Dynamic Modeling of Streaklines for Motion Pattern Analysis in Video

Anonymous CVPR submission

Paper ID 13

Abstract

Natural videos consist of multiple objects interacting with each other in complex ways. Analysis of such videos is a very challenging problem. The underlying patterns of motion contain valuable information about the activities and hence, it is necessary to devise efficient mechanisms for extraction and analysis of these patterns. Tracking is often infeasible in these scenarios. Recently, flow-based modeling methods based on the concept of streaklines in fluid mechanics have been proposed to deal with videos of crowds. In this paper, we show how the streaklines representation can be combined with dynamical systems modeling approaches in order to analyze the motion patterns in a wide range of natural videos. This combination provides a powerful tool to identify similar segments in a video that exhibit similar motion patterns. We show results on challenging datasets by demonstrating unsupervised clustering of similar motion patterns (which often represent similar activity patterns) and detection of changes in the video.

1. Introduction

The analysis of motion patterns in a video is a problem that is widely being worked on today. In this paper, we consider a video to be a collection of moving pixels. The motion of objects in a scene constitutes patterns in the motion of these pixels. The analysis of a video is the task of recognition of these motion patterns. We demonstrate the usefulness of optical flow in the analysis of motion patterns in a complex scene consisting of multiple moving objects. The authors in [14] demonstrated a set of features called streaklines which are derived from the integration of optical flow over an interval of time. These were used for crowd segmentation and anomaly detection in a crowded scene without the use of a tracker. In this paper, we will demonstrate the power of streaklines in the modeling and representation of motion in a broader range of applications. Figure 1 shows an example of streaklines for a video of cars moving in a parking lot. We can see that an inspection of the streaklines gives us a fair idea of the motion of the car over a period of time. The challenge is to represent these streaklines using a model in a manner which makes it feasible to compare the underlying motion patterns within and across video sequences.

We consider an outdoor surveillance scenario containing multiple objects simultaneously in the scene, the objective being to identify the activities occurring in the scene. It is well understood that it is not always easy to obtain accurate tracks of objects in such a scene. Also, it is difficult to define a training set on the activities occurring in these videos due to a large amount of variation in position, viewpoint and configuration of objects even when the camera is static. We propose that streaklines provide us with a representation that is suitable for distinguishing between different motions and computing a mathematical model for each motion pattern without having to compute tracks. We demonstrate the application of our model in grouping similar motion patterns in a video in an unsupervised manner.

Optical flow is a vector field which represents the incremental pixel-wise motion in a video from one frame to the next. Streaklines on the other hand, provide us with a set of paths traced by pixels during an interval of time. Therefore, pixels with a common motion pattern results in similar streaklines. We show that the streaklines can be modeled using a linear dynamical model. Thereby, motion in a video is represented as a collection of dynamical models. Similar motion patterns would result in similar models,

Figure 1. The figure shows the streaklines computed for a moving car in a video. a) shows a sample frame, b) shows the streaklines (in blue) computed over an interval of time. These streaklines represent the motion of the objects in the scene during the time interval...
which implies that grouping of similar motion patterns can be achieved by comparing their models. In this paper, we will show how streaklines can be used to group similar activities using ARMA modeling of streaklines and an application of our method to change detection in a multi-object video.

1.1. Related work

Optical flow often serves as a good approximation of the true physical motion projected onto the image plane [17]. Researchers have explored the use of optical flow in motion analysis in recent years. Histogram of Oriented Optical Flow (HOOF) [3] modeled single person videos using histograms of optical flow as features in each frame. Optical flow histograms have also been used to analyze the motion of individual players in soccer videos [8]. Most of such approaches utilize the statistics of optical flow for recognition rather than the flow itself. Crowd motion analysis is an area where optical flow has been commonly used in approaches that model the flow itself rather than the statistics. Optical flow has been used in dense crowd motion analysis and segmentation [9]. Helmholtz decomposition has been used to segment different motions in crowd scenes by streakline computation in [14]. Here, we show how flow based methods can be used in the analysis of multi-object scenes with sparse motion.

Linear and non-linear dynamical systems have been shown to be useful in the modeling of dynamic scenes. The authors in [18] modeled motion as a non linear dynamical model of the changes in configurations of point objects. However, they utilized hand-picked location data with no observation noise in their experiments. Auto-Regressive Moving Average (ARMA) models have been used to model movements as a linear dynamical system. The authors in [19] utilize ARMA models for single person gait recognition using shape deformations of the silhouette. ARMA models have been used for track based activity and gait recognition in [2]. Dynamic textures have been represented using ARMA models in [7].

There has been a significant amount of work in the analysis of motion in videos in the past. Most of these methods use local spatio-temporal regions as features. Some examples of such features are SIFT (scale invariant feature transform) [12] and STIP (space time interest points) [10]. Our features in comparison can extend over several frames of the video, and represent temporal evolution of the scene over a period of time. Analysis of crowded scenes using foreground blobs for the purpose of estimating the number of objects in the scene has been performed in [5]. This method however does not analyze motion patterns. Patterns in motion of pedestrians was modeled by utilizing relative distances within a coupled HMM using synthetic agent priors [15]. Group motion segmentation has been performed in sports video by learning a Spatio-Temporal Driving Force Model for the motion trajectories [11]. Most of such methods require tracking of individual objects which is not always feasible. Multi-person activity analysis at close range was performed using spatio-temporal features in [16]. This work dealt with close range videos with a fixed viewpoint. Many of these approaches also required a training phase with the set of activities known apriori. Our proposed flow based approach can handle such situations as well as a much broader class of applications, like segmentation, unsupervised clustering and analysis of long term patterns.

1.2. Contribution of this work

We propose a framework to utilize streaklines in the analysis of a multi-object scene. The uniqueness of these features is that they can provide the temporal evolution of scene over an interval of time unlike local features which provide information about a small time interval. Although streaklines have been used before in the analysis of scenes in [14], the authors there had dealt with crowded environment where the motivation was to segment out regions with common motion patterns. The authors had also demonstrated abnormal activity recognition using the change in the potential functions obtained from the Helmholtz decomposition. These potential functions, although effective in distinguishing between patterns occurring simultaneously in a video, do not provide a representation for these motion patterns. We, on the other hand, define a model for each motion pattern in the scene. We not only perform a segmentation but also model these segments using a dynamical system. Our representation is independent of the scene to which the pattern belongs. We will show that it is possible for us to match patterns not just in the same scene but also across viewpoints and across videos with the help of dynamical models. This combination of streaklines and dynamical models is the main contribution of this work.

We perform experiments on a natural dataset containing arbitrary number of objects moving in and out of the scene. We do not assume to know the number of activities occurring in the scene and do not make assumptions on the camera viewpoint or scale. Our method tries to find different patterns and compute similarities between them in an unsupervised manner. We also show that by comparing adjacent temporal segments we can detect changes in the motion patterns and segment the video.

2. Overview

In this paper, we address the problem of analysis of multi-object behaviors in real world situations. Complex behaviors involving multiple interacting objects are characterized by a dynamically evolving spatio-temporal pattern. One approach of analyzing such videos would be to com-
pute and analyze the tracks of all objects in a scene. However, it is not always possible to extract reliable tracks, especially in low resolution or noisy sequences. An alternative would be to adopt a vector field approach. We show that vector field analysis, coupled with a dynamical model, can be used to characterize motion patterns in a video. Optical flow is a motion field which describes the pixel wise apparent motion in a frame from one time instant to another. This motion field can be used as a reliable representation of the motion information in a video.

Since we are looking at constantly evolving scenes, it is not easy to obtain the motion patterns by a direct inspection of the optical flow. Also, any inaccuracies in the computation of optical flow can result in spurious patterns which should not be confused with motion of objects. Therefore, a better approach would be to integrate the flow over an interval of time. Some examples of such data are streamlines, pathlines and streaklines. It has been shown in [14] that streaklines are efficient in representing motion in a crowded scene. In this paper, we show that streaklines can also be used to perform an in-depth analysis of motion patterns in a non-crowded scenario.

After representing a scene using streaklines, the next step in motion pattern analysis would be to model these streaklines. Since the motion between consecutive frames is typically small, we assume that there is a linear relationship between the motion of an object in one time instant to the next. Therefore, we segment motion patterns and model these segments as linear dynamical systems. We show that we can use streaklines to represent and compare motion patterns in a way which is invariant to scale, viewpoint and speed of motion.

In the next section, we will describe the extraction and modeling of motion patterns in detail. We will then look into some applications of our algorithm to unsupervised clustering of motion patterns and change detection. We will analyze the performance of our algorithm on a challenging multi-object dataset.

3. Motion pattern analysis using streaklines

A Streakline can be defined as locations of all particles at a given time that passed through a particular point. It can be computed by initializing a set of particles at every time instant in the field and propagating them forward with time based on the flow field at every instant. This results in a set of paths, each belonging to one point of initialization. The authors in [14] show that the streakline representation has advantages over other representations like streamlines and pathlines in being able to capture changes in the field as well as in smoothness of the resulting representation.

We will now look at the steps involved in the extraction and modeling of motion patterns.

![Figure 2](image-url) Figure 2. The figure shows a sample frame of a video showing two cars moving in different directions in a). Clustering of streaklines is performed to obtain the regions associated with the two motion patterns. These have been marked in different shades in b).

3.1. Extraction of Motion Patterns

We begin with the computation of streaklines in a video. Consider a video sequence of time duration $T$ seconds. We divide this video into $n$ fixed time intervals, say $(T_1, T_2, \ldots, T_n)$, each of duration $\tau$ seconds. Optical flow provides us the flow field at each time instant for the computation of streaklines. We compute streaklines for each interval separately.

We initialize a particle at every pixel of every frame over the interval of time. The particles are propagated according to the optical flow at each instant. Let $(x_i, y_i)$ be the position of a particle $p$ at time $t$. Suppose the particle is initialized at $(x_{Ti}, y_{Ti})$ at time instant $T_i$, $i \in \{1, 2, \ldots, n\}$. The position of particle $p$ at any time instant $t_j > T_i$ is computed as described in [14].

$$x_{t_j} = x_{t_{j-1}} + u(x_{t_{j-1}}, y_{t_{j-1}}, t_j)$$

$$y_{t_j} = y_{t_{j-1}} + v(x_{t_{j-1}}, y_{t_{j-1}}, t_j)$$

where $u(x, y, t)$ and $v(x, y, t)$ are the X and Y components of the optical flow at position $(x, y)$ at time instant $t$. At every time instant, the existing particles are propagated forward using this recursive relationship and a new set of particles are initialized at every pixel. A streakline is the collection of all particles which were initialized at a particular pixel. If there are $N$ pixels in a frame of the video, we have $N$ streaklines $s_1, s_N$ computed over each interval. For ease of notation, we will collectively denote the streaklines computed over the interval $T_i$ by $S_T_i$. Therefore, a video is now represented by the streaklines $S_1, \ldots, S_n$. Figure 1 shows the streaklines of a video computed over a single interval of time. After computing the streaklines, we separate the different motion patterns in the video by clustering similar streaklines. We define a distance measure to compute the similarity between any two streaklines $s_a$ and $s_b$ in a manner similar to [14].

$$D(s_a, s_b) = \|s_{aN} - s_{bN}\|$$

where $s_{aN}$ and $s_{bN}$ are the normalized streaklines. The normalization is performed by centering the streakline at zero
and scaling to norm one. We form a similarity map of every streakline in its neighborhood. After the similarity map is computed, we compute peaks in the similarity map which are the boundaries between regions of similar motion. A watershed segmentation on these boundaries gives us regions which correspond to motion patterns. The streaklines belonging to each region correspond to a cluster which represents the motion pattern. If there are \( K \) clusters in time interval \( T_t, S_{T_t} \) is now divided into clusters \( C_1, C_2...C_K \). Figure 2 shows an example of two clusters identified for a video with two cars moving differently. The marked areas give us an idea of the region over which the objects might have traveled in the time interval.

### 3.2. Modeling of motion patterns

We have shown that streaklines represent motion of a pixel over time. This gives rise to the idea that a streakline can be thought of as a noisy sample of the underlying process which defines the motion of the object at that pixel over the interval of time. Let \( m(t) \) be the underlying motion of pixel \( p_k \) at time instant \( t \). The value of the streakline \( s \) initiated at that pixel at time instant \( t \) can be defined as a function of the underlying motion \( m(t) \), i.e. \( s(t) = \phi(m(t)) \) with an additive noise, say \( w \). The motion of the pixel at time \( t \) can be defined as a function of the motion at the previous time instant \( t-1 \), i.e. \( m(t) = \psi(m(t-1)) \). Let \( v \) represent the additive noise in the motion model. This can be expressed as a linear dynamical system with additive stationary noise giving rise to an ARMA model as follows:

\[
\begin{align*}
m(t) &= Am(t-1) + Bv(t) \\
s(t) &= Cm(t-1) + w(t)
\end{align*}
\]  

(3)

Here, \( A, B \) and \( C \) are parameters of the model to be computed. \( A \) defines the relationship between motion in one time instant to the next, \( B \) is the function of additive noise and \( C \) defines the relationship between the streaklines and the underlying motion. Since similar motion patterns give rise to similar streaklines, we can expect that the motion models of the two patterns are also similar. We will solve this system of equations using the closed form solution described in [7], with \( m(t) \in \mathbb{R}^{n_1} \), where \( n_1 \) is the order of the model, \( s(t) \in \mathbb{R}^{n_2} \), where \( n_2 \) is the size of the cluster, \( v \) and \( w \) are zero mean Gaussian noise given by \( v \sim N(0,Q) \), \( w \sim N(0,R) \). The parameters \( A, C, P \) and \( Q \) can be computed as explained below.

Let \( P^{\tau} = [s(1),...,s(\tau)] \in \mathbb{R}^{n_2 \times \tau} \) where \( \tau > n_1 \), then for \( t = 1...\tau \) we can write

\[
P^{\tau} = CM^{\tau} + W^{\tau}; C \in \mathbb{R}^{n_2 \times n_1}
\]  

(4)

where \( M \) is defined as the time series of motion of the pixel given by \( M^{\tau} = [m(1),...,m(\tau)] \in \mathbb{R}^{n_1 \times \tau} \) and \( W \) is the time series of the noise vector given by \( M^{\tau} = [m(1),...,m(\tau)] \in \mathbb{R}^{n_2 \times \tau} \). If the singular value decomposition of \( P^{\tau} \) is \( P^{\tau} = USV^{T} \), where \( \Sigma \) is a diagonal matrix, \( U \in \mathbb{R}^{n_2 \times n_1}, U^{T}U = I, V \in \mathbb{R}^{\tau \times n_1}, V^{T}V = I \), then the solution to the system is given by

\[
\hat{C}(\tau) = U \\
\hat{M}(\tau) = \Sigma V^{T}
\]  

(5)

\[
\hat{A}(\tau) = \Sigma V^{T}D_{1}V(V^{T}D_{2}V)^{-1}\Sigma^{-1}
\]  

(6)

(7)

where \( D_{1} = \begin{pmatrix} 0 \\ I_{\tau-1} \end{pmatrix} \) and \( D_{2} = \begin{pmatrix} I_{\tau-1} & 0 \\ 0 & 0 \end{pmatrix} \).

Here, \( P \) is the collection of \( n_2 \) streaklines belonging to a common cluster representing a single motion pattern and \( \tau \) stands for the number of frames over which the streaklines were computed, or the data points over which motion is modeled. We derive the ARMA model for every cluster of streaklines, and the video is now represented by a set of motion models.

#### 3.2.1 Similarity between motion patterns

The similarity between two motion patterns is quantized by the distance between their corresponding dynamical models. As seen in Figure 3, similar motion patterns result in similar streaklines, hence their models are expected to be similar. We will follow the method described in [4] to compute the distance between two ARMA models. The above ARMA model is converted into a forward innovation model...
Figure 4. The figure shows the similarity matrix for five actions. The actions in order are left turn, right turn, go straight, u-turn and curbside parking.

given by
\[ \hat{\mathbf{m}}_t = A\hat{\mathbf{m}}_{t-1} + K_G e_{t-1} \]  (8)
\[ \hat{s}_t = C\hat{\mathbf{m}}_{t-1} + e_{t-1} \]  (9)

where \( K_G \in \mathbb{R}^n_+ \) is the Kalman gain. The distance between two ARMA models is computed in terms of the angles between their subspaces. The subspace angles are computed using an eigenvalue problem involving the parameters of forward and inverse innovation models. To compute the distance between two models based on the subspace angles, we use the metric defined in [13]. For two models \( M_1 \) and \( M_2 \), with their first \( n_1 \) subspace angles given by \( \theta_1, \theta_2, \ldots, \theta_{n_1} \), the distance between them is given as

\[ D(M_1, M_2)^2 = -\log \prod_{i=1}^{n_1} \cos^2 \theta_i. \]  (10)

To make the system view and scale invariant, some preprocessing of the data is needed. We first normalize the cluster of streaklines using the method described in 3.1. Next, we define an affine transformation between the data points in one cluster to those of another. This will align the sequences which can then be modeled and compared.

4. Experiments

In this section, we will describe the various experiments we performed using the algorithm described in the previous section. We applied the algorithm to some common motion pattern analysis problems in computer vision. The goals of our experiments were:

1. To group similar motion patterns in a video in an unsupervised manner;
2. To perform change detection in a single video sequence;
3. To match motion patterns across videos to test robustness with change in scale and viewpoint

We applied the algorithm to videos taken from two publicly available datasets: The VIRAT video dataset [1] and the UCR videoweb dataset [6]. The videos taken from the VIRAT data consists of a 15-minute data with clips of parking areas. There are vehicles coming in, turning left, right, exiting the parking lot and parking. The UCR data is a 5-minute multi-camera parking lot data with a road intersection in view. There are vehicles moving along the roads, taking turns at the intersection, into the parking lot or within the parking lot. The video has been captured from eight different cameras, giving different viewpoints of the scene. There is a difference in the scale of the two datasets and the orientation of different videos also differ. Although there are people moving in the scene in both cases, in this work we have focused on the modeling of vehicle motion. There are no assumptions made on the number of objects or motion of objects at any time in the scene. All the video was captured in a natural outdoor environment.

4.1. Unsupervised grouping of similar motion patterns

A complex video sequence such as the one used in our experiment contains a large amount of variation in position, orientation and nature of motions in the scene. This makes it impractical to analyze motion in a supervised framework with the help of training data. Therefore, for any system analyzing such data, the capability of being able to group similar motions would be an important step. The objective of this experiment is to be able to group similar motion patterns by an examination of their streaklines in a completely unsupervised manner.

The main steps in the experiment involve separating out different motion patterns in a given video and comparing them with each other. Given a video sequence, we divide it into segments of a certain time interval \( \tau \), in our case taken to be 3 seconds. We compute streaklines for each interval of time and perform a clustering of streaklines. The task of grouping these clusters involves a pairwise comparison of the clusters over the entire video. The distance between every pair is computed in three steps: alignment of the pair of motion patterns by normalization and affine transformation, fitting an ARMA model to the aligned clusters and computation of distance between the two clusters using subspace angles. The motion patterns can now be grouped based on the similarity matrix and a distance threshold. We have considered different time intervals of a video containing the motion of cars in and around a parking lot. The activities analyzed are left turn, right turn, go straight, u-turn and...
The figure shows some examples of similar activities grouped using our algorithm. Each row depicts an activity as given by the label on the left. The shaded regions illustrate the cluster of streaklines which were modeled. Four examples of each case are shown.

The ROC curve for unsupervised grouping of motion patterns using our algorithm is shown in Figure 6. The sensitivity of the curve denotes the ratio of \( \frac{\text{True Positive}}{\text{False Negative}} \) and specificity is given by the ratio of \( \frac{\text{False Positive}}{\text{True Negative}} \) for varying thresholds.

### 4.2. Application to change detection in video sequences

Consider a video of a natural scene consisting of an arbitrary number of objects. Typically, the events which can be considered as useful in a surveillance scenario are sparse. The video would consist of periods of no motion, some instants of uniform motion and periods with changing motion. Therefore, the first step in the analysis of such scenes would be to identify segments which would require further processing. In this work, we define this to be the task of change detection. A change detection system would identify segments where the movement of one or more objects change, in other words, a change in the motion pattern is observed. Some examples are a vehicle changing direction, or entering or exiting a scene. Here, we will demonstrate the application of our algorithm to change detection in a video.

The procedure for change detection is as follows: We divide the video into intervals of 3 seconds just as in the previous case. At each time interval, we cluster streaklines and model the motion pattern they represent. Between every two successive intervals, change detection is performed by computing distances between the motion patterns of one time interval with the next. An overall distance is computed by summing the distances between all matched patterns. A
change is said to have occurred at interval $T_i$ if the number of segments in $T_i$ differs from the number of segments in $T_{i-1}$ or the distance between motion patterns is greater than a predefined threshold which is set empirically. Experiments were conducted on the motion of cars in the dataset. Some sample frames of successive time intervals along with the distances computed between their models is shown in Figure 7. We notice that when the car is moving along a straight path, there is little change in the motion and the distance between the models is small. As the car takes a turn, the distance increases due to a change in the motion model. The distance again decreases as the car moves straight into the parking lot. It was seen that this method was effective in identifying segments of the video where the cars change their direction of motion and enter or exit the scene. These segments can be considered as the useful portions of the video for further analysis.

### 4.3. Matching motion patterns across viewpoints

Our algorithm can compare motion patterns across changes in scale and viewpoint. To demonstrate this, we used the UCR Videoweb dataset which consists of video sequences captured using eight cameras placed in different locations around the scene. Neither the scale nor the direction of the camera is the same between two views. Since the videos are low to medium resolution, it is assumed that a change in viewpoint leads to an affine transformation between the corresponding images.

To match the motion patterns, we first need to align the two patterns using an affine transformation. Instead of com-

Figure 8. The figure shows two viewpoints of an activity taken from the UCR dataset. a) shows sample frames of each video marked with the cluster region. b) shows a plot of the streaklines of the figure on the left in a). c) shows a plot of the streaklines of the figure on the right without alignment. d) shows the plot of the streaklines of the figure on the right with alignment. The distance between the two models without alignment was found to be 3.39 and with alignment was found to be 0.87.
puting the correspondences between the two scenes using
local features, we use the streaklines to define the affine
transformation. Since we are not aware of the temporal
or spatial alignment between the scenes, we first compute
an average of the streaklines over the entire cluster. We
then resample this computed average at uniform intervals
to minimize the error in time between the two intervals.
An affine transformation is computed between these two
curves. This is considered as the affine transformation over
the entire cluster. The dynamical model is computed over
the transformed clusters and then compared. An example of
sequences matched across views is shown in Figure 8. It is
seen that the distance between two models decreases signifi-
cantly with alignment. This shows that streaklines can be
used to find similarity between motion patterns in a robust
manner.

5. Conclusion

In this paper, we have explored the use of streaklines
in analysis of motion patterns in a scene. We have dealt
with complex videos containing multiple motion patterns
occurring over a period of time. It has been illustrated that
streaklines can capture the spatio-temporal variation in mo-
tion. Similar motion patterns result in similar patterns of
streaklines. The task of motion pattern analysis involves
segmentation of the streaklines into clusters, each represent-
ing a motion pattern and modeling of these clusters. We
have shown that the motion pattern can be considered as
an ARMA process and shown how the ARMA model can
be derived from these clusters. We have shown a represen-
tation of these clusters that is robust to changes in view-
point and scale. We have conducted experiments on natural
video sequences and shown that our method is successful
in identification and modeling of motion patterns. We have
demonstrated some application of our method to tasks such
as unsupervised grouping of motion patterns, change detec-
tion and matching of motion patterns across views. Future
work would involve an extension of the method to a larger
set of activities and increasing the robustness of our algo-

References