Analyzing Crowd Behavior in Naturalistic Conditions: Identifying Sources and Sinks and Characterizing Main Flows

Sultan D. Khan[1], Stefania Bandini[1], Saleh Basalamah[2], Giuseppe Vizzari[1]*

[1] *Complex Systems and Artificial Intelligence Research centre, Università degli Studi di Milano–Bicocca, Milano, Italy*
[2] *Department of Computer Engineering, Umm Al Qura University, Makkah, Saudi Arabia*

Abstract

Pedestrians, in videos taken from fixed cameras, tend to appear and disappear at precise locations such as doors, gateways or edges of the scene: we refer to locations where pedestrians appear as sources (potential origins) and the locations where they disappear as sinks (potential destinations). The detection of these points and the characterization of the flows connecting them represent a typical preliminary step in most pedestrian studies and it can be supported by computer vision approaches. In this paper we propose an algorithm in which a scene is overlaid by a grid of particles initializing a dynamical system defined by optical flow, a high level global motion information. Time integration of the dynamical system produces short particle trajectories (tracklets), representing dense but short motion patterns in segments of the scene; tracklets are then extended into longer tracks that are grouped using an unsupervised clustering algorithm, where the similarity is measured by the Longest Common Subsequence. The analysis of these clusters supports the identification of sources and sinks related to a single video segment. Local segment information is finally combined to achieve a global set of traces identifying sources and sinks, and characterizing the flow of pedestrians connecting them. The paper presents the

*Corresponding author
Email address: sultan.khan@disco.unimib.it, bandini@disco.unimib.it, smbasalamah@uqu.edu.sa, vizzari@disco.unimib.it (Sultan D. Khan[1], Stefania Bandini[1], Saleh Basalamah[2], Giuseppe Vizzari[1])

defined technique and it discusses its application in a real-world scenario.

Keywords:
crowd analysis
2010 MSC: 68T42,68U20

---

1. Introduction

Crowded scenes are composed of a large number of people, exhibiting different behaviors in a relatively constrained space. The vagueness of this definition is strictly related to the difficulties in defining what a crowd of pedestrian is; we will not try here to be more specific or precise, but rather highlight the growing need to consider the presence and behaviors of pedestrians in the environment by designers, planners and decision makers (see, e.g., a recent report commissioned by the U.K. Cabinet Office on this subject [1]). In particular, public safety in crowded situations (e.g. concerts, religious or political gatherings) has become an important research area in the last years, with relevant contributions from physics, psychology, computer science and, of course, civil engineering. Acquiring data for this kind of study is obviously absolutely crucial for sake of understanding the implied phenomena and evaluating developed solutions for analysis, decision support, prediction. In video surveillance, scene modeling and understanding is also an important research area. Important tasks of scene modeling and understanding are (i) extracting motion information (e.g. trajectories), (ii) identification of entry and exit points of trajectories in the analyzed scene, (iii) characterization of the interaction of trajectories (highlighting, for instance, crossings or potential conflicts).

Pedestrians in videos taken from fixed cameras tend to appear and disappear at relatively precise and recurring locations, such as doors, gateways or particular portions of the edges of the scene. Moreover, pedestrian behavior in a given scene might imply waiting at a certain location then moving whenever certain conditions are met or given events happen. We refer to locations where pedestrians appear or start moving as *sources* (potential origins of a trajec-

tory) and the locations where they disappear or stop moving as *sinks* (potential destinations). Traditionally, crowd analysis is performed by the analyst who manually identifies and detects different relevant activities in the scene. A portion of video is given to each analyst together with a list of events (behaviors) and objects to look for. The analyst informs the concerned authorities if any of the given events or objects are detected. Such kind of manual analysis of video is labor intensive, time consuming and prone to errors due to weak perceptive capabilities of humans, but also to the repetitiveness of the activity.

In this paper, we propose an approach for crowd behavior analysis (and, to a certain extent, understanding in the acceptation of the term adopted by [2]) adopting two novel algorithms, the first able to generate long, dense, reliable and accurate pedestrian trajectories and the second clustering them to generate long term reliable and abstract information describing flows in the whole video. The final results provide directly information characterizing flows but it also represents a starting point for further high-level analyses of crowd behavior. The approach starts by dividing the input video into multiple *segments* of equal length and, considering that the frame rate of the video is constant, duration. The initial frame of each segment is overlaid by a grid of particles initializing a dynamical system defined by optical flow, as discussed by [3]. Time integration of the dynamical system over a segment of the video provides particle trajectories (tracklets) that represent motion patterns in the scene for a certain time interval associated to the analyzed segment. We detect sources, sinks and main flows in the segment (for sake of brevity sometimes we will refer to this information as segment local *track*) by analyzing motion patterns followed by clusters of tracklets, obtained using an unsupervised hierarchical clustering algorithm, where the similarity is measured by the Longest Common Sub-sequence (LCS) metric. To achieve final global tracks, covering all the video, we cluster the achieved local tracks through the same hierarchical clustering algorithm. Our main contributions are: (1) Generating dense and long trajectories, (2) identifying sources and sinks, (3) understanding behavior of the crowd in the scene by considering full length video, (4) achieve the above results without requiring

object detection, tracking, nor training, targeting employment in naturalistic conditions. The paper breaks down as follows: the following Section presents the current state of the art in the identification and characterization of pedestrian flows in crowded scenes, while Section 3 presents the overall proposed approach. Section 3.1 focuses on the algorithm to extract long, dense, accurate and reliable trajectories and Section 3.4 describes in details the clustering algorithm applied to generate local and global tracks. Section 4 describes the achieved experimental results, also by comparing the proposed approach with the most relevant existing alternatives. Conclusions and future developments end the paper.

2. Related Works

With the advancement in computer vision technology, researchers developed tracking methods that in certain conditions can automatically detect, track and identify specific activities in the scene. [4] developed a tracking algorithm by modeling human shape and appearance as articulated ellipsoids and color histogram respectively for crowded scenes. [5] use Markov chain Monte Carlo based particle filter to handle interaction between multiple targets in crowded scene. [6] detects interest points in each frame by tracking pedestrians, and this activity is performed by finding correspondence among points between frames. [7] developed an unsupervised bayesian clustering method to detect individuals in crowd: for each frame, detection of individuals is performed ignoring the relationship between frames. [8] detect individual objects on the basis of assumption that objects move in different directions. [9] develop a tracking system by solving data association problem by utilizing Generalized Minimum Clique Graph (GMCP) in order to detect an individual in different frames of a video. Intuitively, detection and tracking of individuals rely on the performance of detection and tracking algorithms. However, in crowded scenes, where the number of objects is often in the order of hundreds, these tasks usually fail due to (i) the variable and potentially low number of pixels per object and (ii) frequent and

severe occlusions related to the constant interaction among the objects (pedestrians) in the scene. These challenging characteristics of the analyzed videos can be at least partly avoided in laboratory situations: for instance, in [10] the authors successfully gather pedestrians' trajectories and gather useful data about their behavior but they employ a manual or automatic but facilitated form of identification. Moreover, as we will discuss in the experimental evaluation of the presented approach, the adopted tracking algorithm (Lucas-Kanade tracker - KLT [11]) does not provide sufficiently accurate results in naturalistic conditions.

Intuitively, detecting sources and sinks (as introduced above) implies detection and tracking of objects, potentially followed by an analysis of the trajectories: this kind of approach was adopted by [12], which analyses low density situations and essentially relies on the performance of the tracking algorithm, which is low in crowded situations. Research in this area has therefore instead assumed that raw data about pedestrian paths should be considered as noisy or unreliable: [13], for instance, employ a so-called *weak tracking* system and they aggregate raw *tracklets* through a mean-shift clustering technique allowing them to identify entry and exit zones in the scene. More recently, in order to overcome the limitation of traditional tracking methods, research has focused on gathering global motion information at higher level, often based on optical flow analysis.

Trajectories capture the local motion information of the video. Long and dense trajectories (that is, trajectories representing a large number of paths followed by different pedestrians, reaching a significant length) provide good coverage of foreground motion as well as of the surrounding context. There are two types of representations for characterizing motion information from the video: space-time local features (like corner points, SIFT features etc.) and dense optical flow [14]. In the first type, features are detected in one frame which are then tracked through rest of the frames of a video, whereas the second type is based on dense optical flow, where a flow vector is estimated for every pixel. Since dense optical flow estimates a change for every pixel it provides a better

representation of motion in video. A large number of approaches for extracting feature trajectories from video exist:

- [15] extracts feature trajectories by tracking Harris3D interest points; [16] used KLT for extracting trajectories represented as a sequence of log-polar quantized velocities which later on used for action classification;
- [17] also used KLT for extracting trajectories, that are then clustered and affine transformation matrix representing trajectories is computed for each cluster;
- [18] extract trajectories by matching SIFT descriptors between two consecutive frames;
- [19] combine both KLT tracker and SIFT descriptor matching to extract long-duration trajectories, and random points are sampled for tracking within the region of existing trajectories in order to assure dense coverage;
- [20] extract feature point trajectories in the regions of interest; they compute histogram of gradient (HOG) and histogram of optical flow (HOF) descriptors along the trajectories.
- [21] also uses KLT method for extracting sparse trajectories. The authors propose Random Topic Model (RTM) for learning semantic regions from the motions of pedestrians in crowds
- [22] use KLT trajectories and propose Mixture model of Dynamic pedestrian Agents (MDA) that analyse the collective behavior of pedestrian in crowds after learning from the real data.

Resulting trajectories from these approaches are effectively long duration but they are typically sparse and can not capture whole motion information of the video because only few feature points are detected.

On the other hand, dense optical flow captures whole motion information of the video, as we estimate a flow vector for every pixel of a frame; but due to the

unpredictable nature of the pixel (due to its sensitivity to the illumination), we can not extract reliable long duration trajectories. There is limited literature about dense trajectories:

- [23] extract long range trajectories using dense optical flow;
- [24] extract objects from video using dense optical flow trajectories;
- video is represented as set of particles and their trajectories are computed using variational optical flow in [14];
- in [3], particle trajectories are obtained by overlying a grid of particles on the initial frame of video, initializing a dynamical system. Time integration of this dynamical system provides particle trajectories that represent motion in the scene. This method represents a very useful starting point for our goals, especially for generating robust local movement trajectories, although it is not aimed at providing global pedestrian motion information but just for identifying specific crowding situation or movement patterns.

Generally, these techniques are quite reliable when so called *structured crowds* [25] are analyzed: this is mostly due to the nature of this kind of situations, when flows of pedestrians can include a very large number of individuals that, however, follow relatively stable flows that are generally well separated and not conflicting (e.g. people in a marathon, pilgrims performing Tawaf during the Hajj). Achieved particle trajectories in high density unstructured crowds are, instead, normally not accurate and unreliable due to (1) severe occlusions that occur frequently, (2) ambiguities arising at the boundary of the conflicting flows as reported in [26]. In these cases, a particle can drift to the side of another motion boundary and it can mix with different motion.

The related works most close to the presented approach are [27] and [28], which extract motion trajectories using KLT and adopt hierarchical clustering algorithms for detecting dominant flows in scene. These methods do not consider the whole video, but they rather consider a portion of it; moreover they do not actually try to identify sources and sinks of different flows but rather capture

information about a low number of frames which provide inadequate information for understanding the overall behavior of the scene. In this paper, we adopted a similar approach as [3] for extracting motion information, but we overcome the limitations of the previous approaches by employing rules for extracting highly accurate and reliable particle trajectories.

3. Proposed Framework

In this paper, we propose an approach for crowd behavior understanding adopting two novel algorithms, the first able to generate long, dense, reliable and accurate pedestrian trajectories and the second clustering them to generate long term reliable and abstract information describing flows in the whole video. The final results provide directly information characterizing flows but they also represent a starting point for further high-level analyses of crowd behavior. As shown in Figure 1, the approach starts by dividing the input video into multiple *segments* of equal length and duration, considering videos with a constant frame rate. The initial frame of each segment is overlaid by a grid of particles initializing a dynamical system defined by optical flow, as discussed by [3]. Time integration of the dynamical system over a segment of the video provides particle trajectories (*tracklets*) that represent motion patterns in the scene for a certain time interval associated to the analyzed segment. We detect sources, sinks and main flows in the segment (for sake of brevity sometimes we will refer to this information as segment *local track*) by analyzing motion patterns followed by clusters of tracklets, obtained using an unsupervised hierarchical clustering algorithm, where the similarity is measured by the Longest Common Sub-sequence (LCS) metric. Results achieved so far are intrinsically related to a single segment of the analyzed video, associated to a relatively short time frame. To achieve final *global tracks*, covering all the video, we cluster the achieved local tracks through the same hierarchical clustering algorithm. Our main contributions are:

1. generating dense and long trajectories,

2. identifying sources and sinks,
3. understanding behavior of the crowd in the scene by considering full length video,
4. achieve the above results without requiring object detection, tracking, nor training, targeting employment in naturalistic conditions.

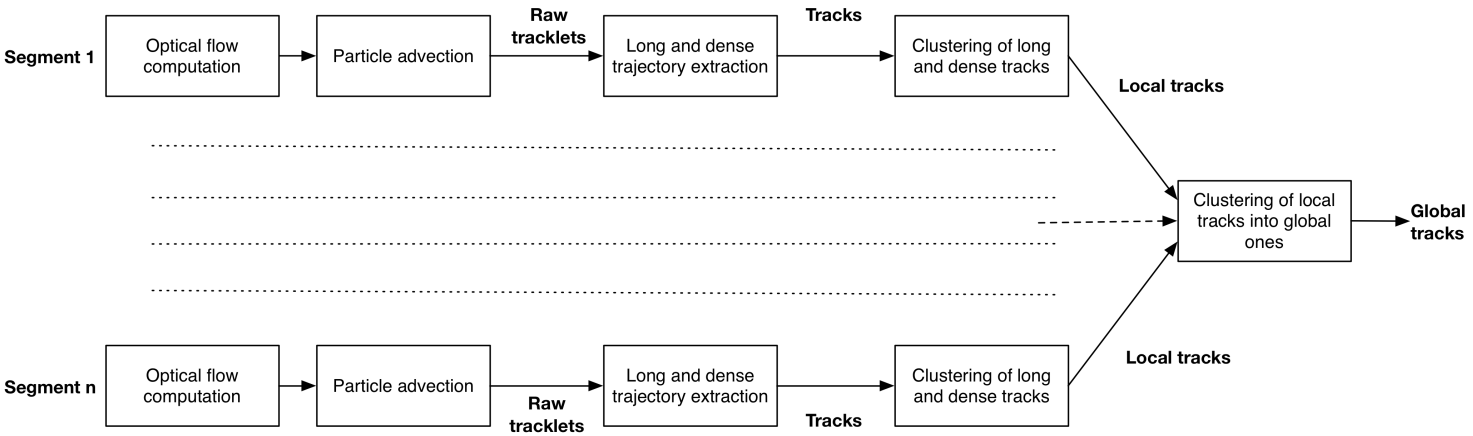
3.1. Achieving Reliable Descriptive Motion Information

The input to our framework is a sequence of frames and, as summarized in the previous section, a first phase of the overall approach is aimed at achieving reliable descriptive motion information that will be then further processed to obtain local and global tracks. As already mentioned, we adopt an overall divide-and-conquer approach, splitting the overall frame sequence into n segments, each containing k frames. We then perform a segment local analysis to achieve tracklets that will be clustered later.

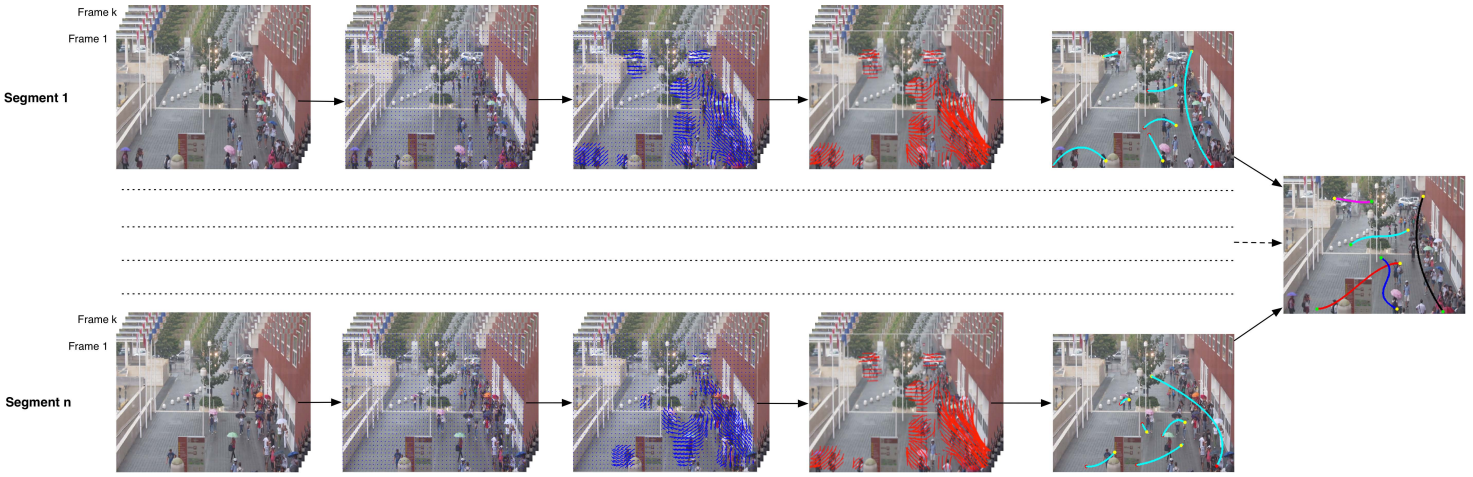
The first step to achieve tracklets is the computation of dense optical flow between two consecutive frame of every segment. We employ the method proposed by [29] where gray value constancy, gradient constancy, smoothness, and multi-scale constraints were used to compute highly accurate optical flow. Consider a feature point i in the frame associated to time t of a segment: its flow vector $Z_{i,t} = (X_{i,t}, V_{i,t})$ includes its location $X_{i,t} = (x_{i,t}, y_{i,t})$ and its velocity vector $V_{i,t} = (v_{x_{i,t}}, v_{y_{i,t}})$ (i.e. the velocity vector is made up of the change in the horizontal and vertical positions); moreover, for each feature point, we can compute θ_i, that is the angle or direction of V_i, where $0^\circ \leq \theta \leq 360^\circ$. Then $\{Z_1, Z_2, \ldots, Z_m\}$ is the motion flow field of all the foreground points of an image.

We can thus initialize a continuous dynamical system in which the velocity of a point at time t is essentially related to the optical flow of the same point, which is given by equation 1

$$V_{i,t} = F(X_{i,t}) \tag{1}$$



(a) Block diagram of the proposed approach.



(b) Sample frames of the application of the approach to the case study.

Figure 1: The proposed framework for sources and sinks detection.

3.2. Particle Advection

The next step is to advect a grid of particles over the optical flow field, that corresponds to the time interval 1 to T for each segment. We launch a grid of particles over the first optical flow field of every segment and each initial position of the particle represents the source point. Ideally, the grid should have the same resolution of the frame and size of the particle is same as size of the pixel; nonetheless this would imply huge computational costs. To avoid this problem, we reduce the resolution of the grid by dividing it by a non negative constant: consider $res_x \times res_y$ the resolution of the image and $c > 1$; the resulting grid G will have a size $g_x \times g_y$ where $g_x = res_x/c$ and $g_y = res_y/c$.

Considering the initial location $X_{i,t} = (x_{i,t}, y_{i,t})$ of particles with $i \in G$, their next location $X_{i,t+1}$ at time $t+1$ can be computed by numerically solving the system of equations achieved by considering equation 1 for all the particles in G by using following approximation:

$$X_{(i,t+1)} = F(X_{(i,t)}) + X_{(i,t)} \tag{2}$$

To achieve a trajectory Ω_i for every particle $i \in G$, taking the form $\Omega_i = \{X_{i,1}, \ldots, X_{i,T}\}$, where T is the integration time, with $T = k$ (we will use time and frame number interchangeably), we need to compute a pair of flow maps ψ_x and ψ_y. These maps contain the initial position of each particle and all the subsequent positions computed according to the above equation, as discussed in [3].

The trajectory achieve by means of this process represents a movement from the initial position through time (and through frames) according to the optical flow. However, when this kind of analysis is carried out on an unstructured crowded scene (e.g. a subway corridor with pedestrians getting out and in a platform), where people move towards different and potentially changing directions, in many cases the particle trajectory could drift from a flow of pedestrians characterized by a certain direction to a spatially close but distinct and different flow, moving towards a significantly different direction. In this case, the

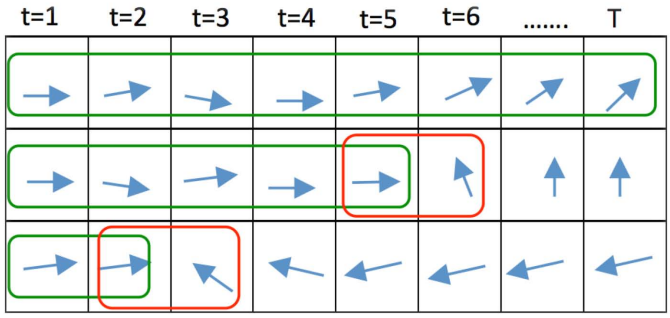


Figure 2: Flow associated to different particles: the first one is considered legitimate throughout the whole segment, whereas the second and third are trimmed due to significant changes in the flow direction in intermediate frames.

trajectory is erroneous, since pedestrians do not actually change direction so quickly, and this can effect the final outcome. Consider, for instance, Figure 2: each row shows the flow information at a given time for a given particle; the first row exemplifies a normal and legitimate trajectory, whereas the second and third rows show a situation that we consider an error, since the direction of the optical flow associated to the particle violently changes in too little time. The second and third particles, therefore, according to our approach will generate much shorter tracklets than those generated by current approaches (such as [3]) in which these changes are accepted.

More precisely, to avoid the above introduced defect, we modify equation 2 in the following way:

$$X_{(i,t+1)} = X_{(i,t)} + F(X_{(i,t)}) * B_i \tag{3}$$

$$B_i = \begin{cases} 1, & \text{if } \| \theta_{i,1} - \theta_{i,t} \|_2 < \lambda \\ 0 & \text{otherwise} \end{cases}$$

The particle, therefore, continues moving forward if circular distance [30] between its initial direction $\theta_{i,1}$ computed initially and its direction at time t, $\theta_{i,t}$ is less than a specified threshold λ.

This approach, avoids errors due to particles drifting from a pedestrian flow

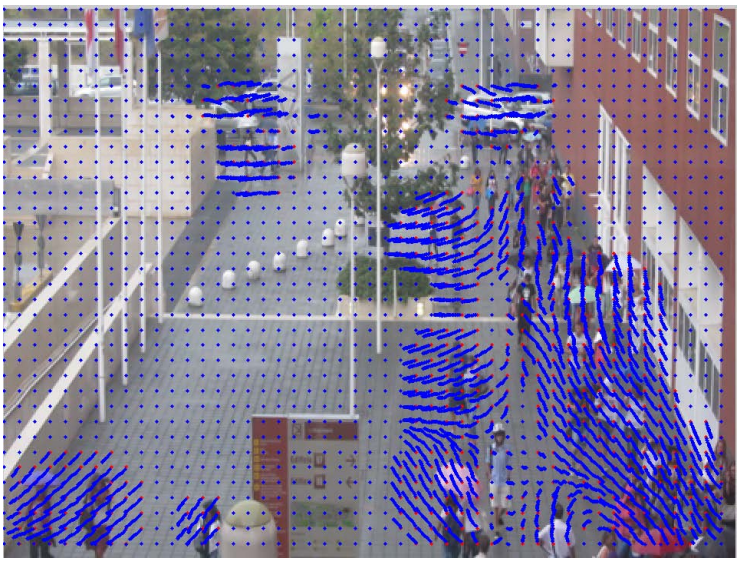


Figure 3: Tracklets achieved after particle advection.

to a different one, however the achieved trajectories are in general shorter in length than those extracted by [3]. In the previously mentioned Figure 2, the length of the extracted tracklets is equal to T frames only for the first particle whereas the other approach would always lead to tracklets of T frames: the number of frames of a tracklet is not necessarily an indicator of the actual length of the associated trajectory, but limiting the number of tracked frames inevitably leads to achieving shorter trajectories.

After particle advection, short duration particle trajectories called *tracklets* are obtained as shown in Figure 3. Some of these tracklets correspond to the background of the scene or noise and they are not actually part of our analysis. Therefore, in order to remove these tracklets, we estimate their actual length by computing the euclidean distance between the start and end points (remind that abrupt changes in direction of the particles block the trajectory construction, so most of the tracklets are very close to straight lines). We discard those tracklets for which $\| (x_i^1, y_i^1) - (x_i^T, y_i^T) \|_2 < \delta$ (i.e. those tracklets whose length is very likely lower than a given threshold δ).

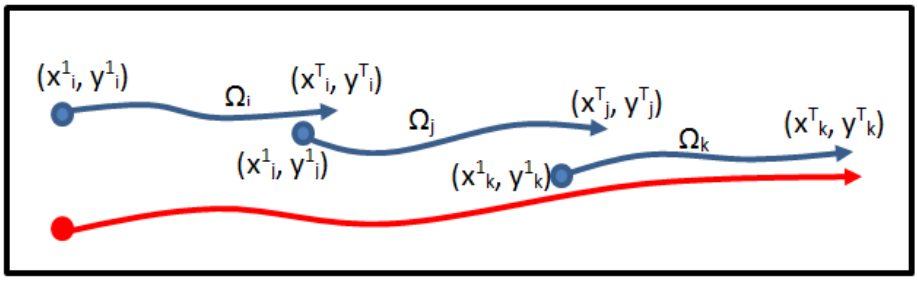


Figure 4: Example situation of generation of a long track from small tracklets.

3.3. Clustering Tracklets to achieve Local/Global Tracks

Tracklets extracted through particle advection fail to represent important characteristics of the overall motion and they provide inadequate information for identifying source and sink points of the dominant flows and also for understanding behavior of the scene. To achieve these goals, we need dense and long trajectories which we can obtain thanks to the following assumptions: (i) a large number of tracklets corresponding to dominant flows is identified by the previous phase; (ii) source and sink points of tracklets associated to a common flow are often spatially close to each other. Our goal is to combine these accurate but generally short tracklets into longer trajectories. This induces a combinatorial matching problem that we define and solve *recursively* for all tracklets detected for each segment of the video sequence. The example frame shown in Figure 3 intuitively supports the claim that, for most scenes including a relatively large number of moving pedestrians, these two assumptions generally hold.

Let us now focus on the implications of the second assumption. Some of the achieved tracklets corresponding to single movements can be quite similar (in orientation), but their sources and sinks can be spatially different. Our goal is to combine similar tracklets into longer trajectories. For example, consider three tracklets Ω_i with source point (x_i^1, y_i^1) and sink point (x_i^T, y_i^T), Ω_j with source point (x_j^1, y_j^1) and sink point (x_j^T, y_j^T), and Ω_k with source point (x_k^1, y_k^1) and sink point (x_k^T, y_k^T) as shown in Figure 4. These tracklets start and end at different locations but the sink of one of them is spatially very close to the source

of a different one: for instance, tracklet Ω_j starts very close to the sink point of Ω_i and the source of Ω_k is close to sink point of Ω_j. We exploit this spatial closeness of the tracklets in order to obtain longer trajectories. The similarity among the tracklets is computed by employing longest common sub-sequence algorithm, which will be discussed in details in the next section. The rationale of the approach, however, is that similar tracklets are identified and combined by means of k^{th} order least square polynomial regression as exemplified by the red line in Figure 4 (computed with $k = 3$). The outcome of the process, the red track, is therefore a long trajectory with source point (x_i^1, y_i^1) and sink point (x_k^T, y_k^T).

Let us now more formally define the above intuitive approach to achieve long tracks from shorter tracklets. First of all, we call a tracklet for which we would like to extract a longer trajectory a *query tracklet*. Let us now consider the analyzed scene: we have already overlaid a grid for particle advection, organizing the scene in "cells". The query tracklet will be positioned in a cell c, and we can define *neighbor tracklets* those ones positioned in the the Moore neighborhood of c. Finally, we call *candidate tracklets* those ones that we are considering for extending the query tracklet. The pseudo-code of the proposed algorithm is presented in Algorithm 1 and its description is reported here below.

The input to the algorithm is the *query tracklet* and output is the long trajectory; we also assume that the overall grid including the other tracklets is available as global information. The function *LongTrajectory* is divided into two steps: first of all, we collect all the tracklets that, due to to spatial arrangement, represent a plausibly connected path, but we also filter out tracklets that are not sufficiently similar to the starting one, because the resulting track would present an abrupt change of direction. This operation is executed by the *CompleteTrajectory* function that operates on an array L initialized by the calling environment (lines 2 and 3 of *LongTrajectory*) as containing only the query tracklet Ω_q. The function considers the tracklets present in the neighborhood of the query tracklet, and evaluates if they could represent a plausible continuation of the related path (line 4), inserting them into an array of candidate tracklets.

If this array is not empty, the candidate tracklet that best matches the query one (line 9) is selected, inserted in the array L and then the *CompleteTrajectory* is recursively called considering the added query as next starting point (line 13). When the candidate tracklets array is empty the algorithm ends, returning the array L containing the tracklets that were added during the recursive execution. Finally, the resulting set of tracklets is then combined by means of k^{th} order least square polynomial regression (line 5 of *LongTrajectory*).

This function is applied to tracklets positioned in every cell of the grid. Some of them will basically not be extended at all; moreover some of the achieved tracks will be actually very similar to portions of larger ones: by definition of the algorithm, in fact, the existence a track spanning across $k > 2$ cells makes it very likely that additional $k - 2$ shorter (but not atomic) tracks are achieved later on, considering cells explored during the first computation.

The set of achieved tracks still contains also short tracklets, for which no extension was possible. The goal of this step, however, is to obtain dense and long trajectories covering all the scene and representing the most significant motion patterns, therefore we can filter out tracks that are shorter than a given threshold, analogously as we did to remove noise in the particle advection step (in this case the euclidean distance estimation of the actual length of the track is even more plausible since the considered tracks are, by construction, quite smooth).

Figure 5 shows the achieved long and dense tracks with increasing thresholds: the number of tracks decreases with growth of the threshold, but trajectories are still dense enough to represent whole motion of the scene even at the higher thresholds. Even though it is of course important to avoid setting a threshold so high that tracks representing important flows are filtered, it must be noted that the reduction of the number of tracks simplifies the computation associated to subsequent steps of the overall approach without suppressing important information.

After achieving dense tracks, the next step is to combine similar tracks into local tracks by adopting novel hierarchical clustering algorithm. The classical

Algorithm 1 Generating long tracks starting from tracklets

Input: tracklet Ω_q

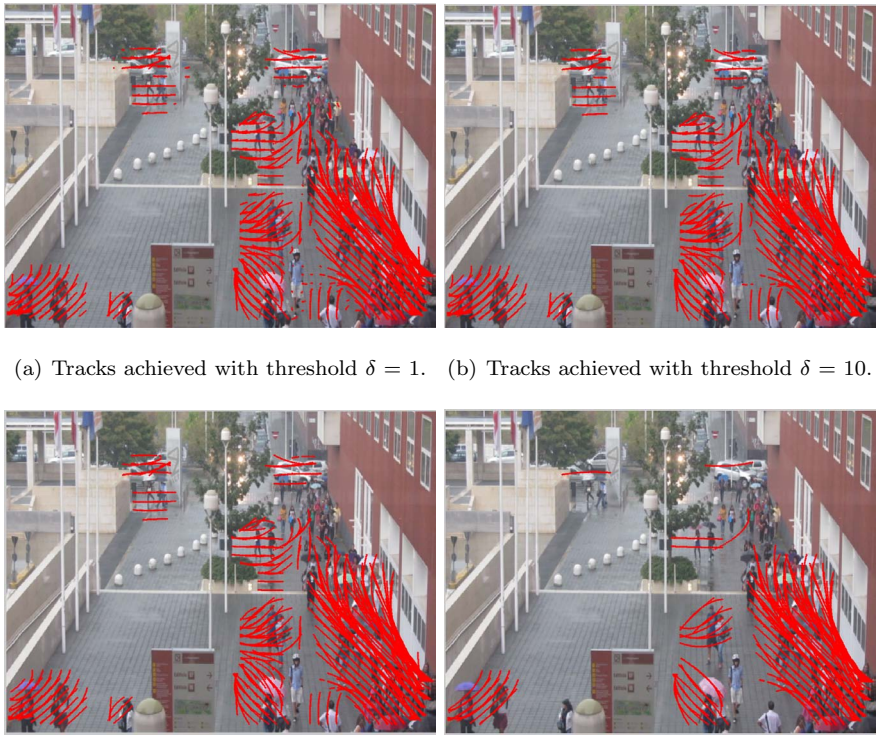
Output: track L_t

function LONGTRAJECTORY(Ω_q)
    initialise array L as empty
    insert Ω_q in first position of L
    L = completeTrajectory(Ω_q,L)
    return polynomial regression on L
end function

Input: tracklet Ω_q**, array of tracklets** L

Output: array of tracklets L

function COMPLETETRAJECTORY(Ω_q, L)
    initialise array C
    for all tracket t in neighborhood of Ω_q **do**
        if $\|X_q^T - X_t^1\| \leq \epsilon$ **then**
            insert t in C
        end if
    end for
    if C is not empty **then**
        $bestCandidate = \arg\max_{c \in C} \frac{LCS(L[0],c)}{min(Len(L[0]),Len(c))}$
        $match = \frac{LCS(L[0],bestCandidate)}{min(Len(L[0]),Len(bestCandidate))}$
        if $match > m_t$ **then**
            insert $bestCandidate$ in tail of L
            return CompleteTrajectory($bestCandidate$,L)
        end if
    else
        return L
    end if
end function



(a) Tracks achieved with threshold $\delta = 1$. (b) Tracks achieved with threshold $\delta = 10$.

(c) Tracks achieved with threshold $\delta = 20$. (d) Tracks achieved with threshold $\delta = 50$.

Figure 5: Tracks achieved after the application of the algorithm to generate long tracks from shorter tracklets with increasing length thresholds.

supervised clustering algorithms can not be used as the number of flows (and therefore desired clusters) are unknown. Therefore, we propose a novel hierarchical clustering algorithm, based on the following procedure.

1. We sort the tracklets in descending order on the basis of their length; in particular, let L $= \{T_1, T_2, \ldots, T_k\}$ represent the sorted list of tracklets and $\{l_1, l_2, \ldots, l_k\}$ the respective length of tracklets, we have than $l_i < l_j$ with $1 \leq i < j \leq k$.
2. We set up a list of tracklets to be considered L_T, which initially is the complete list of tracklets excluding T_1; we also set up a list of clusters L_C, initially containing a single cluster associated to the first tracklet T_1 (the longest one) that is also used as initial cluster center;

3. We select the shortest tracklet from the list L_T, say T_s, and compare it with the centers of all clusters present in L_C using longest common sub-sequence metric, that will be described in the following Section, to compute similarity measure. If this value is greater than a threshold ϵ, then tracklet T_s is assigned to the currently considered cluster, otherwise, a new cluster is inserted in L_C, tracklet T_s is assigned to it and set as its center. We delete the tracklet T_s from the list L_T after assignment to a cluster.
4. If the cluster's size exceeds a positive value of S we consider that sufficient information about the associated flow has already been achieved; therefore we identify source and sink location and update the center of the cluster by using K^{th} order least square polynomial regression. We used $S = 30$ in our experiments. The cluster source and sink are selected according to a simple procedure: the selected pair is made up of the source point of a tracklet and the sink point of (generally another) tracklet that are part of the cluster and, in particular, the pair for which the euclidean distance between source and sink is maximum.
5. We repeat the previous step until L_T is not empty.

A MatLab-like implementation of the above clustering algorithm is described in Algorithm 2.

3.4. Longest Common SubSequence Computation

At this stage, we define similarity measure for comparing and clustering similar trajectories. There are number of approaches for measuring similarity of the moving object trajectories, such as [31] and [32]. A survey of different similarity measures for trajectory clustering is reported by [33]: Euclidean and Dynamic Time Warping (DTW) approaches are more sensitive to noise whereas Longest Common Sub-Sequence is efficient for series of unequal lengths and it is more robust to noise and outliers than DTW, as discussed by [34] and by [35].

The key idea of LCS is to match two time-series of tracklets by not considering all points of the tracklets, that can, to a certain extent, have different

lengths. The following procedure allows verifying to which extent two trajectories can be considered similar (or *matching*, according to a certain similarity measure) and therefore what is the longest portion they have in common.

Let T_1 and T_2 represent two tracklets with size n and m respectively: $T_1 = \{(x_t, y_t), t = 1, ..., n\}$ and T_2 having analogous structure but m elements; with $T_1(i)$ we denote (x_i, y_i) with $0 \leq i \leq n$ and analogously for T_2. We compute the similarity among two tracklets by recursively finding a matching M between portions of these trajectories using a dynamic programming procedure that we will only briefly introduce for sake of space. Two constants are needed, respectively Φ controlling matching sequences in time, determine as $\Phi = \frac{\max(Length(T_1), Length(T_2))}{2}$ and Ω which is the spatial matching threshold. Formally the matching matrix M comparing T_1 and T_2 can be computed recursively as follows:

$$M_{i,j} = \begin{cases} 0, & \text{if } i \text{ or } j \text{ are } 0 \\ 1 + M_{i-1,j-1}, & \text{if } \| \ T_1(i) \text{ - } T_2(j) \ \|_2 < \Omega \text{ and } \mid i - j \mid < \Phi \\ max(M_{i-1,j}, M_{i,j-1}), & \text{otherwise} \end{cases}$$

The similarity measure between two tracklets T_1 and T_2 is therefore $S(T_1, T_2) = \frac{LCS(T_1,T_2)}{min(n,m)}$, where LCS is the number of matching points between T_1 and T_2, according to the above matching matrix.

4. Experimental Results

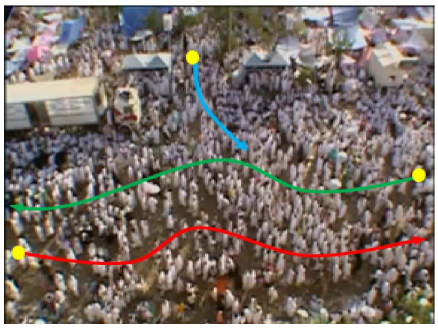
This section presents qualitative and quantitative analyses of the results obtained from experiments on the application of the proposed approach to video sequences made available from other research groups and acquired through field observations. In particular, we carried out our experiments on a PC of 2.6 GHz (Core i5) with 4.0 GB memory, running a Matlab implementation of the presented algorithms; the analyzed data set includes videos made available from other research groups and described in [36, 27], in addition to videos we acquired

in past researches described in [37, 38]. The overall set of video includes situations including both the so called structured and unstructured crowds [39](i.e. situations with respectively stable and varying flows in the scene), and very different density conditions.

The analyzed videos we will discuss in the remainder of the section are the following:

- *airport* video, Figure 6(a) [36]: this sequence shows a portion of an airport, including stairs and escalators, with relatively stable pedestrian flows in medium-low density conditions;
- *Hajj* video, Figure 6(b) [36]: this sequence was taken in the context of the yearly pilgrimage to Makkah, Saudi Arabia, and it shows a very high density situation in which the overall velocity of pedestrians is very low but characterized by three main and relatively stable movement directions;
- *station* video, Figure 6(c) [27]: this footage shows a platform in which pedestrians try to get on and off of a train; flows change in time due to the congestion that arises near one of the entrances of the wagon, and the density conditions are very different in distinct areas of the scene;
- *escalator* video, Figure 6(d) [27]: this is a footage of a portion of a platform in which two main flows lead to and from an escalator; the density conditions are medium-low and the flows are quite stable, although occlusions due to the presence of a column and other infrastructural elements are present in the scene;
- *university* video, Figure 6(e) [37]: this sequence shows the arrival of students that are going to undertake an admission test to a bachelor course at the University of Milano-Bicocca; the density conditions are medium-low but the number of pixels per person is quite low and many occlusions are possible also due to the presence of infrastructural elements;
- *gallery* video, Figure 6(f) [38]: this footage was taken in a commercial/turistic gallery in Milan's city center, in a Saturday afternoon; the

(a) Airport video and ground truth.



(b) Hajj video and ground truth.

(c) Station video and ground truth.

(d) Escalator video and ground truth.

(e) University video and ground truth.

(f) Gallery video and ground truth.

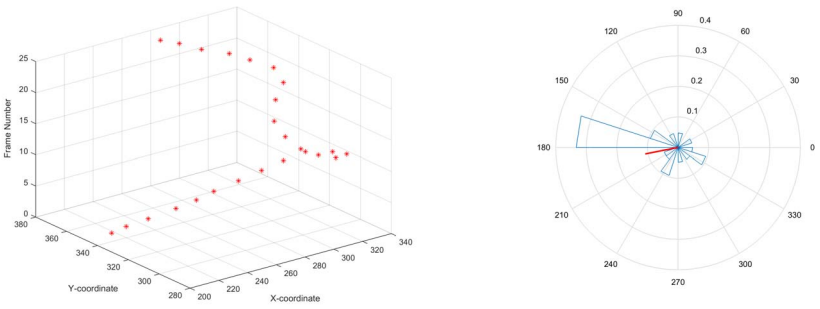
Figure 6: Dataset of analysed videos and associated manually defined ground truth.

density conditions are medium-high and the point of vantage causes a very high number of occlusions, also due to the irregular and varying nature of pedestrian flows.

All of the above figures also report a manual annotation describing the dominant flows identified by a human observer, that can be qualitatively compared to the achieved results, that will be presented later on. Different color codes are used for representing different flows while source points are always marked with yellow circles.

Since our framework consists of two major parts, the first aimed at generating dense long trajectories from short and accurate ones, the second able to detect sources and sinks of dominant flows, we describe two types of experiments. In the following subsection, we compare our method for the extraction of long and dense trajectories with baseline tracking techniques, in particular we consider KLT and SIFT based trajectories by analyzing all of the above mentioned videos. In this case, we adopt both a qualitative and quantitative approach, by showing the generated trajectories and also by comparing the number of trajectories extracted employing different thresholds for their length, to evaluate the capability of the approach to generate sufficiently long trajectories.

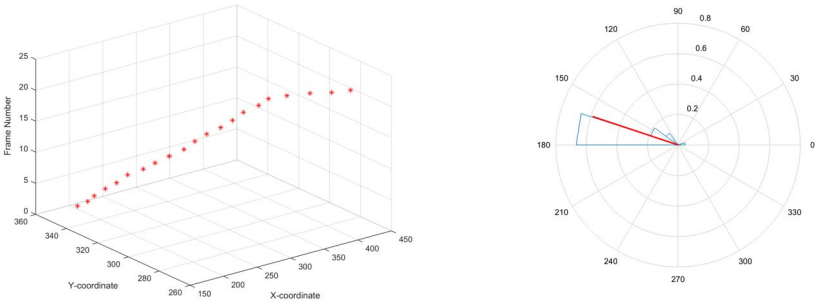
In subsection 4.2, instead, we describe the overall results about the detection of sources and sinks of dominant flows and we discuss them considering results achieved in those situations by state-of-the-art methods.



(a) Temporal Plot of Trajectory.

(b) Histogram of Trajectory.

Figure 7: Trajectory in Error.



(a) Temporal Plot of Trajectory.

(b) Histogram of Trajectory.

Figure 8: Stable Trajectory.

4.1. Comparison with Baseline Trajectories Extraction Approaches

In order to evaluate improvement obtained with our proposed framework, we compare our method of generating long and dense trajectories with state of the art trackers considered as a baseline: in particular, we consider KLT trajectories adopted by [27, 28, 17, 19], SIFT trajectories adopted by [19, 39], approach described by Solmaz et al. in [3].

Due to the unavailability of consolidated ground-truth data for this kind of application, it is difficult to evaluate and compare the precision and, more generally speaking, *quality* of the results achieved by the proposed approach and baseline trackers. We propose here a combination of different quantitative and qualitative measurements both in the above mentioned videos and in additional situations.

4.1.1. Quantitative Analysis

In particular, we first examine the performance of the above approaches when analyzing a synthetic rendered video. High quality rendered videos should incorporate deforming objects, complex light reflectance, camera motion, optical artifacts which make mimicking the real world videos very hard and challenging. However, our intention with this test is to evaluate the ability of the above

approaches to consider some background considerations and knowledge about pedestrian movement employing an extremely simple video including a particle following different trajectories, to isolate the conceptual analysis the related paths from technical issues of trackers. In fact, trajectories extracted from complex videos in crowded environments imply errors and noise due to the severe occlusions and we want to be able to filter out erroneous paths.

Trajectories belonging to one motion pattern (e.g. the trajectory of the head of a pedestrian) may drift and become the part of different motion patterns (e.g. the trajectory of a body part of another pedestrian moving in a different direction). We call these trajectories as erroneous or occluded. The effect of this kind of occlusion is schematized in Figure 7: in particular, Figure 7(a) plots a trajectory extracted from a 25 frames synthetic video of a simple particle moving in the captured area, while Figure 7(b) shows the orientation histogram of the trajectory. Since the frame rate of the video is 25 frames per second, this trajectory is associated to just one second and therefore, considering normal human locomotion, it should not present a wide variety of orientations, but rather a main direction with relatively little changes. The orientation histogram in Figure 7(b) instead reports a wide range of orientations, highlighting the fact that the trajectory either belongs to noise or occluded with different motion patterns. In contrast, a more stable trajectory is shown in Figure 8, and it is characterized by a limited range of orientations as shown in Figure 8(b). On the basis of these considerations, we defined a plausibility test for each individual trajectory extracted from the proposed approach and other baseline methods. For computing this plausibility factor for a given trajectory $T\{X, \theta\}$, where X represents the spatial locations and θ represents orientations of the trajectory, with T containing k points, we perform following steps

1. Compute circular mean, i.e, θ_μ of θ as in [40] for the given trajectory T.
2. Compute circular distance from the mean for all trajectory points, i.e, $CircDist_i = (\theta_\mu - \theta_i)$, with $0 \leq i < k$ and where θ_i is i^{th} point of the trajectory.

3. Compute an indicator of smoothness

$$Smooth_i = \begin{cases} 1, & \text{if } (\ \theta_\mu \text{ - } \theta_i\ ) < \Psi \text{ where } \Psi \text{ is set to } 0.7854 \\ 0 & \text{otherwise} \end{cases}$$

4. Compute the overall trajectory smoothness indicator

$$Smooth = \frac{\sum_{i=0}^{k-1} Smooth_i}{k}$$

We consider a trajectory T as accurate if its smoothness indicator $Smooth \geq \gamma$, where γ is set to 0.5 in all the following experiments. This kind of test, for instance, would label as accurate the trajectory in Figure 8 but not the one in Figure 7.

By following this procedure we performed plausibility tests on all other trajectories extracted from the real world analyzed videos. We then compute an overall plausibility rate by computing the ratio between the number of accurate trajectories and the total number of extracted trajectories. The mean plausibility rate, mean length, minimum and maximum lengths of extracted trajectories for all methods using all the analyzed videos is summarized in Table 4.1.1. Results show that the sparse methods, i.e, KLT and $SIFT$, produce a relatively low number of trajectories compared to other particle based dense methods. KLT trajectories have approximately the same mean length as the dense methods but the plausibility is relatively lower. In case of $SIFT$, the mean precision rate is high but the mean length of the trajectories is too small as compared to other methods, and therefore, the trajectories extracted by this method could not be able to capture the whole motion of the scene. We also run Solmaz et. al [3] and our algorithm with the same configuration and by initializing the same number of particles for all the analyzed videos. The plausibility rate of trajectories extracted by our Short Dense Trajectories (SDT) method discussed in section 3.1 is very high relative to other methods but with approximately the same mean length, that would also be insufficient to describe the whole motion in the scene. We improve the mean length of trajectories by employing our Long Dense Trajectories (LDT) method discussed in section 3.3 by paying a small cost

in terms of plausibility. In fact, plausibility rate for LDT is reduced because the tracks are clustered based on the similarity measure 3.4, which implies the potential connection of tracklets leading to a change in the flow direction and therefore to a less smooth but still plausible trajectory.

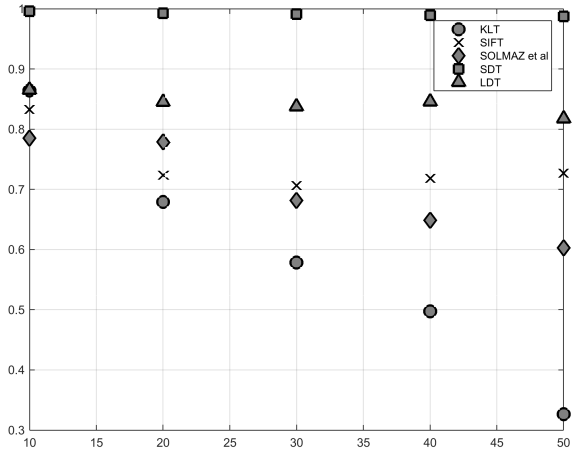
We further investigate the variation of performance of our and baseline methods with a changing segment size of the analyzed videos. We divide each analyzed video into five segments of different size, ranging from 10 to 50 frames. For each segment, we extract features (in case of $SIFT$ and KLT) or initialize particles (in case of other methods) in the first frame and tracked through last frame. In case of LDT, we extract tracks for each segment and then apply algorithm 2. The results of this analysis on the mean plausibility rate are shown in Figure 9(a): we observe that it generally drops with the growth of the segment size for all approaches but DLT whose plausibility decreases only slightly. As we already discussed, the plausibility rate of DLT is lower then DST but still higher than other methods. The mean length plot for the same analysis is shown in Figure 9(b): we observe that mean length slightly increases with the growth of segment size, but for DLT it remains almost constant. This means that this method is able to capture global motion information in the scene also with relatively small segments.

Table 1: Summary of mean plausibility and mean length of different methods

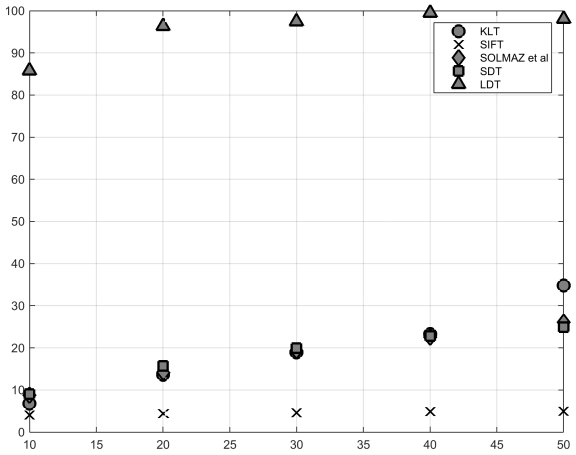
Methods	# of traj.	plausib.	mean length	max_length	min_length
KLT	2576	0.4973	23.2686	73.0258	2.0142
SIFT	3636	0.7268	4.9273	58.7974	2.0031
Solmaz et al	7633	0.6027	26.0615	86.8846	2.0104
SDT	7633	0.9876	24.9281	87.0447	2.0075
LDT	7633	0.8173	98.0238	320.8929	2.2279

4.1.2. Qualitative Analysis

The qualitative analyses will translate into understandable examples the implications of the above quantitative analysis.



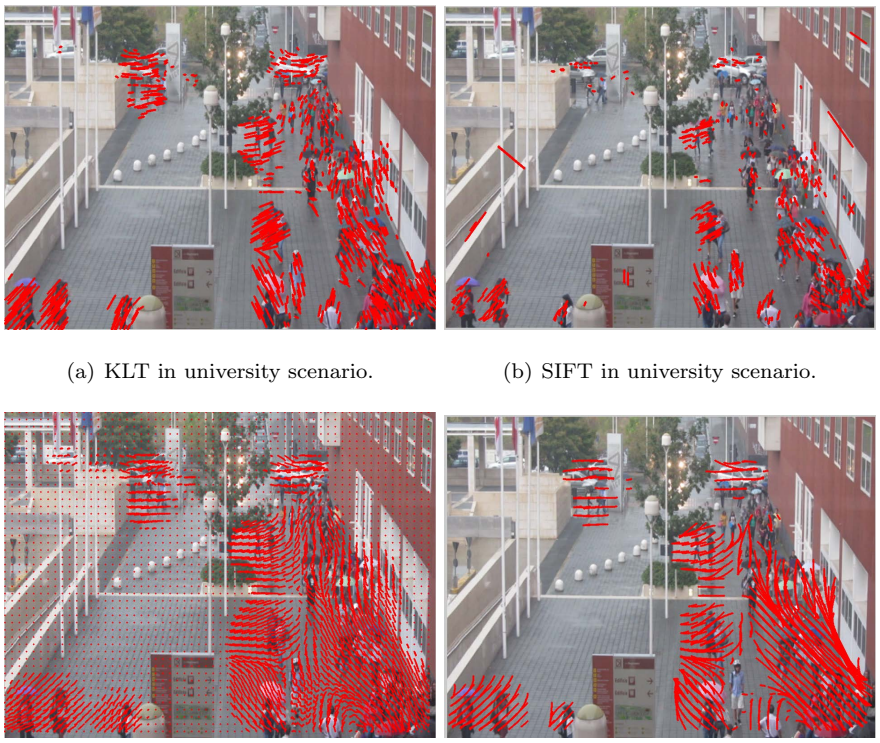
(a) Mean Plausibility Rate Plot.



(b) Mean Length Plot.

Figure 9: Evaluation of mean plausibility and length of trajectories with different segment size.

In order to obtain KLT trajectories, we first identify low-level features (corner points) in the initial frame using standard Shi-Tomasi-Kanade detector [41]. These corner points are tracked over time by using [42].



(a) KLT in university scenario.

(b) SIFT in university scenario.

(c) Solmaz et al. in university scenario.

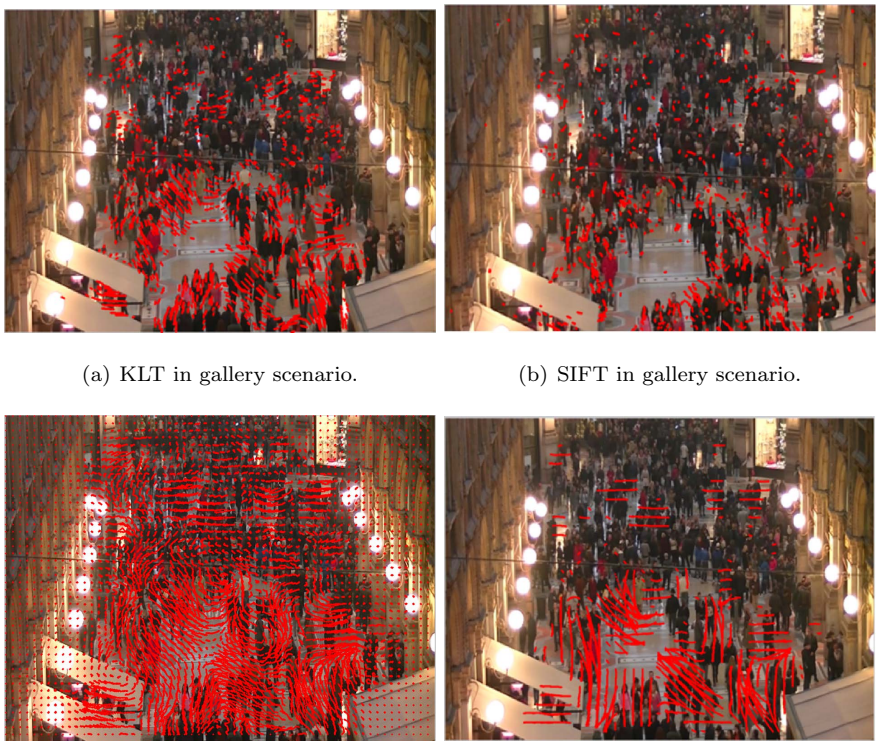
(d) Proposed approach in university scenario.

Figure 10: Comparison between state of the art trackers (KLT and SIFT) and the proposed approach in the university scenario.

On the other hand, in order to extract SIFT trajectories, we first extract SIFT interest points from the initial frame; these points are then tracked through multiple frames by matching euclidean distance between SIFT descriptors within a neighborhood. More details about SIFT feature tracking can be found in [43].

Finally, trajectories generated by the application of [3] are more aimed at supporting crowd behavior understanding rather than implementing a traditional tracking system; due to this perspective, they represent the closest approach to the one we are proposing.

Before providing a quantitative analysis of the performances of the above approaches, a qualitative comparison in the university and gallery videos is



(a) KLT in gallery scenario.

(b) SIFT in gallery scenario.

(c) Solmaz et al. in gallery scenario.

(d) Proposed approach in gallery scenario.

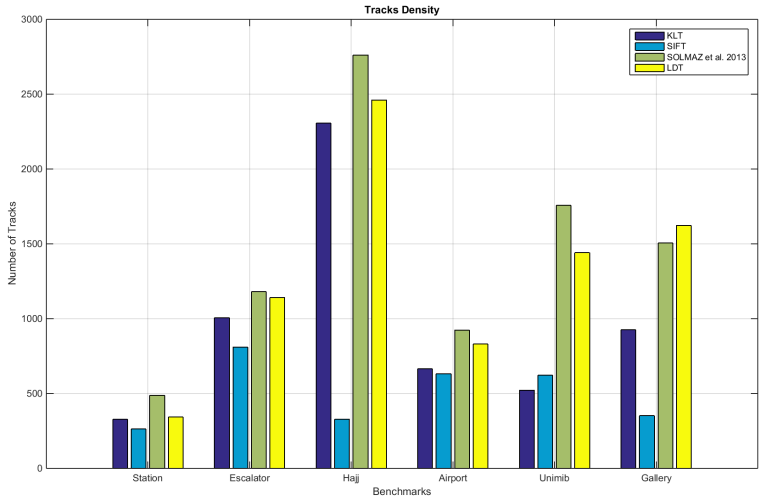
Figure 11: Comparison between state of the art trackers (KLT and SIFT) and the proposed approach in the gallery scenario.

provided in Figures 10 and 11: in the university scenario, SIFT is actually unable to generate trajectories in good accordance with the ground truth, and it generates even noticeable false positives, whereas KLT is able to generate plausible but short trajectories, due to the fact that features that are used for tracking are not visible in every frame. The approach of Solmaz et al. [3], instead, produces results that are relatively similar to the trajectories generated by the proposed approach, although the trajectories are generally shorter and sometimes erroneous (i.e. continuous but associated to paths that are not really associated to real pedestrian flows). This difference is due to additional rules in our approach that avoid the generation of long tracks when base tracklets have different orientation, and it is even more apparent in the gallery scenario. In

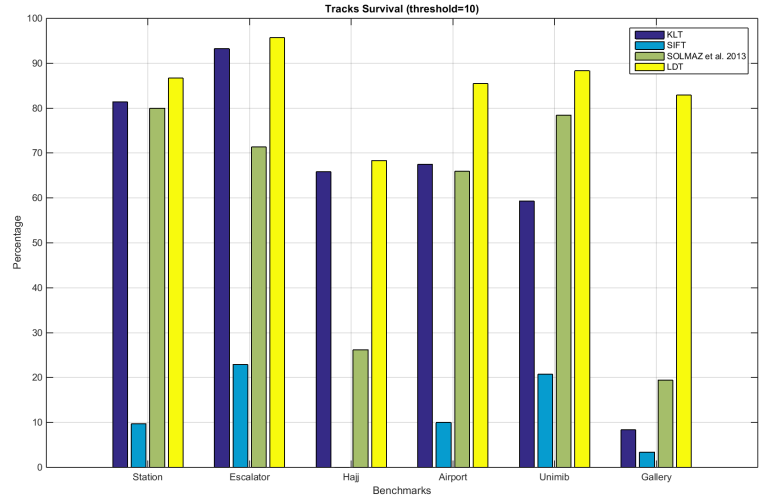
this situation both SIFT and KLT fail, since this video is extremely problematic for feature–based approaches due to dynamic occlusions and clutter, whereas Solmaz et al. [3] produces an extremely high number of tracks basically due to the fact that optical flow in the walkable area of the gallery is dense and in a large number of varying directions. The additional rules for filtering non plausible trajectories we included in the proposed approach are instead able to consistently reduce this noise.

As above two trackers produce trajectories based on feature points extracted from the initial frame of video segment, therefore these trajectories are sparse. Another problem with feature base trajectories is that in high density situations, due to complex movement of people, and due to severe inter and intra object occlusions, feature points can not be tracked for long period of time. Therefore, in high density situations, feature base trajectories are short. These short and sparse feature based trajectories are inadequate to capture crowd dynamics.

For a quantitative comparison of the results of these approaches, we report in Figure 12(a) a graphs describing the track density in different scenarios, that is, the raw number of tracks generated by the different approaches after removing noise and tracklets whose length is less than 2. Per se, this metric is not actually an indicator of success of the approach, nonetheless the very low number of tracks generated by the SIFT approach is an indicator that it is simply unable to grasp the fundamental motion information of the scene. In Figure 12(b), instead, we show the percentage of the above tracks that are longer that a threshold set to 10: once again, SIFT is not adequate to this task since even if the produces tracks are few, most of the produced ones are not even long. The other approaches perform similarly in most of the scenarios, in case of medium-low density and/or structured crowds (i.e. with flows of pedestrians that are neatly separated and generally stable), whereas some difference can be highlighted in the university and gallery videos. In these cases, flows of pedestrians actually mix and cause occlusions (generating problems to KLT) and a very high number of possible ways of connecting optical flow tracklets (for Solmaz et al. [3]).



(a) Track density in the analyzed videos.



(b) Track survival with threshold 10 in the analyzed videos.

Figure 12: Quantitative analyses about track density and survival comparing the proposed approach and state-of-the-art tracking approaches.

To further characterize these differences, we also report in two extremely

different scenarios, station and gallery, the variation in the survival rate of generated tracks with the growth of the length threshold. Results of this analysis are shown in Figure 13: while for the station video Solmaz et al. [3] produce a percentage of surviving tracks that is very similar to our approach, and quite higher than both SIFT and KLT, in the gallery scenario the difference between the survival tracks ratio is already significant for a length threshold set to 10.

4.2. Identification of Sources and Sinks, and Characterization of Dominant Flows

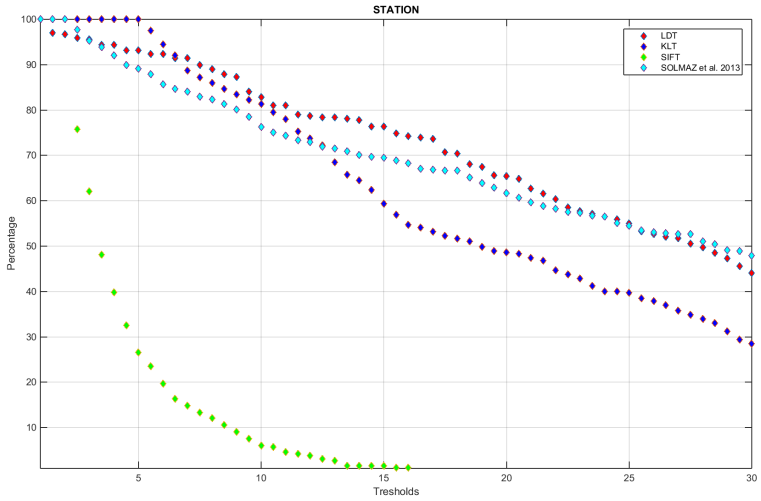
In this section, we present results of our proposed framework with reference to the capability to identify sources, sinks (as defined in the introduction) and in general to characterize pedestrian flows in the scene. We analyze different videos to highlight different features and discuss the performance of the proposed approach also with reference to current approaches to this problem present in the literature. Once again, we propose both quantitative and qualitative analyses.

4.2.1. Quantitative Analysis

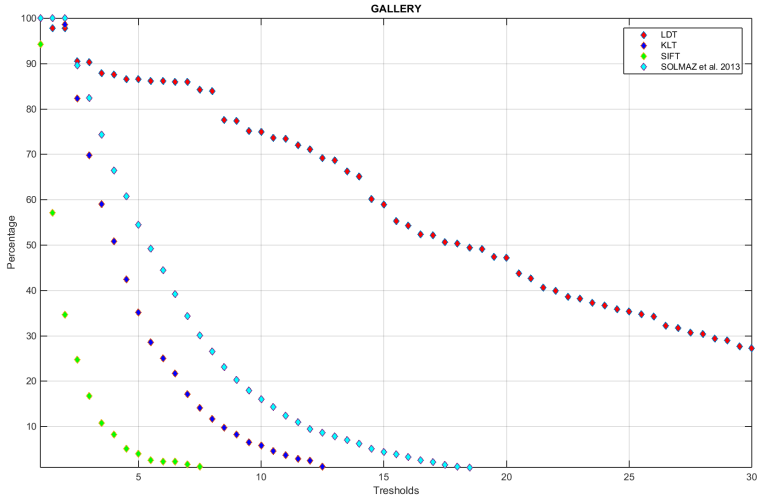
The input to our framework is represented as a sequence of frames and we divide each video sequence into different temporal segments. The length of each analyzed video is 350 frames and we set the length K of each segment equal to 50 frames.

We extract global flows by using the trajectories extracted by using our approach and other baseline methods and finally applying our clustering algorithm 2. In order to quantify the accuracy of each method after employing our clustering algorithm, we compare the achieved results with ground truth global flows. We obtained ground truth data for each analyzed video by manual identification of global flows: the visual plot of manually detected global flows for each analyzed video is shown in Figure 6.

Since to the best of our knowledge there is no agreed upon mechanism for evaluating the accuracy in the detection of sources and sinks, and in the characterization of main flows in a scene, we introduced and computed two metrics

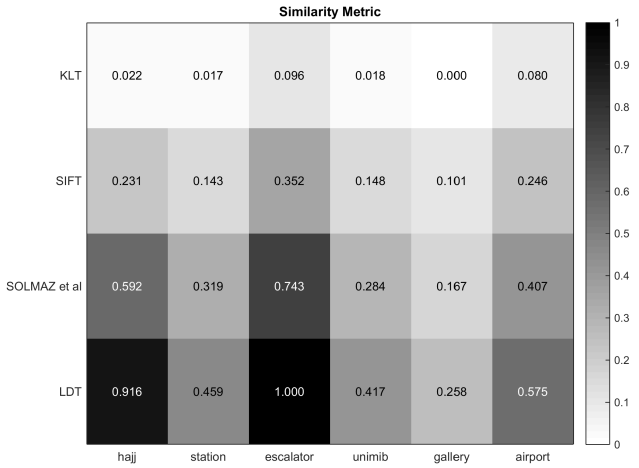


(a) Percentage of remaining tracks with growing thresholds in the station video.

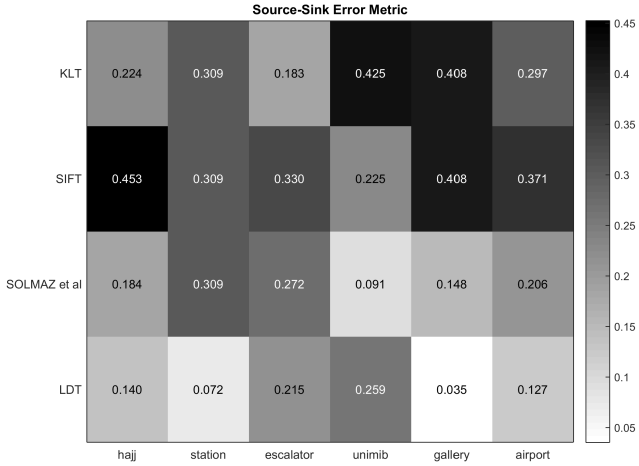


(b) Percentage of remaining tracks with growing thresholds in the gallery video.

Figure 13: Quantitative analyses comparing the survival of trajectories with varying length threshold in station and gallery videos for the proposed approach and state-of-the-art tracking approaches.



(a) Similarity Metric.



(b) Source/Sink Locations Error Metric.

Figure 14: Similarity and source/sink error metrics comparison between proposed approach and baseline methods.

and in particular: (1) flow *similarity* metric and (2) source/sink error metric.

We define and compute flow similarity metric by comparing global flows *Gtrack* detected by each tracks generation method followed by our clustering

algorithm with ground truth Gth. The similarity is measured exploiting LCS and by using the following equation:

$$Sim = \frac{\left(\sum_{i=1}^{N} \arg\max_{j\in[1,M]} \frac{LCS(Gtrack_j, Gth_i)}{Length(Gth_i)}\right)}{N} \tag{4}$$

In particular, N represents total number of ground truth tracks while M represents total number of global tracks detected by method for the analyzed video. The equation considers all the actual N global tracks inn the ground truth data and selects the extracted track that is most similar to the ground truth.

We observe that, in this experimentation, $N \leq M$ uniformly for all approaches; this is likely caused by the fact that our clustering approach works very well with long and dense tracklets but cannot merge into a single cluster tracklets that are too short and not similar according to LCS. We also observe this kind of situation in clustering tracks achieved with baseline methods, since these methods generally produce small and implausible tracks in contrast to DLT that is generally able to capture each dominant motion and to produce almost the same number of global tracks present in the ground truth as shown in Figure 15.

We computed mean similarity Sim for each analyzed video adopting the different track generation techniques and results are shown in Figure 14(a), which supports both a quantitative and qualitative evaluation: darker blocks, in fact, show that global tracks identified by our proposed method is closer to the ground truth than the lighter blocks, associated to the baseline techniques.

The second metric simply measures how far the source/sink locations of extracted global tracks from the source/sink locations of ground truth data. The simplest way to compute this metric would be to calculate euclidean distance between the source/sink locations of global tracks and source/sink location of ground truth tracks for the analysed video. However, this is implausible for mostly two reasons: first of all, distance expressed in pixels is dependent on the type of scene and not necessarily proportional to actual errors in the real world,

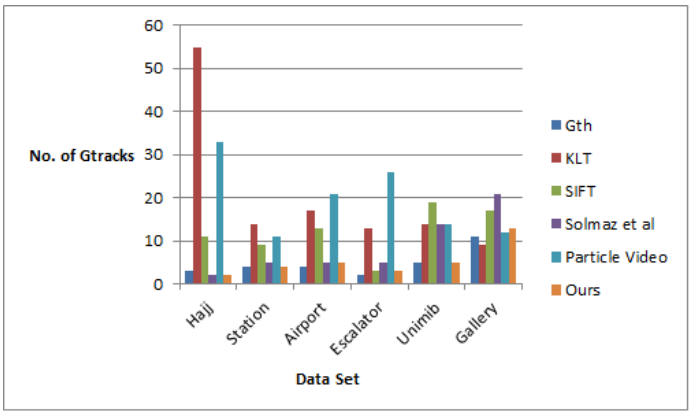


Figure 15: Total number of global tracks found for each method by our clustering algorithm.

due to perspective; second, it is very hard to provide a way to normalize in a sensible way this kind of metric.

Therefore, in order to alleviate this problem, we build an *Association Matrix* that captures the joint probability distribution of source and sink locations of all the trajectories in the analyzed video. Actually, we build two matrixes, one constructed employing the ground truth data and another employing the global trajectories extracted automatically that present the best match to ground truth ones, as for the similarity metric.

In order to build this matrix, we assume two discrete jointly distributed random variables $\mathbf{X}$, representing "source" locations of the trajectories and $\mathbf{Y}$ representing "sink" locations. An *Association Matrix* for n trajectories is shown below.

$$P(X,Y) = \begin{Bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1n} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & p_{n3} & \cdots & p_{nn} \end{Bmatrix}$$

Each row/column of an *Association Matrix* shows the probability distribution of the source and sink points of single trajectory P_k over all other n trajectories in the analyzed video. Let P_k is the distribution of sources and sinks of trajectory k with all other n trajectories and represented as $\{p_{k1}, p_{k2}, p_{k3}, \ldots, p_{kn}\}$,

where $p(k, j)$, is the joint probability of start and stop locations for trajectories k and any trajectory j. We use a Gaussian likelihood model [44] to compute the probability of a trajectory k to start (or stop) from the initial (or final) location of a (potentially different) trajectory j in the scene as equation 5:

$$p_x(k, j) = \mathrm{e}^{-\| \frac{x_k - x_j}{\sigma} \|} \tag{5}$$

Where x_k and x_j are the source (or sink) locations respectively of trajectory k and j. Assuming independence among the trajectories, we multiply the above values for start and sink points to calculate joint probability $p(k, j)$ for trajectories k and j. In the same way, we compute joint probabilities of all other trajectories and after normalization, we obtain an *Association Matrix*.

Following this procedure, we computed *Association Matrixes* for ground truth tracks (AM_{Gth}) and the selected global tracks produced by the compared methods (AM_{Gtrack}), in all the analyzed videos. Finally, we computed the difference between the *Association Matrices* by using *Kullback-Leibler (KL) divergence*, also known as relative entropy, denoted by $D_{KL}(AM_{Gth}||AM_{Gtrack})$, computed by using equation 6. The value $D_{KL}(AM_{Gth}||AM_{Gtrack})$ is associated to the loss of information caused by using *Gtrack* instead of ground truth data *Gtrack*, and it should be considered, therefore, an indicator of how distant the results are from the ground truth.

$$D_{KL}(AM_{Gth}||AM_{Gtrack}) = \sum_i AM_{Gth}(i) \ln \frac{AM_{Gth}(i)}{AM_{Gtrack}(i)} \tag{6}$$

Figure 14(b) reports the values of the above metric for evaluating the distance between the source/sink locations achieved with the proposed method and other baseline approaches from ground truth. This metric is associated to an error, so the low values indicate that source/sink of the global tracks lie close to the ground truth. Results are in line with those related to the flow similarity metric.

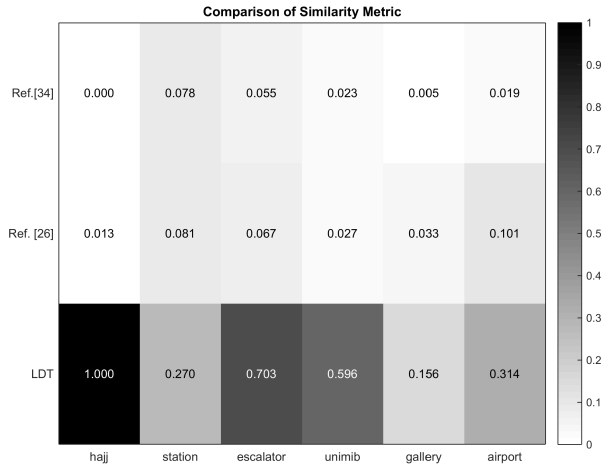
Finally, we also compare our method with most relevant state of the art techniques, i.e, [35] and [27], in a quantitative way. Both these methods use

KLT method for extracting trajectories from the crowded scene followed by clustering algorithm. Since KLT is a sparse method the extracted trajectories are unreliable and short enough to cover just essential parts of the motion in the scene. Another limitation with [35] is that during clustering phase, instead of updating the cluster center, the long trajectory among the set of clustered trajectories is selected as new cluster center. In this way, several trajectories representing the single true flow appeared at the end of clustering. The clustering algorithm is improved in [27] by updating the cluster center, however, the trajectories produced at the end of clustering are still short and they appear as different parts of a single unique actual flow. As shown in 16(a), methods [35] and [27], produce very low similarity values. The reason is that the trajectories produced by these methods are short and hence equation 4 gives very low values, and [27] does not provide significant improvement over [35].

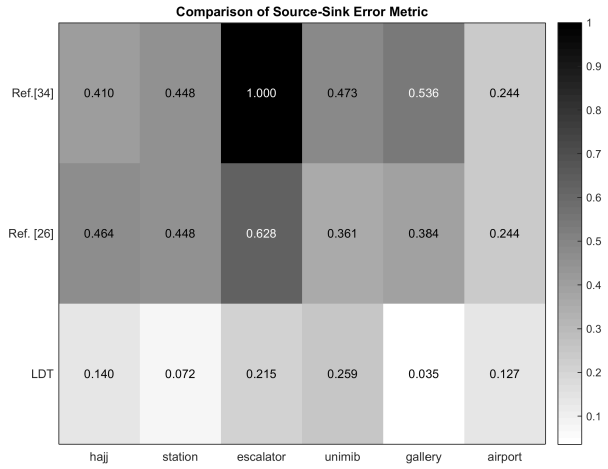
4.2.2. Qualitative Analysis

Local tracks can be considered as a by-product of the overall process, but they can represent useful indications of changes in the situation between different time slices associated to the different segments. For instance, Figure 17 shows different local tracks associated to different segments of the station video as well as the overall global detected tracks: the overall flows are detected correctly (qualitatively comparing Figure 18(c) with Figure 6(c)), moreover during the analysis, some flows detected in a given segment (i.e. Figure 17) are not detected in a following one. This kind of event, beyond the specific situation, could be a signal that could be interpreted by a higher-level module, performing semantic analysis of the results, indicating that an area is changing from free flow to a congested state.

Other situations, similarly characterized by medium-low density situations and relatively stable flows, yield similar results: Figure 18(a) shows that in the airport video the main flows are correctly detected in a multi-floor scenario; some of them are actually correlated, as one merges into another: pedestrians climbing two staircases actually merge into a single flow in a T-junction, but they are



(a) Comparison of Similarity Metric.



(b) Comparison of Source/Sink Locations Error Metric.

Figure 16: Comparison with state of the art source and sink identification methods.

detected as two flows. In an analogous way, the university video is also correctly analyzed, in terms of detection of main flows, as shown in Figure 18(e), but in this case one of the detected flows is actually generated as a split from another larger one. These considerations also call for a subsequent phase of semantic

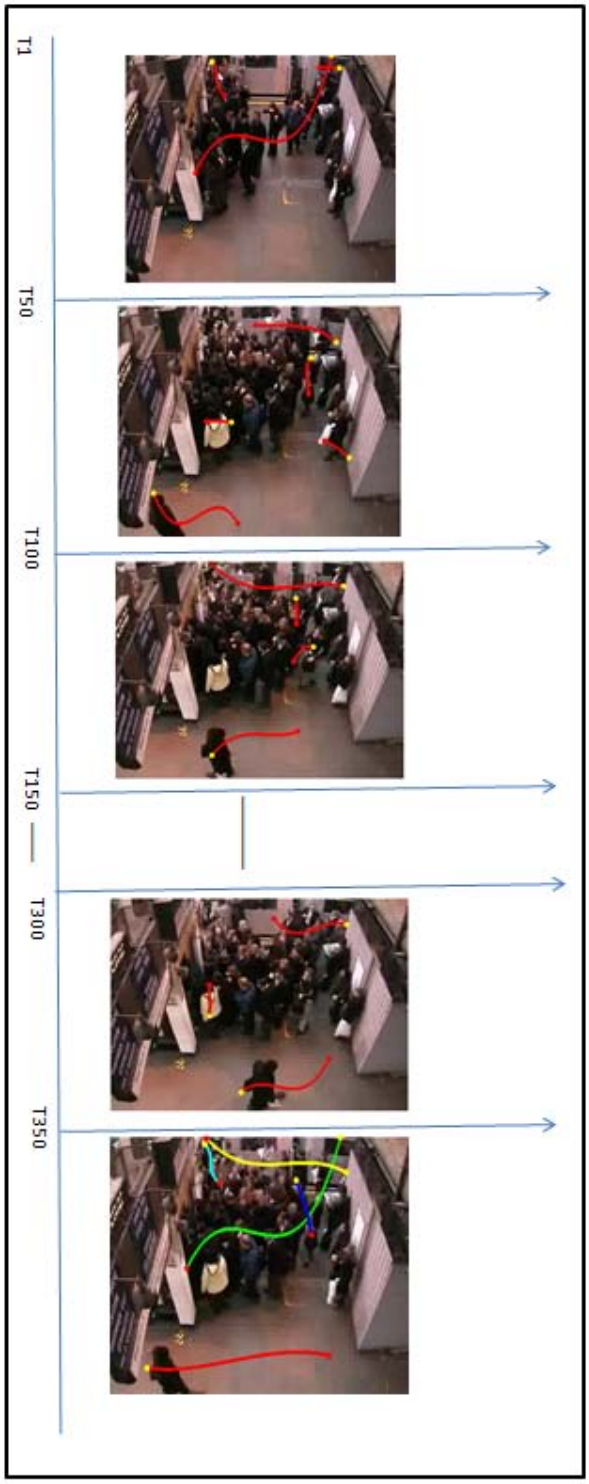


Figure 17: Local tracks resulting from analysis of segments and final global tracks in the station scenario.

analysis after the algorithm, in addition to a quantitative characterization of the flows that would be necessary to actually define an O/D matrix.

The university video analysis also shows the fact that the proposed approach, if properly configured (i.e. with thresholds' values adequate to the specific scenario that is being analyzed), is robust to occlusions due to infrastructural elements that interrupt the visibility of a given flow of pedestrians: poles and tree branches, in fact, do not avoid the completion of tracklets into proper global trajectories. The escalator video analysis, though, shows that this robustness has limits: overall flows are in fact detected but the large obstacle (i.e. a column) combined with the value of the thresholds cause the upper flow to be split into two motion patterns. Adjusting the thresholds, in this case, could solve the problem but there could be downsides, such as the acceptance of incorrect trajectories representing implausible completions of short tracklets.

Finally, the gallery video 18(f) represents a rather extreme scenario that is being mostly reported to show how the proposed approach is robust to occlusions, difficult lighting conditions, high pedestrian density and lack of apparent dominant motion patterns. The scenario, in fact, should be analyzed for a longer time-frame for more interesting and substantial results, that could lead to an improved understanding of the attractiveness of shops and other potential interest points in the area.

In conclusion we can stress the fact that the proposed approach can uniformly provide very interesting results, from the perspective of characterizing dominant pedestrian flows, in all the considered crowding conditions. In the station and escalator videos the approach described in [27] accurately detects the flows but the detected tracks are not long enough to capture the whole motion information, leading to an incomplete characterization of the overall flows.

Similar considerations can be done considering the approach described in [28], in particular for the airport footage: this approach identifies small tracklets but complete information about the motion is missing while our results completely describe flows with their respective sources and sinks. In the Hajj video, moreover, the approach introduced in [28] detects redundant flows while our method

(a) Airport video results.

(b) Hajj video results.

(c) Station video results.

(d) Escalator video results.

(e) UniMiB video results.

(f) Gallery video results.

Figure 18: Crowd flow characterization results of the proposed approach in the different scenarios.

correctly summarizes horizontal flow, although both approaches miss the vertical flow that the human annotator detected, as shown in Figure 6(b).

4.3. Parameters of Algorithms

Here we discuss parameters setting for proposed algorithms 1 and 2. Since both algorithms use different parameters therefore we describe parameter setting for each algorithm separately. Parameters setting for algorithm 1 is described in table 4.3. *step* is the first parameter, though not the actual parameter of algorithm 1, specifies the resolution of grid of particles to be overlaid on the scene. We fix the value of step to 10 for all the analyzed videos. The resolution of grid of particles for a given image of size 300x400 with step 10 is 30x40. It implies that we drop the particle at every 10^{th} pixel location while scanning from left to right (row wise) or top to bottom (column wise) of an image. We can also increase the resolution of the grid by lowering the value of *step*, but this will make the algorithm computational expensive ending up with the similar results. ϵ controls the euclidean distance between the sink point of query tracklet and source point of neighbor tracklet. We set this value to 7.5 for all analyzed videos. m_t controls the matching ratio. We fix this value to 0.4 for all the analyzed videos, that implies that our algorithm accepts candidate tracklet which is at least 40% similar to that of query tracklet.

The description of parameter settings for algorithm 2 is shown in Table 3. Ω controls the spatial matching of any two input trajectories. Tuning of Ω is required in order to obtain semantically useful results for a given sequence, since it depends on the video frame resolution, crowd density, crowd type, i.e structured or unstructured. We set a low threshold value for gallery sequence, since it involve complex movement of people. We use higher value of Ω for structured crowds. This parameter is determined experimentally. Before running the algorithm 2 on a long video sequence, an analyst can tune this parameter to an appropriate value by observing the video for a short time. Parameter φ is the same as m_t and we set it to 0.5 for all video sequences.

Some considerations must be done, finally, on the fact that the passage between local results, related to a single video segment, and final overall global motion flow description does not employ temporal information associated to the local flows (which would probably be necessary for a proper tracking algorithm).

Table 2: Parameter Setting for Algorithm 1

Variable	Description	Value
step	Control the resolution of grid	10
ϵ	allowed distance between tracklets in Algorithm 2	$step(\frac{3}{4})$
m_t	Matching ratio	0.4

Table 3: Parameter Setting for Algorithm 2

	Hajj	Station	Unimib	Airport	Escalator	Gallery
Resolution	384x576	360x480	360x480	360x480	480x480	360x480
Ω	120	90	90	90	120	70
φ	0.5	0.5	0.5	0.5	0.5	0.5

The example shown in Figure 17 shows that the clustering technique devised and adopted for this final passage actually allows considering all flows that, even just temporarily (i.e. not in all segments), represented a relevant and noticeable flow of pedestrians. Moreover, the proposed approach actually exploits the fact that pedestrians tend to follow similar paths in the environment, sometimes imitating the movement of other pedestrians: the trajectory completion function described in Algorithm 2, in fact, supports the detection of an overall pedestrian flow even in a single video segment, even though a single pedestrian would not be able to cover it, as long as other pedestrians are moving along a similar path.

5. Conclusions and Future Works

The paper presented a framework for the automated analysis of videos in naturalistic conditions and the identification of points of entrance (sources) and exit (sinks) of the most significant pedestrian flows. The approach adopts optical flow for the identification of pedestrian movements, and it considers the analyzed video as a set of sequences. The latter are analyzed separately, producing tracklets that are then clustered to produce global trajectories, defining both sources and sinks, but also characterizing the movement of pedestrians in the scene. The algorithms work according to geometric considerations essentially

considering the plausibility of extending tracklets associated to optical flow by connecting them when they represent a smooth continuation one of another, and then clustering those sharing a significant subsequence.

The approach has been presented in details, also setting it in the current state of the art. Results of its application to the analysis of videos made available by other researchers and by our research group have been discussed mainly with reference to two aspects: (i) the capability of producing long and dense tracks associated to pedestrian movements, also with reference to the most relevant approaches present in the computer vision literature, and (ii) the capability of summarizing pedestrians' movements, identifying at the same time sources and sinks. For both aspects, the considered videos cover a wide range of crowding situations, from medium-low to relatively high crowding conditions, in cases of structured and unstructured crowds. For both aspects quantitative and qualitative analyses of results have been produced. We plan to make available both the original videos, ground truth and achieved results through a publicly available data set by the time this paper is published.

Future works are mainly headed towards four directions:

1. extensions of the approach to produce information that can be more directly used by modelers for the configuration of simulation scenario, that is, origin destination matrices and traffic assignments: this point will require a quantitative characterization of the sources, sinks and main flows, and it will also imply a different form of experimentation analyzing longer videos, but the quantitative evaluation of this development is rather straightforward;
2. extension of the approach to consider multi-camera scenarios: the present approach is very promising but so far we did not analyze large scenarios in which the analyzed area can only be covered by more cameras and scenes; this will require a higher level of correlation among results of the application of the same approach to different videos, a higher level that is also related to the next point;

3. extension of the approach to perform a semantic analysis of the results, for the identification and treatment of situations essentially characterizable as (i) confluence of different flows in a single one, (ii) separation of initially joint flows; these situations, as well as the previous one, will likely require the adoption of some form of knowledge representation and reasoning on graph-like structures associated either to static spatial representations or to the results of the application of the proposed approach.

References

[1] R. Challenger, C. W. Clegg, M. A. Robinson, Understanding crowd behaviours: Supporting evidence, Tech. rep., University of Leeds (2009).

[2] J. C. S. J. Junior, S. R. Musse, C. R. Jung, Crowd analysis using computer vision techniques, IEEE Signal Processing Magazine 27 (5) (2010) 66–77.

[3] B. Solmaz, B. E. Moore, M. Shah, Identifying behaviors in crowd scenes using stability analysis for dynamical systems, Pattern Analysis and Machine Intelligence, IEEE Transactions on 34 (10) (2012) 2064–2070.

[4] T. Zhao, R. Nevatia, Tracking multiple humans in crowded environment, in: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, Vol. 2, IEEE, 2004, pp. II–406.

[5] Z. Khan, T. Balch, F. Dellaert, An mcmc-based particle filter for tracking multiple interacting targets, in: Computer Vision-ECCV 2004, Springer, 2004, pp. 279–290.

[6] C. Hue, J.-P. Le Cadre, P. Perez, Posterior cramer-rao bounds for multi-target tracking, Aerospace and Electronic Systems, IEEE Transactions on 42 (1) (2006) 37–49.

[7] G. J. Brostow, R. Cipolla, Unsupervised bayesian detection of independent motion in crowds, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, Vol. 1, IEEE, 2006, pp. 594–601.

[8] D. Sugimura, K. M. Kitani, T. Okabe, Y. Sato, A. Sugimoto, Using individuality to track individuals: clustering individual trajectories in crowds using local appearance and frequency trait, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 1467–1474.

[9] A. R. Zamir, A. Dehghan, M. Shah, Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs, in: Computer Vision–ECCV 2012, Springer, 2012, pp. 343–356.

[10] M. Boltes, A. Seyfried, Collecting pedestrian trajectories, Neurocomputing 100 (0) (2013) 127 – 133, ¡ce:title¿Special issue: Behaviours in video¡/ce:title¿. doi:10.1016/j.neucom.2012.01.036.
URL http://www.sciencedirect.com/science/article/pii/S0925231212003189

[11] S. Baker, I. Matthews, Lucas-kanade 20 years on: A unifying framework, International Journal of Computer Vision 56 (3) (2004) 221–255. doi:10.1023/B:VISI.0000011205.11775.fd.
URL http://dx.doi.org/10.1023/B:VISI.0000011205.11775.fd

[12] C. Stauffer, Estimating tracking sources and sinks, in: Computer Vision and Pattern Recognition Workshop, 2003. CVPRW'03. Conference on, Vol. 4, IEEE, 2003, pp. 35–35.

[13] M. Nedrich, J. W. Davis, Learning scene entries and exits using coherent motion regions, in: Advances in Visual Computing, Springer, 2010, pp. 120–131.

[14] P. Sand, S. Teller, Particle video: Long-range motion estimation using point trajectories, International Journal of Computer Vision 80 (1) (2008) 72–91.

[15] R. Messing, C. Pal, H. Kautz, Activity recognition using the velocity histories of tracked keypoints, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 104–111.

[16] I. Laptev, On space-time interest points, International Journal of Computer Vision 64 (2-3) (2005) 107–123.

[17] P. Matikainen, M. Hebert, R. Sukthankar, Trajectons: Action recognition through the motion analysis of tracked features, in: Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 514–521.

[18] J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, J. Li, Hierarchical spatio-temporal context modeling for action recognition, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 2004–2011.

[19] J. Sun, Y. Mu, S. Yan, L.-F. Cheong, Activity recognition using dense long-duration trajectories, in: Multimedia and Expo (ICME), 2010 IEEE International Conference on, IEEE, 2010, pp. 322–327.

[20] M. Raptis, S. Soatto, Tracklet descriptors for action modeling and video analysis, in: Computer Vision–ECCV 2010, Springer, 2010, pp. 577–590.

[21] B. Zhou, X. Wang, X. Tang, Random field topic model for semantic region analysis in crowded scenes from tracklets, in: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 3441–3448.

[22] B. Zhou, X. Wang, X. Tang, Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 2871–2878.

[23] T. Brox, J. Malik, Object segmentation by long term analysis of point trajectories, in: Computer Vision–ECCV 2010, Springer, 2010, pp. 282–295.

[24] W.-C. Lu, Y.-C. Wang, C.-S. Chen, Learning dense optical-flow trajectory patterns for video object extraction, in: Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on, IEEE, 2010, pp. 315–322.

[25] M. Rodriguez, S. Ali, T. Kanade, Tracking in unstructured crowded scenes, in: IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009, 2009, pp. 1389–1396. doi:10.1109/ICCV.2009.5459301.
URL http://dx.doi.org/10.1109/ICCV.2009.5459301

[26] S. D. Khan, G. Vizzari, S. Bandini, S. Basalamah, Detecting dominant motion flows and people counting in high density crowds.

[27] A. M. Cheriyadat, R. J. Radke, Detecting dominant motions in dense crowds, Selected Topics in Signal Processing, IEEE Journal of 2 (4) (2008) 568–581.

[28] W. Chongjing, Z. Xu, Z. Yi, L. Yuncai, Analyzing motion patterns in crowded scenes via automatic tracklets clustering, Communications, China 10 (4) (2013) 144–154.

[29] T. Brox, A. Bruhn, N. Papenberg, J. Weickert, High accuracy optical flow estimation based on a theory for warping, in: Computer Vision-ECCV 2004, Springer, 2004, pp. 25–36.

[30] P. Berens, M. J. Velasco, The circular statistics toolbox for matlab, MPI Technical Report No 184.

[31] J.-G. Lee, J. Han, K.-Y. Whang, Trajectory clustering: a partition-and-group framework, in: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, ACM, 2007, pp. 593–604.

[32] J. Sander, M. Ester, H.-P. Kriegel, X. Xu, Density-based clustering in spatial databases: The algorithm gdbscan and its applications, Data mining and knowledge discovery 2 (2) (1998) 169–194.

[33] Z. Zhang, K. Huang, T. Tan, Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes, in: Pattern Recognition, 2006. ICPR 2006. 18th International Conference on, Vol. 3, IEEE, 2006, pp. 1135–1138.

[34] M. Vlachos, G. Kollios, D. Gunopulos, Discovering similar multidimensional trajectories, in: Data Engineering, 2002. Proceedings. 18th International Conference on, IEEE, 2002, pp. 673–684.

[35] A. M. Cheriyadat, R. J. Radke, Automatically determining dominant motions in crowded scenes by clustering partial feature trajectories, in: Distributed Smart Cameras, 2007. ICDSC'07. First ACM/IEEE International Conference on, IEEE, 2007, pp. 52–58.

[36] S. Ali, M. Shah, A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis, in: CVPR, 2007.

[37] M. L. Federici, A. Gorrini, L. Manenti, G. Vizzari, Data collection for modeling and simulation: Case study at the university of milan-bicocca, in: G. C. Sirakoulis, S. Bandini (Eds.), ACRI, Vol. 7495 of Lecture Notes in Computer Science, Springer, 2012, pp. 699–708.

[38] S. Bandini, A. Gorrini, G. Vizzari, Towards an integrated approach to crowd analysis and crowd synthesis: A case study and first results, Pattern Recognition Letters 44 (2014) 16–29.

[39] O. Ozturk, T. Yamasaki, K. Aizawa, Detecting dominant motion flows in unstructured/structured crowd scenes, in: Pattern Recognition (ICPR), 2010 20th International Conference on, IEEE, 2010, pp. 3533–3536.

[40] P. Berens, Circstat: a matlab toolbox for circular statistics, J Stat Softw 31 (10) (2009) 1–21.

[41] J. Shi, C. Tomasi, Good features to track, in: Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on, IEEE, 1994, pp. 593–600.

[42] B. D. Lucas, T. Kanade, et al., An iterative image registration technique with an application to stereo vision., in: IJCAI, Vol. 81, 1981, pp. 674–679.

[43] S. Battiato, G. Gallo, G. Puglisi, S. Scellato, Sift features tracking for video stabilization, in: Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on, IEEE, 2007, pp. 825–830.

[44] K. Sankaranarayanan, J. W. Davis, Learning directed intention-driven activities using co-clustering., in: AVSS, 2010, pp. 400–407.

Algorithm 2 MatLab-like pseudo-code for the clustering long trajectories into local dominant flows

Input: Long Trajectories T

Output: Local dominant flows T

while $c \neq 0$ **do**
  $idx \leftarrow sort(T,' descend')$
  while $idx \neq \emptyset$ **do**
    $topidx \leftarrow idx(1)$
    $lidx \leftarrow length(idx)$
    $lowidx \leftarrow idx(lidx)$
    $count \leftarrow 1$
    $idx(topidx) \leftarrow \emptyset$
    $cluster[clust_{idx}] \leftarrow T(topidx))$
    for $i \leftarrow 1 : length(idx)$ **do**
      $R \leftarrow \frac{LCS(cluster[clust_{idx}],T(lowidx))}{min(cluster[clust_{idx}],T(lowidx))}$
      if $R \geq \varphi$ **then**
        $cluster[clust_{idx}] \leftarrow UpdateCenter(cluster[clust_{idx}], T(lowidx)))$
        $idx(lidx) \leftarrow \emptyset$
        $lidx \leftarrow lidx - 1$
        $lowidx \leftarrow idx(lidx)$
      else
        $lowidx \leftarrow idx(lidx - 1)$
        $lidx \leftarrow lidx - 1$
      end if
    end for
    $clust_{idx} \leftarrow clust_{idx} + 1$
  end while
  if $length(T) \neq length(cluster)$ **then**
    $c \leftarrow 1$
  else
    $c \leftarrow 0$
  end if
  $T \leftarrow cluster$
end while