Meta-Tracking for Video Scene Understanding

Pierre-Marc Jodoin, Yannick Benezeth, Yi Wang

Meta-Tracking for Video Scene Understanding

Pierre-Marc Jodoin† Yannick Benezeth†† Yi Wang†
† Université de Sherbrooke
2500 Boul de l'Université, Canada
†† Université de Bourgogne
21000 Dijon cedex France
[Pierre-Marc.Jodoin;Yi.Wang]@usherbrooke.ca yannick.benezeth@u-bourgogne.fr

Abstract

This paper presents a novel method to extract dominant motion patterns (MPs) and the main entry/exit areas from a surveillance video. The method first computes motion histograms for each pixel and then converts it into orientation distribution functions (ODFs). Given these ODFs, a novel particle meta-tracking *procedure is launched which produces* meta-tracks, i.e. *particle trajectories. As opposed to conventional tracking which focuses on individual moving objects, meta-tracking uses particles to follow the dominant flow of the traffic. In a last step, a novel method is used to simultaneously identify the main entry/exit areas and recover the predominant MPs.*

The meta-tracking procedure is a unique way to connect low-level motion features to long-range MPs. This kind of tracking is inspired by brain fiber tractography which has long been used to find dominant connections in the brain. Our method is fast, simple to implement, and works both on sparse and extremely crowded scenes. It also works on highly structured scenes (highways, traffic-light corners, etc.) as well as on chaotic scenes.

1. Introduction

Surveillance video camera networks are increasingly ubiquitous. As a result, the need for intelligent surveillance systems is becoming a glaring economical issue.Intelligent surveillance systems often focus on anomaly detection [21] (detecting illegal U-turns, locating people in restricted areas, etc.) and scene monitoring [17] (people/car counting, traffic jam detection, etc.). In some cases, the system has a clear definition of what it is looking for (e.g. finding people in restricted areas, cars running against traffic, etc.). However, in other cases, the system has to learn the content of the scene while the video streams in. In general, scene understanding means either learning the *scene layout* (locating roads, building, bridges, etc.) [3], the *scene status* (traffic light, congestion, crowd behaviors, etc.) [17] or predominant *motion patterns* (MPs) [19].

It is widely accepted that grasping the content of a scene is not equally easy for all scenarios. Due to their content, some videos are challenging and require complex solutions. Common issues reported are the following:

1. **Clutter:** the number of moving objects in a video can lead to challenging situations. That is true in crowds and in heavy traffic jams where several moving persons/vehicles partly occlude other moving objects (see Fig. 1(a)). These scenarios make it difficult (if not impossible) to track every moving object.
2. **Structure of the flow:** the nature of the flow is another critical variable. In some sequences, the flow is highly structured with objects following well-defined patterns. An example is shown in Fig. 1(b) in which vehicles all run at the same speed without changing lanes. In other scenarios, the flow is chaotic with persons and vehicles crossing each other and heading towards different directions. This is true in shopping malls, airports, and where street traffic is poorly regulated as in Fig. 1(c).
3. **Layout:** the layout of the scene determines heavily the path of the flow. A simple layout might involve one or two predominant directions (see Fig. 1(b)), but more complicated scenes include splitting and merging of non-linear MPs (roundabouts, crossroad, Y-shape road as in Fig. 1(d)). Recovering and representing such non-linear and long-range MPs (especially when they cross each other) is a difficult task.

In this paper, we present a method to recover MPs that works on sparse scenes as well as on crowded scenes, on structured and chaotic scenes, and can handle non-linear (and yet long-range) MPs. To do so, our method relies on pixel-based orientation distribution functions (ODFs) computed from motion histograms. These ODFs are used to recover *meta-tracks*, *i.e.* trajectories of particles transported along the ODFs. Meta-tracks are then clustered to recover predominent MPs and the main entry/exit points.

Our method is inspired by MRI brain tractography [13] which recovers long-range, intricated and cluttered connections between different areas of the brain.

2. Previous Work

Since activities in public areas often follow basic rules, several scene understanding methods quantify space and time into a number of states with transition probabilities. Common models are HMMs [12], Bayesian networks [6], context free grammars [18], and other graphical models [16]. Methods focusing on global behavior understand-

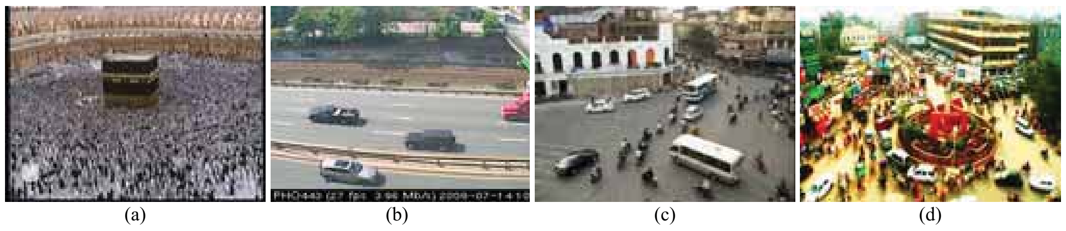

Figure 1. Common issues when learning motion patterns. (a) Highly crowded scene with a simple layout (people are all turning in the same direction). (b) Low-density scene with structured flow and a simple layout (flow goes left or right). (c) Example of a chaotic flow in which vehicles, bicycles and pedestrians cross each other in an unstructured way. Vehicles have different speed, do not follow lanes and the flow is not regulated by traffic lights. (d) Scene with complex layout with several entry/exit points and long-range motion patterns.

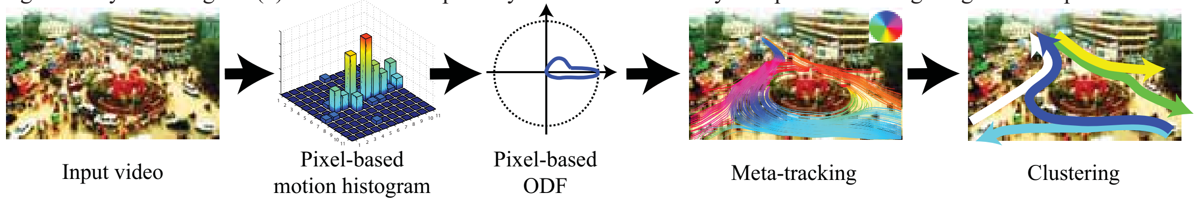

Figure 2. Our method is made of the following 4 steps: 1) compute motion histogram for each pixel, 2) convert each motion histogram into an ODF, 3) meta tracking and 4) recover entry/exit point and cluster meta-tracks to recover the main motion patterns (MPs). Meta-tracks color code indicate point-wise orientation.

ing often rely on object tracking [21, 6, 18]. The main advantage of tracking-based methods is their ability to cope with complex layouts and recover non-linear long-range MPs.However, it is broadly accepted that object-tracking is ill-suited for videos with a large number of moving objects, especially in crowds as in Fig. 1 (a) and (d).

As a solution, non-tracking methods have been proposed. Similar to our method, particle advection methods have been shown successful to handle dense videos [17, 20, 1]. However, some of these methods were designed to detect local anomalies [20] and thus are not suited to recover long-range MPs. Alternatively, Ali and Shah [1] proposes a scene segmentation method based on particle advection which can recover long-range MPs. Unfortunately, that method was not meant to deal with overlapping MPs (as in Fig. 1(c) and (d)) since each pixel is assigned only one motion label. Solmaz *et al.* [17] use particles to recognize among five crowd behaviors based on the eigenvalues of a flow-based Jacobian matrix. It is not clear though how this method can recover individual MPs, especially if their shape differs from the five predefined behaviors.

Work by Hu, Ali and Shah [1, 2, 10] is to our knowledge the closest contribution to ours. Given flow vectors, they find motion paths with a tracking method which they call *sink seeking*. Sink seeking leads to *sink paths* which are then clustered into *super tracks*. However, since their method relies on pixel-based flow vectors, the tracking method cannot recover overlapping MPs which are frequent in crossroads and in Y-shape roads (results are reported in section 4). From the same lab [11], a scene segmentation method based on a motion flow field but without particles has been proposed. Motion flow vectors are clustered together with a hierarchical clustering based on a geodesic distance matrix. As for [10], this method was not meant to handle crossing and overlapping MPs.

Let us also mention work by Wang *et al.* [19] which decomposes the video into *clips* in which local motion is quantized into *words*. These words are then clustered into *topics*, each clip being modeled as a distribution over these topics. Although these topics are motion patterns close to the ones we are looking for, since this technique works only on recurrent activities, it is ill-suited for unstructured flow (as in Fig. 1(c)). Similar approaches can be found in [9, 12].

3. Our Method

As shown in Fig. 2, our method is a 4-step pipeline which: 1) computes a motion histogram for each pixel, 2) converts motion histograms into ODFs, 3) performs meta-tracking and 4) clusters meta-tracks to recover MPs as well as the entry/exit points.

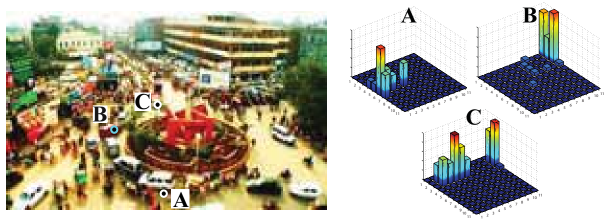

Figure 3. Motion histograms showing (A) horizontal, (B) diagonal right, and (C) bimodal motion (upper left and upper right).

3.1. Motion Histograms

Consider I, a video with constant inter-frame interval, and (u_t, v_t) the optical flow at time t computed by a motion estimation method. In this work, we use the Horn/Schunk -

Lucas/Kanade method proposed by Bruhn *et al.* [5] which we found to be a good compromise between precision and speed. Although ODFs (to be defined later) could be obtained with tracklets as in [23], we empirically found that optical flow is much easier and faster to compute. Once optical flow has been computed for each frame, a motion histogram $M_p(u,v)$ is computed at each pixel p. This histogram contains the number of times a pixel p had its motion vector $(u_{t,p}, v_{t,p})$ equal to (u,v) during the entire video. Since M_p takes integer indices, motion vectors $(u_{t,p}, v_{t,p})$ are rounded up to the nearest integer. Fig. 3 shows three pixel-based motion histograms.

3.2. Converting Motion Histograms into ODFs

Motion histograms are ill-suited for meta-tracking. Our tracking technique relies on orientation sampling based on an opening angle (to be define in Section 3.3) and computing it on a motion histogram is not only time consuming but also error prone due to aliasing problems. Furthermore, our tracking method is only concerned with flow orientation and not flow amplitude.

An ODF is the probability density function of finding a given motion orientation. As opposed to motion histograms, ODFs have no information on the magnitude of the flow. This makes the ODF representation simpler (1D instead of 2D) which is a key advantage computational wise for the upcoming meta-tracking algorithm. A pixel-based ODF $\Phi_p(\theta)$ is computed through radial integration:

$$\Phi_p(\theta) = \frac{1}{\lambda_p} \int_0^R M_p(\theta, r) dr \tag{1}$$

where λ_p is a normalizing constant and $M_p(\theta, r)$ a motion histogram in polar coordinates (entry (u,v) is represented by an angle $\theta \in [0, 360]$ and a distance to the origin $r > 0$).

3.3. Meta-Tracking

Once every pixel has been assigned to an ODF, the meta tracking function can be launched. This procedure lays a grid of particles and have them advected in the scene following an iterative algorithm. Let q be a particle at position $\rho_q^\tau = (i,j)$ and iteration τ. At each iteration, the goal is to estimate the next particle position $\rho_q^{\tau+1}$ given the ODF $\Phi_q(\theta)$. Mathematically, this boils down to

$$\rho_q^{\tau+1} = \rho_q^\tau + \delta \vec{v}_q^{\,\tau+1} \tag{2}$$

where δ is a the step size and $\vec{v}_q^{\,\tau+1}$ is the unit-length motion vector of particle q at iteration $\tau+1$. Since $\vec{v}_q^{\,\tau+1}$ is provided by $\Phi_q(\theta)$, it should be parallel to the flow. One way of computing $\vec{v}_q^{\,\tau+1}$ is by taking the average ODF direction,

$$\vec{v}_q^{\,\tau+1} = f\left(\frac{\int \Phi_q(\theta)\theta d\theta}{\int \Phi_q(\theta) d\theta}\right) \tag{3}$$

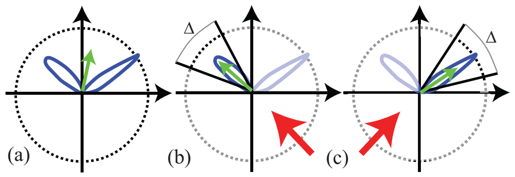

Figure 4. Sampling a bimodal ODF to get the resulting vector flow $\vec{v}_q^{\,\tau+1}$ (in green). In (a) $\vec{v}_q^{\,\tau+1}$ has been obtained by taking the average ODF orientation. In (b) and (c), the ODF is sampled according to the incoming vector $\vec{v}_q^{\,\tau}$ and an opening angle Δ.

where $f(.)$ converts an angle θ into a 2D vector of unit length. But as shown in Fig. 4 (a), in multimodal areas, Eq. (3) does not give a motion vector corresponding to an actual direction mode. Let us mention that this approach is similar to scene understanding methods relying on motion vectors averaged over time [1, 2, 15].

Alternatively, we sample the ODF in the area defined by the incoming direction $\vec{v}_q^{\,\tau}$. As shown in Fig. 4 (b) and (c), given an opening angle Δ, sampling is done by the following function:

$$\vec{v}_q^{\,\tau+1} = f\left(\frac{\int h_\Delta(\theta - \theta_i)\Phi_q(\theta)\theta d\theta}{\int h_\Delta(\theta - \theta_i)\Phi_q(\theta) d\theta}\right) \tag{4}$$

where θ_i is the incoming direction vector angle and $h_\Delta(\theta - \theta_i)$ is a kernel function. In our implementation, $h_\Delta(\theta - \theta_i)$ is a 1D zero-mean Gaussian function with standard deviation set to $\Delta/4$ (a rectangular function of size Δ could also be used). As shown in Fig. 4 (c), this allows particle to follow different directions depending on their incoming direction. This is a key advantage of our method which makes it robust to overlapping motion patterns.

In order to prevent a particle from being abruptly deviated by a noisy or corrupted ODF, Eq. (4) is incorporated within a temporal filter

$$\vec{v}_q^{\,\tau+1} = \alpha \vec{v}_q^{\,\tau} + (1-\alpha) f\left(\frac{\int h_\Delta(\theta - \theta_i)\Phi_q(\theta)\theta d\theta}{\int h_\Delta(\theta - \theta_i)\Phi_q(\theta) d\theta}\right) \tag{5}$$

where $\alpha \in [0,1]$ is a persistency constant (0.3 in our case). Algorithm 1 (in the supplementary material) summarizes the meta-tracking procedure for a particle q.

Stochastic sampling The meta-tracking algorithm stops whenever one of the following criteria is satisfied: (1) if the iteration number has reached a MAX value (2000 in our case), (2) if the particle took a sharp turn between τ and $\tau + 1$ (in our case, more than 20^o), and (3) if the particle enters an area where no activity has been recorded. Unfortunately, since ODFs are sometimes imprecise near the edges of occluding objects such as buildings and bridges, meta tracks next to these structures may be wrongly deviated towards areas with no activity. This results into unexpected stops and outlying meta tracks. As a solution, each

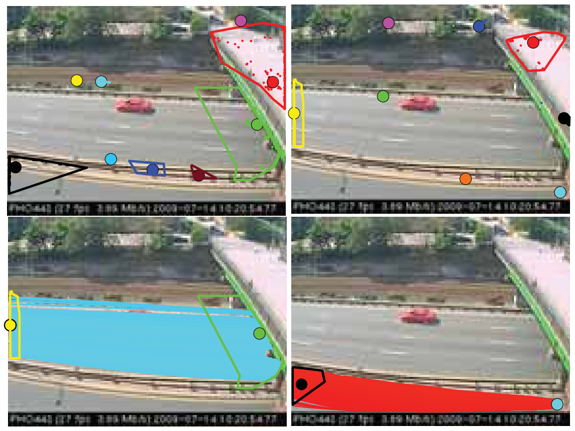
Figure 5. Results from our clustering method. Top left: 9 entry areas, top right: 8 exit area. Bottom left and right, MPs with the largest number of tracks together with their entry and exit areas.

time a meta-track hits a region with no activity, the tracking procedure backs off l pixels and sends N new particles (usually $l = 5$ and $N = 20$). These particles are tracked for $2l$ pixels following a stochastic ODF sampling. As opposed to deterministic sampling shown in Fig. 4, stochastic sampling randomly selects a direction with respect to the ODF shape. In this way, the N particles end up following different paths. If all those particles end up in an area where no activity has been recorded, the current meta-track is stopped. Otherwise, the valid tracks are averaged and Algorithm 1 restarts form that point [1].

3.4. Recovering Entry/Exit Areas and MPs

MPs are usually obtained by grouping tracks with similar shape, length and direction [22, 14]. Here, we propose a new method to group tracks without having to extract trajectory features or compute sophisticated inter-track distance metrics as is usually the case. Our method relies on a simple assumption: tracks within a single MP all start from a common entry point and finish into a common exit point. Our method thus aims at identifying the entry and exit areas and then cluster the meta-tracks accordingly.

Entry points are localized by analyzing the first point of each track[2]. Since the number of entry areas is not known *a priori*, we use a bottom up hierarchical clustering method which does not assume a predefined number of clusters. In this way, each point starts in its own cluster and, iteratively, the nearest clusters are merged together as the algorithm moves up. The algorithm stops when the intra-cluster distance reaches a maximum value (between 20 and 50 in our case). Typical results are shown in Fig. 5. The small dots in the top images are the first and last points of each track while the large dots are cluster centers. MPs are obtained by grouping tracks connecting the same entry and exit areas.

[1] An illustration of stochastic sampling can be downloaded from the following web page: *http://ilt.u-bourgogne.fr/benezeth/projects/AVSS2013/*.

[2] Exit points are obtained by analyzing the last point of each track

These MPs are sorted according to their number of tracks, predominant MP having more tracks. Every MP with less than 5% of tracks are discarded.

4. Experimental Results

In order to gauge performances, we tested our method on 12 videos representing different challenges (see Fig. 8 and Fig.1 in the supplementary matterial). These videos contain between 300 and 8000 frames. They were taken from the UCF database, the changedetection.net [8] dataset, Youtube as well as our own database. Some videos have a small number of moving objects (*e.g. MIT, Street Light*), while others show highly crowded scenes (*e.g. Marathon, Mecca*). Some have a simple layout (*e.g. Street Light, Boulevard*) while others have a complex one (the 3 *Roundabouts*). Some have well structured dynamics (*e.g. MIT, Rush Hour*) while others show a fairly chaotic scene (*e.g. Indian Market, Hanoi*). The videos as well as the code can be downloaded from our project page.

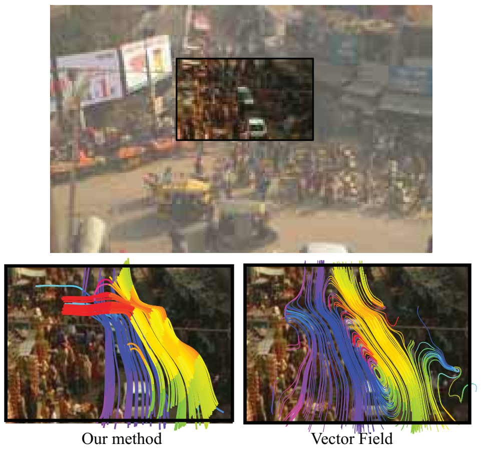

Figure 6. Our method performs well in areas where MPs overlap.

The resulting meta-tracks for each video are shown in Fig. 8 and in the supplementary material. As can be seen, our meta-tracking method works well on every video, including the ones with complex layout and unstructured dynamics. It takes on average 30 secs to track 1000 particles on a 2.8 GHz laptop with Matlab code.

We compared our meta tracking results to those produced by three alternative solutions: two object-tracking methods and a vector field (VF) method similar to [1, 2, 10]. The object-tracking methods are Mean-shift [7] and the L1 tracker from Bao *et al.* [4]. We used the OpenCV implementation for Mean-shift and Bao *et al.*'s code for the L1 tracker. Since Bao *et al.* did not specified how to initialize tracks, we manually initialized tracks by selecting every moving objects (or a subset of the moving objects for highly crowded scenes). As for the VF method, we implemented a tracker similar to [1, 2, 10] which uses a vector field to recover trajectories (or "sink paths"). For each method, small isolated tracklets have been filtered out.

Videos	Methods	Outliers	ATL	**Videos**	Methods	Outliers	ATL	**Videos**	Methods	Outliers	ATL
Street Light	Our method	1	298	**Marathon**	Our Method	1	941	**Hanoi**	Our Method	14	423
	Mean-Shift	10	241		Mean-Shift	94	218		Mean-Shift	70	327
	L1 tracker	58	236		L1 tracker	–	–		L1 tracker	62	310
	VF	0	295		VF	14	876		VF	52	430
MIT	Our Method	18	606	**Mecca**	Our Method	2	660	**Round-about 1**	Our Method	10	346
	Mean-Shift	51	429		Mean-Shift	–	–		Mean-Shift	52	250
	L1 tracker	18	417		L1 tracker	–	–		L1 tracker	40	185
	VF	22	644		VF	23	588		VF	19	297
Escalator	Our Method	12	403	**Indian Market**	Our Method	5	491	**Round-about 2**	Our Method	4	637
	Mean-Shift	75	345		Mean-Shift	89	254		Mean-Shift	86	394
	L1 tracker	81	211		L1 tracker	73	354		L1 tracker	90	326
	VF	23	389		VF	19	517		VF	10	806
Boulevard	Our Method	6	406	**Rush hour**	Our Method	3	284	**Round-about 3**	Our Method	5	943
	Mean-Shift	90	135		Mean-Shift	77	157		Mean-Shift	–	–
	L1 tracker	14	279		L1 tracker	50	186		L1 tracker	–	–
	VF	9	239		VF	13	299		VF	26	619

Table 1. Percentage of outlying tracks and average track length (ATL) obtained on 12 videos by 4 methods : our meta-tracking method, the Mean Shift object tracker [7], the L1 object tracker [4] and a vector field (VF) tracker. The "–" annotation are where a method has failed to recover any good tracks.

Table 1 shows for each method and each video, the percentage of outlying tracks (tracks that do not correspond to the actual flow or to any moving object) and the average track length (ATL). These results clearly show that object-tracking methods quickly diverge when the number of moving objects becomes large. This is especially true with Mean-Shift whose automatic initialization procedure is fairly sensitive. The ATL values show that inlying tracks produced by our method are longer than those of the object-tracking methods. As mentioned in [2], this is a strong indication that our method is more consistent and thus better suited to recover long-range MPs.

Videos	# of Classes	# of True Detections	# of False Detections
Street Light	4	3	0
MIT	8	5	1
Escalator	4	4	0
Boulevard	2	2	0
Indian Market	7	5	1
Rush Hour	6	6	0
Hanoi	4	4	0
Roundabout 1	7	5	0
Roundabout 2	3	3	0
Roundabout 3	4	3	3
Marathon	3	3	0
Mecca	5	5	0

Table 2. Motion patterns segmentation results.

Our method and VF produce similar results on sequences with simple layout (*Boulevard, Rush Hour, Street Light*). However, our experiments reveal that in complex scenes, our method performs better than VF. As shown in Fig. 6, since our meta-tracker is based on ODFs, the bimodal nature of the flow in areas where MPs cross is correctly handled. This is clearly not possible with a vector field.

Fig. 7 shows the four most predominant MPs for sequences *Escalator*, *Roundabout 1* and *MIT*. Our clustering Matlab code takes on average less than 1 second to process a scene. Quantitative results are presented in Table 2; ground truth for each video has been manually generated. The number of classes correspond to the dominant MPs.

Results in table 2 show the number of true and false detections for each sequence. False detections are usually generated in areas where motion is intricated. Let us emphasize good results obtained on complex sequences (*Roundabout 1,Roundabout 2, Hanoi* and *Indian Market*) for which we have no more than 1 false detection. Note that the size of the moving objects do not influence the output of the method.

4.1. Conclusion and Future Work

We introduced a method to recover motion patterns and entry/exit areas that works on sparse and crowded scenes, on structured and chaotic scenes and on scenes with simple and complex layout. This method is based on pixel-based ODFs which summarizes the flow of activity at each point of the scene. A grid of particles is laid on the scene and transported through the ODF field. This results into particle trajectories called *meta-tracks*. Those meta-tracks are then grouped together to form MPs. The experiments show that our method outperforms object-tracking methods, especially on dense scenes and produce better results than the vector field approach in scenes containing overlapping MPs.

References

[1] S. Ali and M. Shah. A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis. In *CVPR*, 2007.

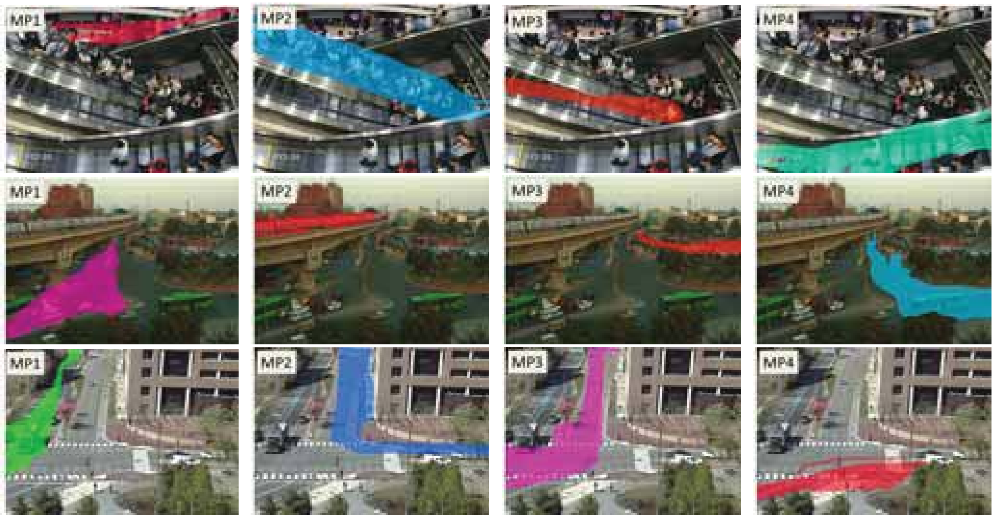

Figure 7. The four most predominent MPs extracted with our method for *Escalator*, *Roundabout 1* and *MIT*.

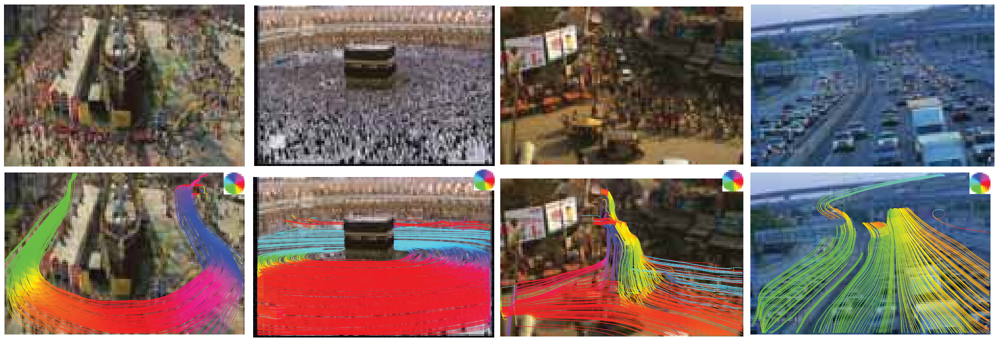
Figure 8. Four results of our meta tracking method. From left to right : *Marathon, Mecca, Indian Market,* and *Rush Hour*.

[2] S. Ali and M. Shah. Floor fields for tracking in high density crowd scenes. In *ECCV*, p. 1–14, 2008.

[3] J. Alvarez, T. Gevers, Y. LeCun, and A. Lpez. Road scene segmentation from a single image. In *ECCV*, p. 376–389, 2012.

[4] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *CVPR*, 2011.

[5] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *Intern. J. Comput. Vis.*, 61(3):211–231, 2005.

[6] S. Calderara, R. Cucchiara, and A. Prati. A distributed outdoor video surveillance system for detection of abnormal people trajectories. In *IEEE Conf. on Dist. Smart Cameras*, p. 364–371, 2007.

[7] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *in CVPR*, p. 142–149, 2000.

[8] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. changedetection.net: A new change detection benchmark datase. In *Workshop on Change Detection at CVPR*, 2012.

[9] T. Hospedales, S. Gong, and T. Xiang. A markov clustering topic model for mining behaviour in video. In *ICCV*, p. 1165–1172, 2009.

[10] M. Hu, S. Ali, and M. Shah. Detecting global motion patterns in complex videos. In *ICPR*, p. 1–5, 2008.

[11] M. Hu, S. Ali, and M. Shah. Learning motion patterns in crowded scenes using motion flow field. In *ICPR*, p. 1–5, 2008.

[12] D. Kuettel, M. Breitenstein, L. Gool, and V. Ferrari. What's going on? discovering spatio-temporal dependencies in dynamic scenes. In *CVPR*, p. 1951–1958, 2010.

[13] M.Descoteaux, R. Deriche, T. R. Knösche, and A. Anwander. Deterministic and probabilistic tractography based on complex fibre orientation distributions. *IEEE TMI*, 28(2):269–286, 2009.

[14] B. Morris and M. Trivedi. Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In *CVPR*, 2009.

[15] M. Rodriguez, J. Sivic, I. Laptev, and J.-Y. Audibert. Data-driven crowd analysis in videos. In *ICCV*, p. 1235–1242, 2011.

[16] C. Simon, J. Meessen, and C. DeVleeschouwer. Visual event recognition using decision trees. *Multimedia Tools and App.*, 2009.

[17] B. Solmaz, B. Moore, and M. Shah. Identifying Behaviors in Crowd Scenes Using Stability Analysis for Dynamical Systems. *IEEE Trans. Pattern Anal. Machine Intell.*, p. 1–8, 2012.

[18] H. Veeraraghavan, N. Papanikolopoulos, and P. Schrater. Learning dynamic event descriptions in image sequences. In *CVPR*, 2007.

[19] X. Wang, X. Ma, and E.Grimson. Unsupervised activity perception in crowded and complicated scenes using hierarchical bayesian models. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(3):539–555, 2009.

[20] S. Wu, B. Moore, and M. Shah. Chaotic invariants of lagrangian particle trajectories for anomaly detection in crowded scenes. In *CVPR*, p. 2054–2060, 2010.

[21] B. Yao, L. Wang, and S. Zhu. Learning a scene contextual model for tracking and abnormality detection. In *CVPR*, p. 1–8, 2010.

[22] Z. Zhang, K. Huang, and T. Tan. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *ICPR*, 2006.

[23] B. Zhou, X. Wang, and X. Tang. Random field topic model for semantic region analysis in crowded scenes from tracklets. In *CVPR*, 2011.