for extracted trajectories).

2.2. Trajectory clustering

A new trajectory clustering algorithm is developed in this section. The proposed algorithm exploits the shape and location information of the trajectories along with their neighborhood density.

2.2.1. Trajectory representation

Consider a trajectory $T = \{(x_s, y_s), (x_{s+1}, y_{s+1})..., (x_e, y_e)\}$, where (x_j, y_j) represent the coordinates of the trajectory at j^{th} frame, and s and e denote the start and end frame.

Shape: To capture the spatiotemporal shape of a trajectory, we model it separately along x and y dimensions, using third order polynomial functions of time.

$$x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (1)$$

$$y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 \quad (2)$$

where $t \in (1, l)$ and $l = e - s$. The shape of a trajectory is thus represented by its *shape feature* $\mathbf{f_s}$, given by $\mathbf{f_s} = [a_0, ..., a_3, b_0, ..., b_3]^T$

Location: While clustering dense trajectories, the location information of a trajectory is important, because trajectories far apart in space, even similar in shape, often belong to different clusters. To account for location information of a trajectory in a video frame, we use its mean x and y coordinates. This is denoted as the following *location feature* vector, $\mathbf{f_l} = [\overline{x}, \overline{y}]^T$.

Flow direction: Another feature that we extract is called the *flow direction* feature. We first determine the dominant flow direction in a video, using the median directions of trajectories as the respective representative direction. For simplicity, we only consider horizontal and vertical flow directions. Each trajectory is then associated with a scalar variable f_m which can take the value of $+1$ or -1 depending on whether its direction is along or against the dominant flow direction.

Density: Inspired by the success of density-based clustering methods [12], we extract a *multi-scale density feature* from each trajectory. For a trajectory T^j, its density in a neighborhood of radius ϵ is

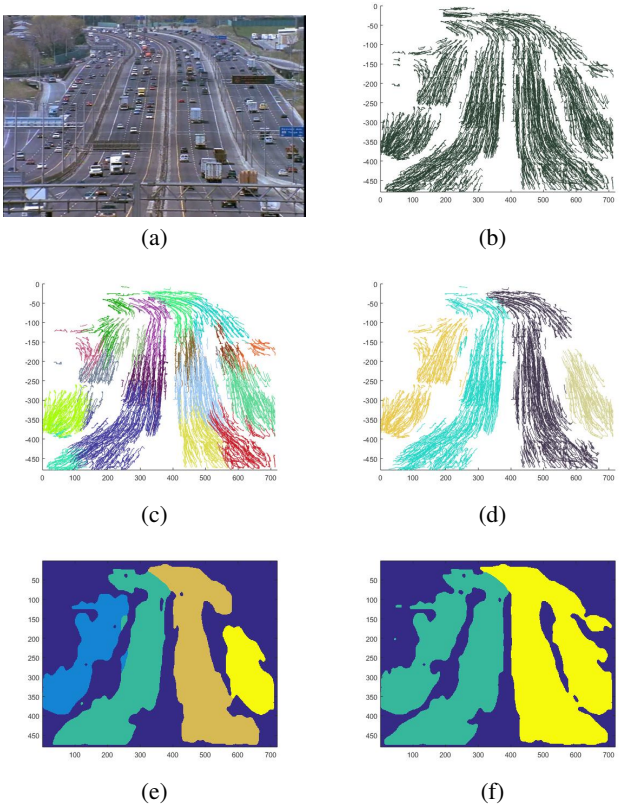

Fig. 2. (a) Sample frame from a test video, (b) All extracted trajectories, (c) Trajectory groups obtained using the proposed clustering algorithm, (d) Crowd flow segments in trajectory domain (e) Crowd flow segments in pixel domain (f) Crowd flow segments in pixel domain with higher value of Δ.

computed as follows:

$$n_{T^j,\epsilon} = |\{T^i | \forall i \neq j, d(\mathbf{f}^j, \mathbf{f}^i) < \epsilon\}| \quad (3)$$

where $d(.)$ denotes the Euclidean distance, and $\mathbf{f}^j = [\mathbf{f_s}^j \mathbf{f_l}^j f_m^j]$. For robustness, we compute a multiscale density feature, corresponding to three values of epsilon. Our multiscale density feature $\mathbf{f_d}$ is given by $\mathbf{f_d} = [n_{j,\epsilon_1}, n_{j,\epsilon_2}, n_{j,\epsilon_3}]^T$.

Finally, all features extracted from a trajectory T^j are concatenated to build a corresponding feature vector $\mathbf{F}^j$ for the trajectory.

$$\mathbf{F}^j = \begin{bmatrix} \mathbf{f_s}^j \\ \mathbf{f_l}^j \\ f_m^j \\ \mathbf{f_d}^j \end{bmatrix} \quad (4)$$

2.2.2. The clustering algorithm

Our next task is to cluster the trajectories, each represented by a feature vector $\mathbf{F}$ as described in equation 4. The trajectories are first clustered using the standard k-means algorithm to produce a set of crude clusters. The value of k is typically large compared to the expected number of flow segments in a video. Nevertheless, choosing the value of k is not critical here (as will become clear later). From each of the cluster obtained by the k-means algorithm,

Algorithm 1 Trajectory clustering

Require: Features from all trajectories: $\mathbf{F}^{all}$
Ensure: Trajectory clusters: $C_1, .., C_N$
$kcluster \leftarrow$ kmeans($\mathbf{F}^{all}$)
$models \leftarrow$ getModels($\mathbf{F}^{all}, kcluster$)
▷ select models from k-means clusters
$\mathcal{P} \leftarrow$ preferenceSet($models, trajecFeat$)
▷ compute preference set using $eq(5)$
$PWdist \leftarrow$ pwdist($\mathcal{P}, \mathcal{P}, Jaccard$)
▷ compute pairwise Jaccard distance using $eq(6)$
for i = 1:num_trajec **do**
$C_i \leftarrow [i]$
while min($PWdist$) < 1 **do**
$[i, j] \leftarrow$ findPos(min($PWdist$))
$C_i \leftarrow [C_i, C_j]$
▷ Merge clusters
Remove C_j
Update $\mathcal{P}(i)$
Remove $\mathcal{P}(j)$
Update $PWdist$
return $C_1, .., C_N$

a fraction (5%) of trajectories is randomly selected, and denoted by $\{M_1, M_2, .., M_\rho\}$. These are considered as the *models*. Selecting the models this way using k-means ensures that there are representatives from all trajectory patterns.

As suggested in [13], each trajectory is compared against all the *models*, and a 'match' is obtained in case the similarity is above a predefined threshold. The idea is that the trajectories from the same motion segment should be similar to the same subset of models. Thus, a binary matrix $\mathcal{P}$, called the *preference set* is obtained which represents the vote of each trajectory to all the models (1 for 'match', and 0 otherwise).

$$\mathcal{P}_{ji} = \begin{cases} 1 & \text{if } ||T_j - M_i|| < \Delta \\ 0 & \text{if } ||T_j - M_i|| \geq \Delta \end{cases} \quad (5)$$

Using the $\mathcal{P}$ matrix, we now perform an agglomerative clustering where the distance between any two trajectories, $d(T^i, T^j)$, is computed using the Jaccard distance.

$$d(T^i, T^j) = \frac{|Q^i \cup Q^j| - |Q^i \cap Q^j|}{|Q^i \cup Q^j|} \quad (6)$$

where Q^i and Q^j represent the two sets of *models* associated with T_i and T_j. Starting with each trajectory as a singleton cluster, we keep merging the clusters with the least Jaccard distance at every iteration, and also compute updated $\mathcal{P}$ matrix accordingly. The process terminates when the minimum Jaccard distance between two clusters is greater than δ (see Fig. 2(c) for the clustered trajectories). The algorithm is described in Algorithm 1 in detail.

2.3. Crowd Flow Segmentation

The final step is to obtain the crowd flow segments from the trajectory clusters, and this is not trivial. We notice that, different trajectory clusters constitute one flow segment. For example, in Fig. 2(c) and Fig. 2(d), multiple trajectory clusters constitute one flow. This kind of situations occur due to objects (in the same crowd flow) having different starting and terminating points in the trajectories, which are eventually grouped into different clusters.

To address the above issue, we propose a slightly modified version of the density-based clustering method [12]. For a cluster C_i, let us define its neighborhood $\mathcal{N}_i$ comprising all clusters C_j ($j \neq i$) that satisfy the following conditions.
(a) Spatial overlap between C_i and C_j should be greater than α.
(b) Difference in flow direction between C_i and C_j should be smaller than β, where flow direction of a cluster is defined as the mean of the directional feature f_m, over all its member trajectories.
(c) Spatial distance between the mean locations C_i and C_j should be smaller than γ, where mean location of a cluster is computed as the mean of the location feature f_l over all its member trajectories.

Starting with an arbitrary trajectory cluster C_i, we compute its neighborhood $\mathcal{N}_i$ (according to above conditions), and merge all clusters in $\mathcal{N}_i$ with C_i. The same process is repeated for every newly added clusters, until the neighborhood is empty. Each of the remaining unvisited clusters is retrieved and processed similarly. The method is summarized in Algorithm 2 (see Fig. 2(e) for final crowd flow segments).

Algorithm 2 Crowd flow segmentation from trajectory clusters

Require: Trajectory clusters: $C_1, C_2, ..., C_N$
Ensure: Crowd flow segments: $S_1, S_2, ..., S_M$
$C = 0$ and $m = 0$
for every unvisited cluster C_i **do**
construct $\mathcal{N}_i$
increment m
expandCluster(C_i,S_m,$\mathcal{N}_i$)
mark C^i as visited
function EXPANDCLUSTER(C_i,S_m,$\mathcal{N}_i$)
add C_i to segment S_m
for every C_j in $\mathcal{N}_i$ **do**
if C_j not visited **then**
construct $\mathcal{N}_j$
merge $\mathcal{N}_i$ and $\mathcal{N}_j$
mark C_j visited
if C_j not a member of any segments yet **then**
Add C_j to the segment S_m
return $S_1, S_2, ...S_M$

3. PERFORMANCE EVALUATION

The database: The proposed approach is validated on a publicly available database of high density crowd videos [7]. This database contains 38 videos of a variety of moving objects such as, traffic and marathon runners, collected from the BBC Motion Gallery and Getty Images website. Among these videos, some are graphic videos and others consist of real life scenes. For evaluation purpose, only the real life videos are considered.

Experimental details: We used the following parameter settings in our experiments. In trajectory extraction: $p = 16$, $r = 40$ and $\tau = 30$; trajectory clustering: $\Delta = 100$, $\delta = 1$; and, in crowd flow segmentation: $\alpha = 10\%$, $\beta = 20$ and $\gamma = 150$, with an exception of $r = 10$ for test sequence #5. We present the crowd flow segmentation results in pixel domain, as done in previous studies. Pixels that belong to the trajectories of one flow segment, are grouped into one region. Assuming the boundaries of crowd flow regions to be smooth, we used a neighborhood sliding filter to smooth the edges of the final segments.

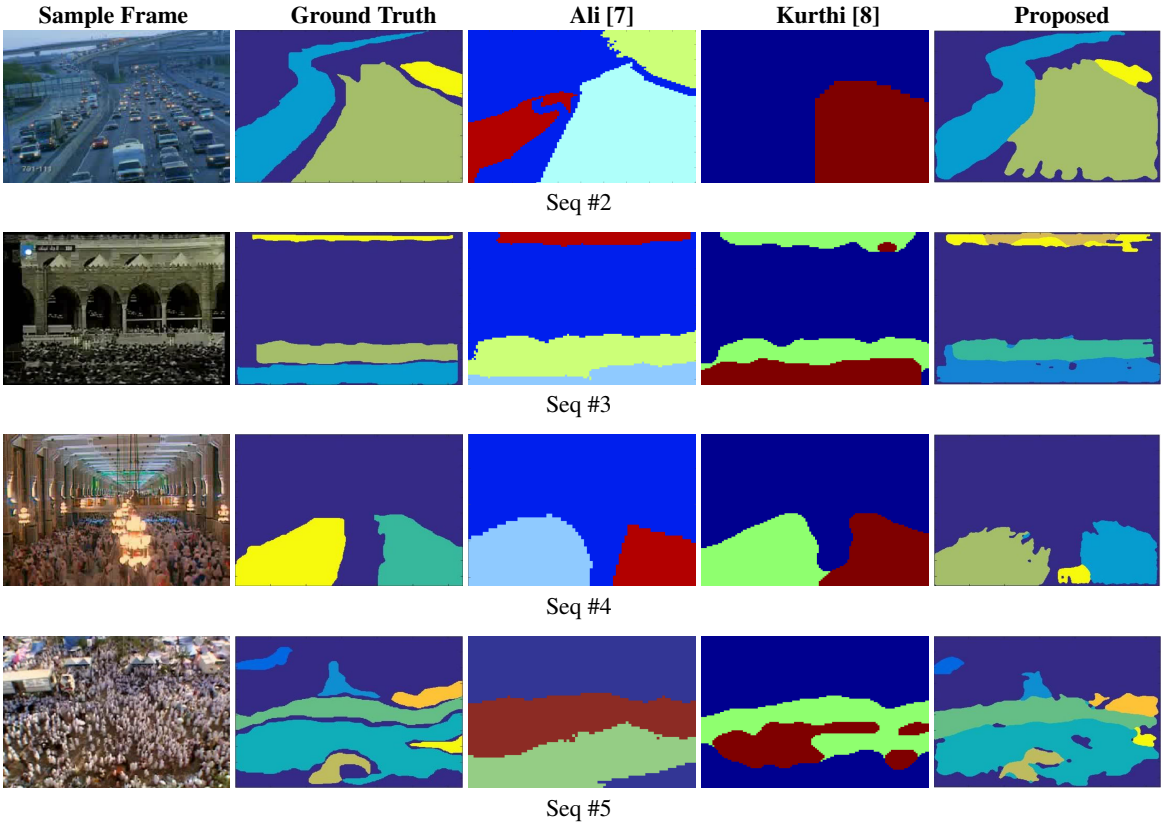

Fig. 3. Qualitative results on crowd flow segmentation

Results: We compare the performance of our approach with several recent works against manually generated ground truth. The objective quality of segmentation results is measured using the Jaccard similarity. For a ground truth segmentation denoted as a labeled set A, and an output denoted as a set B, the Jaccard similarity is given by the intersection over union of the two sets.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{7}$$

The intersection counts the number of matches between the two labeled sets. The union counts the number of pixels rendered as crowded region in any of the two labeled sets. The quantitative comparisons are shown in Table 1. As the set of common videos for which the results from other authors [7, 8, 10] are available, is very small, our comparative study is limited to just 6 videos. Our method outperforms the existing methods by a large margin for sequences 1, 2, and 5. For sequence 6, our system's performance is not as good. In this video, the objects move at a very high speed staying less than 5 frames in the camera. The extracted trajectories are hence too short to be clustered reliably by the proposed algorithm.

The qualitative results in Fig. 3 demonstrates that the proposed system can handle complex crowd scenes (with larger number of segments) much better than other existing methods (see Seq#5). Note that our method can even handle intersecting crowd flow, where all previous approaches have failed. Overall, the proposed approach outperforms the fluid dynamics-based approach [7], and handles complex crowd patterns better than the state-of-the-art [1].

[1]More results available at https://goo.gl/G1qugt

Table 1. Objective evaluation of segmentation performance measured in terms of Jaccard similarity with respect to the manually generated ground truth.

Test samples	Ali [7]	Biswas [10]	Kurthi [8]	Proposed
Seq #1	0.72	0.25	0.92	**0.94**
Seq #2	0.68	0.64	0.47	**0.78**
Seq #3	0.56	**0.69**	0.61	0.66
Seq #4	0.74	0.70	**0.82**	0.76
Seq #5	0.42	0.57	0.40	**0.63**
Seq #6	0.47	**0.63**	**0.63**	0.51
Overall	0.60	0.58	0.64	**0.71**

4. CONCLUSION

We have proposed an unsupervised approach to crowd flow segmentation that relies on trajectory clustering. A novel trajectory clustering algorithm is developed that relies on a robust representation of trajectories by their shape, location, flow direction and density. Our clustering algorithm uses a model-based grouping of trajectories that is able to handle complex motion patterns. Unlike previous approaches, the proposed method can handle the intersecting crowd flows. Limitations of the proposed method includes difficulties in handling crowd scenes with short trajectories and circular motion patterns. Objective and qualitative segmentation results demonstrate that our method can produce superior performance under complex motion scenarios. The proposed clustering algorithm is expected to be useful in other applications requiring high density trajectory clustering.

5. REFERENCES

[1] "1954 kumbh stampede," in $/theguardian.com/world/2003/aug/28/india.maseehrahman)$. 1954, The Gurdian.

[2] "South asia stampede tragedy at hindu shrine," in $http : //news.bbc.co.uk/2/hi/south_asia/255518.stm.$ 1999, BBC News.

[3] S. Ali and M. Shah, "Floor fields for tracking in high density crowd scenes," in *Computer Vision–ECCV 2008*, pp. 1–14. Springer, 2008.

[4] M. Rodriguez, J. Sivic, I. Laptev, and J. Y. Audibert, "Data-driven crowd analysis in videos," in *2011 International Conference on Computer Vision*, Nov 2011, pp. 1235–1242.

[5] S. Wu, Z. Yu, and H. S. Wong, "Crowd flow segmentation using a novel region growing scheme," in *Advances in Multimedia Information Processing - PCM 2009*, 2009, pp. 898–907.

[6] S. Wu and H.S. Wong, "Crowd motion partitioning in a scattered motion field," *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 42, no. 5, pp. 1443–1454, Oct 2012.

[7] S. Ali and M. Shah, "A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis," in *CVPR '07.*, June 2007, pp. 1–6.

[8] S.S. S. Kruthiventi and R.V. Babu, "Crowd flow segmentation in compressed domain using CRF," in *Image Processing (ICIP), 2015 IEEE International Conference on*, Sept 2015, pp. 3417–3421.

[9] R.G. Praveen and R.V. Babu, "Crowd flow segmentation based on motion vectors in H.264 compressed domain," in *IEEE Int. Conf. Electronics, Computing Comm Tech (CONECCT)*, Jan 2014, pp. 1–5.

[10] S. Biswas, R.G. Praveen, and R.V. Babu, "Super-pixel based crowd flow segmentation in H.264 compressed videos," in *Int. Conf. Image Processing (ICIP)*, Oct 2014, pp. 2319–2323.

[11] C. Harris and M. Stephens, "A combined corner and edge detector.," in *Alvey vision conference*. Citeseer, 1988, vol. 15, p. 50.

[12] M. Ester, H.P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," 1996, pp. 226–231, AAAI Press.

[13] M. Fradet, P. Robert, and P. Perez, "Clustering point trajectories with various life-spans," in *Conf Visual Media Production (CVMP), 2009*, Nov 2009, pp. 7–14.