Spatio-Temporal Data Mining, Visual Analytics for Video Annotation

Master thesis submitted to Delft University of Technology in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in **Digital Media Technology**
Faculty of Electrical Engineering, Mathematics and Computer Science

By

Yun Zeng

Student number: 4627636

To be defended in public November 15, 2017

Graduation committee:

Chairperson	: Prof. Dr. Elmar Eisemann – Computer Graphics & Visualisation
Supervisor	: Dr. Anna Vilanova – Computer Graphics & Visualisation
Committee Member	: Prof. Dr. Hans Tonino – Embedded Systems
Industry Partner	: Neda Sepasian –TKH Security Solutions - Siqura B.V.

Spatio-Temporal Data Mining, Visual Analytics for Video Annotation*

Yun Zeng
Delft University of Technology
2600 AA Delft, The Netherlands
Y.Zeng-3@student.tudelft.nl

Abstract

The abundance of video surveillance footage stimulates an emergence of video analysis researches. These researches call for large amount of annotated datasets for tasks like information retrieval, learning-based network training or algorithms evaluation. However, available annotated video datasets covering spatio-temporal information are limited. Moreover, manual annotation is a costly and tedious task, thus makes the generation of annotated data more difficult. In this work we propose a semi-automatic system for video annotation. We employ object tracking to automatically locate the objects and involve visualization to facilitate video data identification, characterization as well as comprehension. It works as an integrated visual analytics system which supports collecting and labeling object/action samples from video on demand. The annotated samples are to be applied to various video analysis researches and applications. In this paper, we present the visual design and propose the solution to semi-automatic spatio-temporal data mining and annotation.

1. Introduction

The advancements in camera technology and the decrease of memory storage costs in the last decades have led to an explosion of video surveillance footage. As a consequence, an increasing number of applications and researches for automatic video analysis emerged, lots of efforts have been put into the development of powerful detection, tracking and recognition approaches[20]. Recent progress in learning-based video analysis researches show bright prospect[3, 36, 28, 24]. Many approaches[26, 50] have demonstrated the power of data-driven analysis given labeled video footage[38]. However, the current annotated video datasets that can be used for training, evaluation or retrieval are limited, especially for motion related researches which require both spatial and temporal[1] information to analysis how objects behave over time (i.e. motion pattern). Since (i) Annotated video datasets with spatio-temporal(e.g. trajectory)[2] coverage are scarce. (ii) In existing annotation works (e.g. *VIRAT*[30] and *iVAT*[6]), the process is often simplified by annotating periodically and using automatic interpolation to recover the annotations in-between key frames[30], while resulting in loss of motion details. (iii) A part of the datasets with trajectory coverage (e.g. *GeoLife Trajectory Dataset*[52] and *T-Drive Taxi Trajectories*[47]) provide spatio-temporal information on geographical level, which do not apply to researches based on scenes such as a parking lot, an intersection or a shop entrance. (iv) Considering the varying nature (i.e. different lightings, viewpoints, background activity, etc) of visual environments[20]. In order to apply to diverse scene conditions, artificial neural networks need to be trained, and algorithms need to be evaluated, by various annotated samples (which are in shortage). (v) Existing annotations lack details, namely the labels are too general. Even in the most detailed *VIRAT*, although the events are labeled as "Person gesturing","Person running", etc., it still cannot fulfill user's more specified (retrieval) demand.

The limitation of annotated video datasets constrains video researches from delving deeper and scaling up to more scenarios, especially learning-based researches which relies heavily on training samples with spatio-temporal information, e.g. anomaly behavior detection. Moreover, video information retrieval and quantitative performance evaluation of computer vision techniques also need large scale annotated datasets. Hence the generation of such datasets are in great demand. The main challenge of this goal is that manual annotating of video data usually costs huge amount of time and man power. The widely used *VIRAT* video dataset consisting of 26-hour surveillance footage of cars and people costs tens of thousands

*This work is performed at TKH Security Solutions - Siqura B.V.

[1] Spatial refers to information based on single image(frame), temporal refers to information based on connection over frames.

[2] Trajectory is a typical spatio-temporal information. Within the scope of this paper, by spatio-temporal we mainly refer to trajectory unless specially stated.

of dollars, and requires up to a year of continuous work to annotate[38], despite sacrificing part of details by interpolation for the sake of less labor. Furthermore, to annotate both spatio and temporal information precisely will be more time-consuming. The manual generation of precise annotations requires the annotator to draw accurate bound of each object on each frame in order to trace the trajectory precisely, an overwhelming repetitive labor work. Therefore, (semi-)automatic video annotation of precise spatio-temporal data is needed, which can be implemented by employing object tracking algorithm (also known as tracker) to automatically locate objects across frames. Thereby support the generation of (and fill the lack of) annotations meanwhile alleviate the burden of the manual annotation. Then considering the automatically collected data are unorganized, which requires user supervision to to screen and organize the results.

In this work we build an integrated visual analysis system that allows for annotating spatio-temporal video data[3] semi-automatically. We involve object tracking to automatically locate moving objects across frames, and apply data visualization to facilitate identifying object/action of interest according to various spatial and/or temporal features or properties. By interacting with the visual data using given selector, filter and marker, the user can identify object/action with certain characteristics and label on demand. The annotated output can be used in multiple computer vision researches or applications as training samples of artificial neutral networks, ground-truth of video analysis algorithms performance evaluation or archives for fast information retrieval. The data exploration process can also facilitate the user's comprehension of motion pattern (e.g., representative tracks or common trends shared by different moving objects) and gain insights on video features (e.g., how discriminative each feature is, what characteristic it can describe and it suits for distinguishing which object/action, etc.).

The primary contributions of this paper are:

- We present an integrated visual analytics system that facilitates annotating spatio-temporal data in video by (semi-)automatic[4] methods, effectively alleviates the amount of manual repetitive labor while guarantees the quality.
- We present a visual design with linked views that can facilitate identifying object/action with certain characteristics in video.
- We introduce several visualization modes for representing video data which facilitate study of motion patterns and comprehension of video/image features.

[3]In current stage, we mainly collect image patches and trajectories of objects.

[4]Automatic tracking combines manual selecting and labeling.

2. Related Work

In this section we review some of the related researches in Video Data Mining, Visualization and Annotation.

Spatio-temporal information is of great importance to motion related research (e.g. action recognition). The methods used in such research can be briefly grouped into two categories: feature-based and deep learning-based methods [5]. Feature-based methods generally start by detecting spatio-temporal interest points (e.g., changing pixels other than constant ones) and then describe these points with motion characteristics(e.g. moving direction)[25]. Zelnik- Manor and Irani [51] use marginal histograms of spatio-temporal gradients at multiple temporal scales to cluster and recognize events in video. Botchen, et al. [10] design a system to enable recognition of primitive actions and their spatial and temporal relationship. They characterize motion by a descriptor splitting optical flow vectors into four channels. Other typical descriptors for spatio-temporal features include Histogram of Optical Flow (HOF)[23], 3D Histogram of Gradient (HOG3D)[22], SIFT-3D[35], Extended SURF[43], Motion Boundary Histogram (MBH)[46], global Color Moment Feature[48] and Cuboids[13]. Recently, Wang et al. propose Dense Trajectory[39], which densely sample local patches from each frame at different scales and then track them in a dense optical flow field, and the latter Improved Trajectory[41] with improvements referring to [18, 23, 37]. Bolbol, et al. [9] analyze the discriminative ability of four properties (i.e. speed, acceleration, distance, and rate of change in heading) of trajectory on six transportation modes (i.e. bus, car, cycle, train, underground, and walk) and identify the speed and acceleration as the best discriminative properties.

The aforementioned spatio-temporal feature extraction of video generally result in complex high-dimensional descriptors (that describes certain characteristics of the objects/movements), which are not intuitive for perception thus call for visualization techniques to facilitate human's comprehension. Caspi, et al. [11] proposed a method for generating visual summaries of video.They fuse key frames into a single static image (dynamic still) or organized into a short video clip representing the essence of the action (clip trailer). Romero, et al. [11] present Viz-A-Vis, an overhead video capture and access system for activity analysis. They use heatmap, denoted as activity table, to visualize the intensity of motion in a video. The intensity of motion at each pixel is obtained by the dierence between neighbor frames, and is mapped to the hotness of color. Schoemann et al. [32, 33] present a novel video browsing tool visualizing video motion by color. Motion vectors are mapped to the HSV color space such that motion direction and intensity statistics become visible by color and brightness vari-

[5]Deep learning-based methods are beyond the scope of this paper.

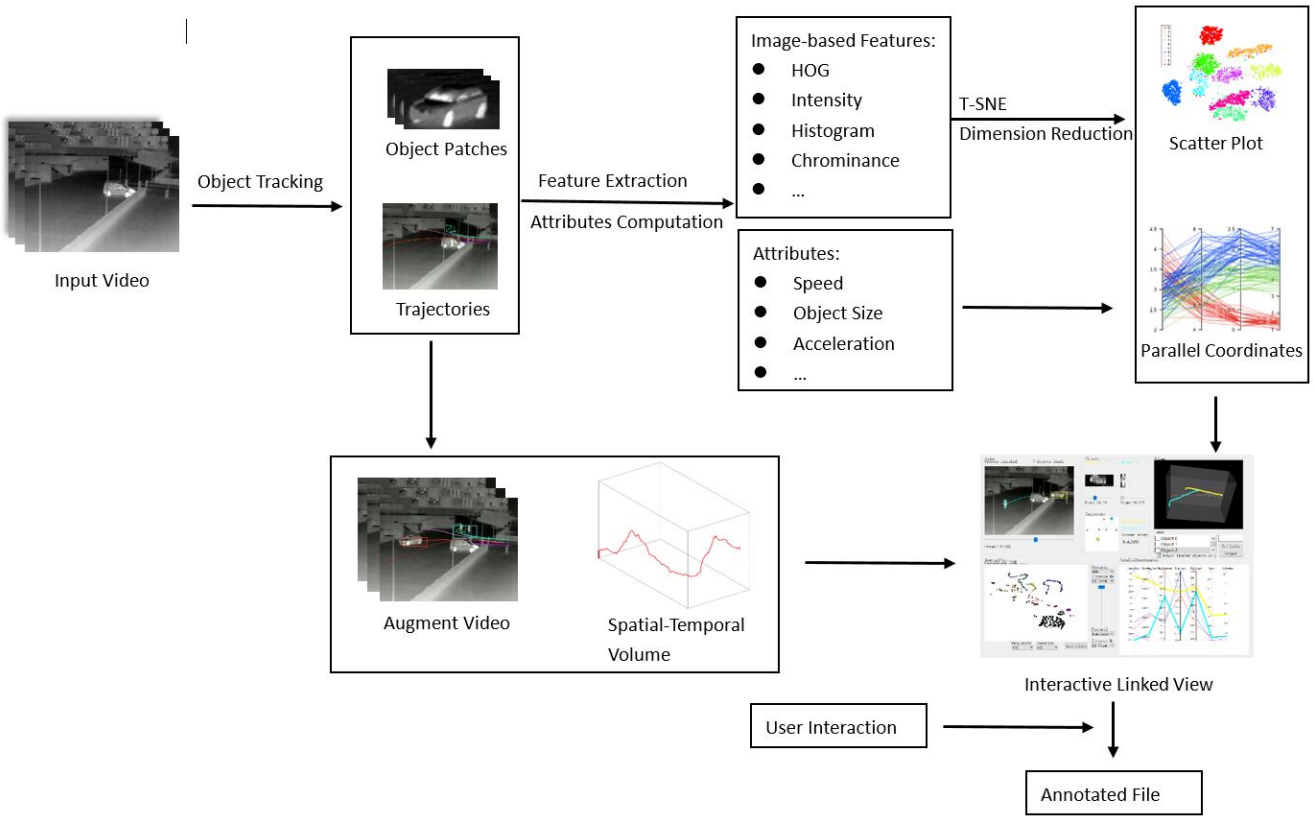

Figure 1. Workflow of our Visual Analytics and Annotation system. Video is first process to collect information of the moving objects, including image patches of objects and trajectories. Next, draw bounding boxes and trajectories of objects on the top of the original video to obtain Augmented Video(on top-left of the linked view). Extend the trajectories in temporal space and draw them in a 3D Spatio-Temporal Volume(on top-right). Extract aforesaid information further to get features and properties, image-based features are to fed to T-SNE Dimension Reduction to generate a 2D scatter plot(on bottom-left), properties such as Speed are to generate parallel coordinates(on bottom-right). These visualizations integrate in an interactive and linked view, which allows users to supervise and export annotated data.

ations. Howe, et al. [16] develop an interactive browser for a sample of the dataset of figures. They use different combination of clustering and dimension reduction methods to map the distribution of figure vectors to 2D space. In [10] depicts a video stream as a series of continuing video volumes, which displays snapshots at relatively sparse interval, and highlights the trajectory of moving objects in 3D spatial-temporal space.

The comprehension of the video obtained from previous process is meaningful for video research, thereby it is important to be recorded for later query (so that the complicated comprehension process can be skipped or at least simplified in subsequent study), in form of annotation. Much of the computer vision progress has been enabled by the availability of public annotated datasets, such as the *KTH* [34] and *Weizmann*[8]. According to the research in [4, 21], the performance of classifiers improve dramatically when a conspicuous set of labeled training data is available[20]. Stimulated by the demand of annotations, tools for annotating start to occur and draw attention. Höferlin, et al. [15] present an integrated visual analytics system which supports comprehension of ad-hoc training classifiers and facilitate overview and efficient annotation by visualization. They introduce a novel visualization called *cascaded scatterplot* that integrates multiple dependent scatterplots to visualize the class distribution of data instances in each stage of the classifier. Yuen, et al. [49] introduced *LabelMe* video, an web-based platform that bounds object with polygonal and uses linear interpolation to locate object in-between frames. Ali et al. [1] present *FlowBoost*, a tool that can annotate videos from sparse set of key frame annotations. Kavasidis et al. [19] present *GTTool*, a tool for generating ground truth data for object detection, tracking and recognition applications. They improved previous propagation strategy (based on interpolation of the boundaries of an object between the starting and ending frame) by using object tracking techniques for automatic detection of objects across frames. Their annotation shows impressive ef-

ficiency with high accuracy. Other annotation tools include *ViPER-GT*[29], *GTVT*[2], *Inter OD*[45], which, however, show their limitations when it comes to generate precise spatio-temporal annotations.

3. System Overview

The proposed system should be able to track objects in video and derive spatial and temporal information, generate meaningful visualizations that represents motion(i.e., change in position of an object over time) related characteristics of the video more intuitive and comprehensible, and aid the user annotate these data on demand. In Section 2, we present a number of tools that are available and widely used for video annotation. However, none of these tools provides spatio-temporal annotation. Since humans have difficulty in perceiving space and time simultaneously, designing this graphical interface presents subtle challenges that, if not properly addressed, make video annotation unnecessarily labor intensive. In this paper, we demonstrate an efficient interface for video visual analytics and annotation, inspired by related works, shown in Figure 3. Details of the views will be described in Section 5.

It has been developed using C++, Qt libraries for the Graphical User Interface (GUI), and Open Computer Vision libraries (OpenCV) for computer vision algorithms. D3.js has been involved to draw two plots and the 3D volume is rendered using VTK libraries. The developed tool integrates a number of computer vision and visualization techniques, with the purpose of enhancing the annotation generation process in terms of efficiency and spatio-temporal coverage, as well as the video data comprehension.

The scheme of the system is presented in Figure 1. Given an input video, we start by tracking and stabilizing each moving object present in it, collect (image patches and trajectories of) the objects using video processing algorithms. This gives us a object-centric spatio-temporal sequence for each object which can be used to generate *Augmented Video* and *Spatio-Temporal Volume*. The objective is to get rid of the background and focus on motion information. Then derive several feature descriptors and properties based on collected data, in order to describe the object in terms of certain characteristics. Synthesize two visual representations to show the distribution of data with respect to certain feature/property. The four main visualizations then integrate into an interactive environment with linked views, which allows the user to explore, select and label data of interest and output annotations in certain data structure. Details of the individual modules are discussed in the following sections.

Note that in current stage of our work, we only discuss video captured from static fixed lens (which can be commonly found in a surveillance footage), and whose scaling effects (result from different distances between the objects and the camera) are negligible. We also ignore the possible

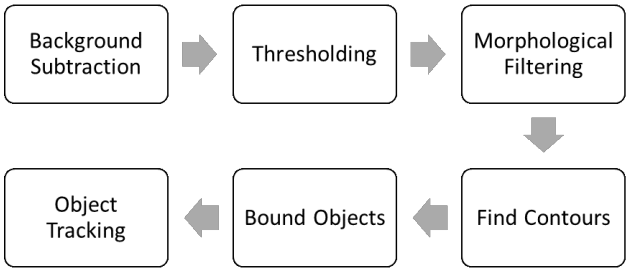

Figure 2. Flowchart of Video Processing

affects come from jittering, change in background, etc. and assume the tracker can locate and bound the objects with certain accuracy and high preciseness.

4. Video Processing

The Video Processing section processes received raw video data, aims to extract motion related information, namely location and pixel values of each moving object, collect corresponding image patches (as a set of images) and trajectory (as a sequence of spatial points) in order to provide information about how the objects move and be used for the feature and property extraction later. Figure 2 shows the procedures of our proposed video processing approach.

Notably, video data is composed of numerous interrelated pixel signals and contains high rate of redundancy which makes browsing time-consuming. Pixels of moving objects generally provide more information about what happen in the scene than (static) backgrounds. Based on this perception, we subtract background information and only focus on motion information in video. For this purpose we employ *BackgroundSubtractorMOG2*, a Gaussian mixture-based background/foreground segmentation algorithm to find pixels which are likely to be background and which are not. One important advantage of this algorithm is its better adaptability to varying scenes due to illumination changes etc[53, 54]. The result from background subtraction contains noise, thus thresholding and morphological filtering are applied to reduce noise. To this step, each pixel is identified as either background pixel or foreground. Each connective region consists of foreground pixels can be denoted as a moving object region. Typical video annotation starts with locating objects of interest by drawing a boundary (the most basic and easily drawn is bounding box) around each object. Theoretically, these boundaries should be drawn on every frame to produce accurate track of each object. In (semi-)automatic annotation, to largely relieve the manual labor, object tracking techniques (trackers) are used to locate instances of the same object across frames[38]. In conjunction with the initialization (draw a bounding box around each moving object), a tracker is used

which takes the objects identified in the previous frames as input and suggests associations with the objects localized (automatically) in the current frame. We take advantage of the accuracy and processing speed of *KCF Tracker* developed in [14] to locate the objects in successive frames[6]. The image patches(i.e., sub-image, pixels within each bounding box) and trajectory(i.e., a sequence of bounding boxes' central positions in temporal order) of each object are recorded, and to be fed to generate visualizations in the next section.

Note that the quality of object tracking is of great importance to the following steps. Although we introduce a module (detailed in Section 5.3) to examine the tracking results and exclude flawed ones. In order to feed the visualizations with more useful information for object/motion identification, the bounding boxes are supposed to be accurate in terms of location and size.

5. Data Visualization

The process in Section 4 results in an object list, where for each object x, there records the image patch and position (the bounding box) of x at a discrete temporal point t $\in$ N, $X(t) = [x_1(t), x_2(t), ..., x_n(t)]^T$. To label each x, the objects/actions need to be identified. In this section, we visualize the data by adopting several visual modes, in order to enable viewers to identify and characterize different detected objects/actions, associate spatial and temporal features with object/action categories, and to empower users to use their superior perceptual reasoning skills to detect and label samples of interest and gain insight on the features. We aim to examine the feasibility of obtaining insight into the features/properties (e.g., what character the feature describes, the property is discriminative for what kind of objects/actions in which circumstance), by arousing the user's comprehension through exploring the visualized data, rather than informing the user of an explicit conclusion which can be very unreliable or partial.

Aided by the visualization, the user should be able to:

1 Identify and characterize different objects/actions and search for objects/actions of interest on demand.

2 Relate selected object/action to the original video.

3 Examine the results of automatic object tracking and exclude flawed one from annotation.

4 Set label to each selected object and export the annotated data(mainly image patches and trajectory as a sequence of points).

5 Associate spatial and/or temporal features/properties (or combinations) with object/action identification.

[6]We provide 6 optional trackers: BOOSTING, MIL, KCF, TLD, MEDIANFLOW, and GOTURN and use KCF by default considering its superiority in speed.

We provide multiple linked views to tackle these tasks. The whole view of our visual design can be seen in Figure 3, the highlight in each view is dynamic in respond to user interaction. A unique color is assigned to each object in order to identify data of this object in all the views, since human eyes are good at discriminating colors.

5.1. Video Viewer

As a fundamental requirement for video annotation, the video sequence should be able to be browsed frame by frame. It is also essential to aforementioned Task 2, which can be achieved by retrieving the frames and regions where the selected object is active. We employ *focus-and-context*[10] visualization techniques, which combine the display of extracted (motion related) data with original video frames: the former is the focus(highlight), the latter serves as context. The *Video Viewer* shown in Figure 3(a) allows to browse the augmented video by sliding the slider, there displays the frames sequentially with the bounding boxes and trajectories superimposed to them. This is especially helpful when checking if the objects have been tracked correctly. When an instance (in Image Scatterplot, stands for an image patch) is selected, the viewer will position to the corresponding frame and highlight the region where the instance occurs. Thereby each object/action is connected to the original video.

5.2. Spatio-Temporal Volume

In order to achieve Task 1 and 5, the user needs to perceive space and time simultaneously, thus we provide a good visibility of motion traces along the temporal axis to facilitate the comprehension of temporal behavior of the objects. The 3D volume in Figure 3(b) displays the extracted trajectories of moving objects in Spatio-Temporal space, by extending the trajectory sequence in temporal order along Z-Axis. But only showing the trajectories conveys restricted amount of detailed information, since visual information about the environment is lacking. Therefore, We again employ *focus-and-context* approach, here we combine the display of trajectories (focus) with the starting frame as background (context). We also show the image patch at respective space-time position when an instance (in Image Scatterplot) is selected, to provide more details (at $x_n(t)$).

The 3D environment allows for rotation and zoom in/out the volume to view the trajectories from different angles and in general/detail. In general, the user is able to see all the performed action sequences that appeared in a video in one continuous illustration at one glance, instead of browsing the whole video. In detail, the user can easily discriminate trails in terms of shape (straight or winding), heading direction, etc. and identify certain actions (Task 1), e.g., a car moving straight or turning. As well as associate these properties with object/action categories, for the purpose of infer-

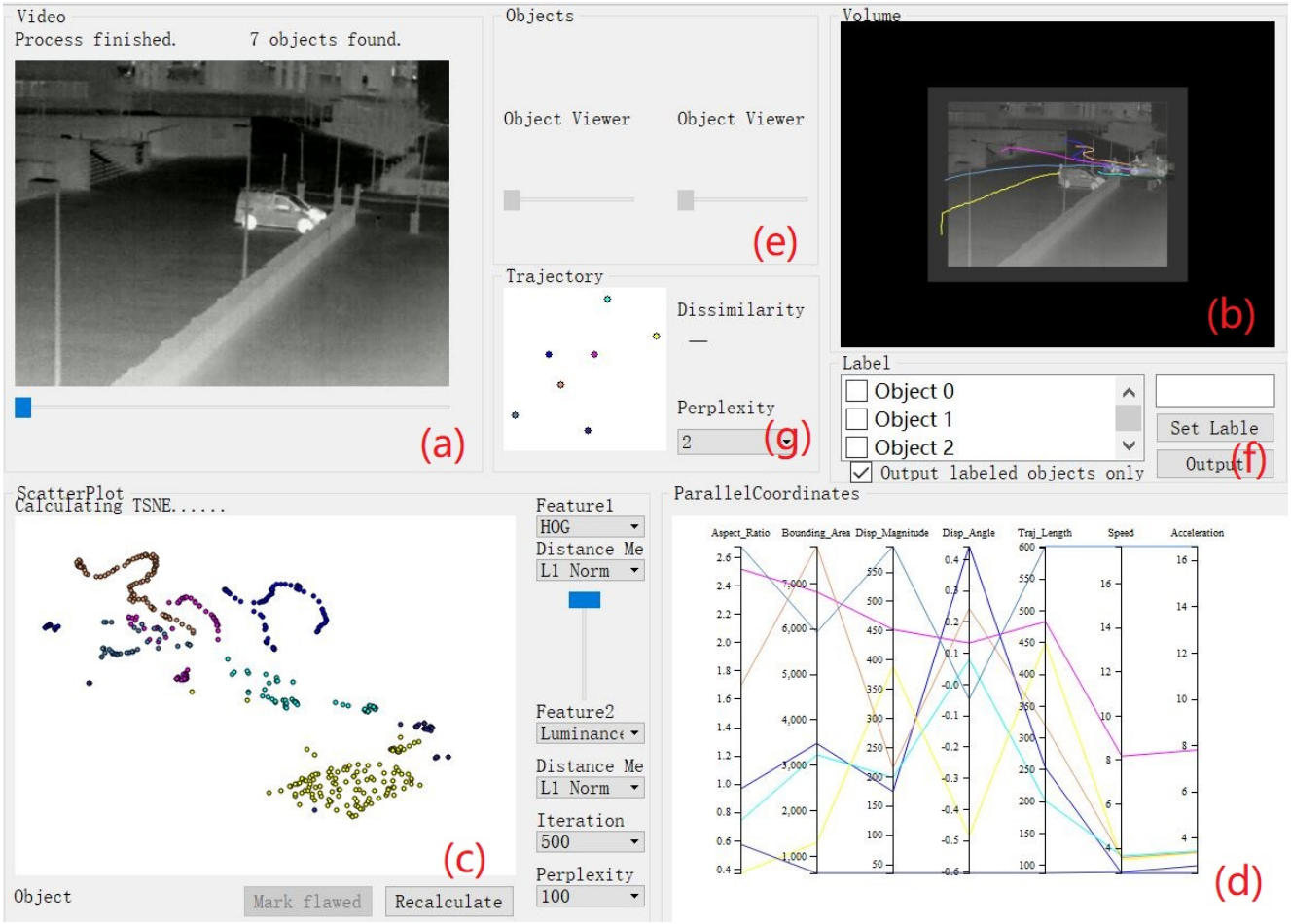

Figure 3. Workspace of our visual analytics and annotation system. (a)Augmented Video viewer. (b) S-T 3D Volume, visualization of trajectories in Spatio-Temporal space. (c) Image Scatterplot, 2D T-SNE embedding of image patches. (d)Parallel Coordinates, visualization of proposed properties. (e)Object Viewers, view of selected image patch. (f)Labeling Panel, used for setting labels by the user. (g)Trajectory Scatterplot, 2D T-SNE embedding of trajectories

ring discriminative properties (Task 5), e.g., if the trails of cars turn out to be more straight than trials of pedestrians, *straightness* would be a property to distinguish cars from pedestrians.

5.3. Image Scatterplot

Aiming to tackle Task 3, we need a module to spot flawed tracking results which should be excluded from annotation. The view shown in Figure 3(b) depicts the distribution of the images patches based on features descriptors. It is a 2D embedding of high-dimensional descriptors by dimension-reduction method (detailed in below). We figure out flawed result by searching for unexpectedly distributed instances.

5.3.1 Feature Descriptor

Due to the imperfection of the tracker[7], namely part of the data collected in object tracking process is flawed which should be excluded from annotation. Instead of manual inspecting the tracked object data one by one, we utilize a visual module to assist inspection.

Each X(t) contains an image patch $x_i(t)$ that can be interpreted as a single high-dimensional data point, hence our tracking results consist of a large number of high-dimensional data points. Since the original data points contains high rate of extraneous information which makes comparison difficult, we introduce feature descriptors that simplifies the image by extracting useful information (that can better discriminate the images) to represent the original data. Each descriptor can also be interpreted as a sin-

[7]The KCF tracker we use claims to have an average 73.2 % accuracy, a relatively high quality among typical trackers.

gle high-dimensional (lower than the original) data point. We depict the distribution of the descriptors and examine the tracking results visually. In general case, images of the same object (assigned with the same color in the views) are supposed to be clustered in the plot (considering that they are probably more similar and closer to each other in distribution). Images that do not conform to this, are likely to be flawed samples. Notably, the tracker searches for target object by comparing certain features of images. Thereby the descriptors are supposed to use features in principles differ from the tracker, in order to provide complementary information that the tracker is missing. For example, *KCF* we use by default is based on low-frequency component from Discrete Fourier Transform(DFT) of image, so we need some feature that describes image details (since the low-frequency component lack detail information) to examine the results from *KCF*. Any image has information distributed in the spatial or color dimensions. The features were proposed so as to exploit information of these dimensions. We introduce several features that are widely used for image identification, include:

- *HOG. Histogram of Oriented Gradients* counts the distribution of oriented gradients in localized portions of an image. It mainly describes local object appearance and shape. *HOG* descriptor is particularly suited for human detection[12], and has shown its outstanding performance for images in which the edges are complete and clear. While due to its scaling-variance and rotation-variance inherence, it is incapable in case of non-ignorable scaling or rotation effects.

- *Intensity/Color Histogram.* Analogous to *HOG*, Intensity/Color Histogram is a representation of the distribution of luminance or colors in an image. It counts the percentage of pixels within each intensity or tonal interval. In [7, 31] Color Histogram shows higher performance than Intensity on head tracking. Color Histogram does not apply to gray-scale image while suits for describing objects with special color. Intensity Histogram is sensitive to luminance change in the scene and the influence of background.

- *Single-channel Pixel value*. Color image consists of pixel values in RGB channels, we convert the original data to HLS (Hue, Lightness, Saturation) and YCbCr(Luma component, blue-difference and red-difference chroma components) spaces, and provide pixel values in each single-channel as descriptors. These descriptors preserve whole information in corresponding channel while lose all information in the rest channels, suit for distinguishing images whose information of one-single channel is dominating.

As a result, each image patch is represented as a feature descriptor (in form of a high-dimensional data point) based on one of the listed feature. The support of examination relies on the discriminating of images through the feature, namely the feature descriptors directly impact the capability of examination.

5.3.2 Dimension Reduction

In general, images of the same object are supposed to be more similar than other objects, represents as instances (of image descriptors) of same object closer to each other (cluster) in distribution. Considering that instances regarded as the same object by the tracker are assigned with a unique color, it can be expected that instances in same color will cluster in distribution if the object is tracked correctly. While instances that do not conform to the expectation are likely to be flawed, we can spot the expectedly distributed instances through visual observation.

Nevertheless, the aforementioned image descriptors are high-dimensional data points that originally distributed in high-dimensional space thus difficult to be intuitively presented for visual examination. We address this problem by introducing dimensionality reduction technique to map the high-dimensional data into low-dimensional (generally 2D or 3D, we use 2D) space. Such techniques take the high-dimensional distances of instances (by comparing corresponding high-dimensional data points, using distance measures such as Euclidean) as input, map the data points into low-dimensional space while reserve the original distances in certain scale. Linear dimensionality reduction algorithms (e.g. *Principal Component Analysis (PCA)*[44]) concentrate on placing dissimilar data points far apart, attempt to preserve geometry at all scales. But in order to cluster images of same object, we focus on local structure, i.e., similar data points must be represented close together. For this purpose, we use *t-Distributed Stochastic Neighbor Embedding (t-SNE)*, a dimensionality reduction algorithm developed by Van der Maaten and Hinton[27], particularly well suited for the visualization of high-dimensional datasets. *t-SNE* preserves the distances between nearby data points while distorts long-distance relationships. The concentration on local structure tends to better cluster instances of the same object (more similar). We plot the 2D embedding in a scatterplot shown in Figure 3(c). Each colored dot indicates an (image)instance which will show in the Object Viewer(3(e)) when the dot is selected.

By this view, the users can quickly spot and exclude the data that incorrectly tracked(Task 3). Furthermore, although not designed for, through the exploration, the user can also get visual feedback about the performance of each spatial(image) feature for distinguishing objects (a discriminative feature is supposed to allow instances of same ob-

ject well clustered in the scatterplot), thereby associate features with object/action identification(Task 5). It also facilitates comprehending the nature of the tracker, gives a clue what (information missing) leads to the failure of tracker and what feature/information could be supplemented to improvement the tracker[8].

5.4. Parallel Coordinates

To complement the illustration in the above views, facilitate object/action identification and characterization(Task 1), and discriminative property inference (Task 5), we derive a *spatio-temporal signature*, $S(s_1, s_2, s_3...s_7)$, from $X(t)_t$, for each object x and visualize the signature in parallel coordinates (See Figure 3(d)).

5.4.1 Property Extraction

$S(s_1, s_2, s_3...s_7)$ exploits some characteristics of objects/actions that are not (well) illustrated in former views, each dimension s_i refers to a derived property that characterizes the objects/actions, include:

- *Aspect Ratio*. Ratio of width and height of bounding box. When calculating *HOG* and *Single-channel* in Section 5.3, the image patches are resized to a fixed size in order to make sure the descriptors in same dimensionality (comparable). However, due to the resizing, the original aspect ratio of image patches are lost (in case of *Intensity/Color Histogram*, no bounding box information involved), which is possible to discriminate some objects. For instance, the aspect ratio of image patch of a pedestrian (width ¡ height) is likely to be different from a vehicle (width ¿ height) in general. This property is useful if the objects to be distinguished differ a lot in aspect ratio, provided that the bounding box has bounded the object accurately (i.e., the size of bounding box is approximate to the size of object).

- *Boundingbox Area*. Multiplication of height and width of bounding box. Analogous to *Aspect Ratio*, except that it particularly suits for distinguishing objects differ a lot in area, e.g., duck and vessel.

- *Displacement* (Magnitude and Angle). 2D vector pointing to the ending position from starting position of the object. Measuring how far the object moved in the scene. This property suits for distinguishing objects moving in different direction(angle) or distance(magnitude), e.g., turn left and turn right.

- *Trajectory Length*. The length of trajectory of each object, measuring how long the object has covered. It can work with *Displacement* to identify certain movement. For example, large trajectory length with small displacement indicates wandering-like movement. The opposite indicates moving straight.

- *Speed*. Average moving speed of each object, equals to length of trajectory divided by occurring time. It is useful to distinguish objects that are supposed to have differentiated moving speeds, e.g. pedestrian and motorcycle.

- *Acceleration*. Average difference of speed. Measuring the uniformity of movement. This property suits for distinguishing movements differ in uniformity, e.g., hopping from walking, and speeding up from slowing down.

Note that by deriving the above properties, we attempt to quantify some real properties of the object/movement, e.g., using moving speed in video to quantify the moving speed in reality, and apply them to discriminate objects/movements that differ a lot in one or multiple real properties. Therefore, a property can work and make sense on the assumption that it truly reflects the property in reality.

5.4.2 Signature Visualization

The *spatio-temporal signature* is multivariate data consists of seven properties (dimensions). We have carefully chosen *parallel coordinates*[17] to visual the signature for their analytical capabilities in analyzing different aspects of multivariate data and revealing relationships between data dimensions. The design (Figure 3(d)), with polylines describing multivariate items that intersect with parallel axes representing variables, can be used for the analysis of multiple properties[42], e.g., identifying multivariate outliers, trends, and clusters. Also consider that there are seven dimensions to depict, the parallel axes can be properly spaced(neither too clutter nor too sparse).

Each polyline (segments in same color) in the parallel coordinates view corresponds to the signature of an object. The usefulness of this view lies on (i) The order of axes in parallel coordinate is switchable to place arbitrary axes adjacent. Relationships between adjacent dimensions are easier to perceive, e.g., one can place *Speed* axis next to *Boundingbox Area* axis, and inspect if large objects tend to move faster or conversely. (ii) It is easy to detect outliers in parallel coordinates, e.g. something moves extremely fast(high in *Speed*) or heads to a special direction (*Displacement-Angel*), these samples are useful for anomaly detection/recognition. (iii) It also allows for brushing, the user can brush over a certain range on an axis (e.g. high value of speed), and highlight therefrom the instances of corresponding objects in all views, thus filter down to a subset for further exploration. (iv) It helps to

[8]We do not discuss details here since improving the tracker is beyond the scope of this paper.

identify clusters, which present as dense lines. Thus infer properties that can distinguish objects/actions (Task 5) and characterize objects/actions(Task 1).

5.5. Trajectory Scatterplot

Analogous to the *Image Scatterplot*, we propose *Trajectory Scatterplot* to illustrate the distribution of trajectories, so as to identify trajectory clusters (similar trajectories) and associate with objects/actions. For example, if objects from a certain category generate similar trajectories, it can be interpreted as a motion pattern of this category (Task 1, characterization).

Nevertheless, to depict the distribution of trajectory sequences (high-dimensional data) in low-dimensional space analogous to Section 5.3, we need to derive distances(dissimilarities) between each pair of trajectories. Different from other simple data types where the distance is well-defined, the distance between trajectory has no general definition. As trajectory is high-dimensional data and contains spatio-temporal information, both of which needs to be considered when measuring[40]. Besides, since the length of data vary from trajectory to trajectory, typical measures (e.g. Euclidean) cannot be easily adapted to measuring trajectory distance. According to the study of Wang, et al. in [40], we choose *Dynamic Time Wrapping (DTW)*, a well-known measure of trajectory dissimilarity, taking advantage of its capability of comparing sequences of different lengths, pertinence in time-series and flexibility in cost function. *DTW* is derived from [5], initially used in audio analysis, aims to find an optimal alignment between two given (time-dependent) sequences.

In detail, *DTW* uses a recursive manner to search all possible point combinations between two sequences (i.e. combine each point in one sequence with a point in the other sequence) for the one with minimal distance[40]. Given two sequences $P = [p_1, p_2, ...p_N], Q = [q_1, q_2, ...q_M]$, the distance $D(P, Q)$ is defined as the accumulation of distance between every pair of combined points $\sum d(p_{nl}, q_{ml})$. The measure of distance between a pair of combined points is known as *cost function*. Since we focus on figuring out motion(change in position) pattern, we define the cost function as difference between the tangent vectors, $d(p_{nl}, q_{ml}) = |tp_{nl} - tq_{ml}|$, tp_{nl} and tq_{ml} are the tangent vectors at p_{nl} and q_{ml} respectively. By this approach, we leave out the absolute position of each data point, only consider how the object is moving (change in position in each interval). It is worth mention that our measure is translation and rotation invariant. The disadvantage of *DTW* is that it is sensitive to noise and decrease sample rate, which is not a problem on assumption that the trajectories are tracked accurately and precisely.

Figure 3(g) displays the scatterplot of trajectories, it is a 2D embedding generated by t-SNE based on above-defined dissimilarity of trajectories. This view and *Image Scatterplot* complement each other, the former illustrates distribution of objects in terms of appearance (image), the latter illustrates distribution in terms of movement (trajectory).

5.6. Interaction

The different views in the workspace are linked to each other through interaction. This facilitates selection, analysis and exploration:

- By clicking a single instance in the Image Scatterplot, this instance and the corresponding instance in Trajectory Scatterplot will enlarge, the Video Viewer will jump to the frame where the selected data instance occur with a mask highlighting the object region, trajectory in S-P Volume and signature in Parallel Coordinates of the object will show in larger thickness, corresponding image patch will show in the left Object Viewer and the slider of the object view will be enabled for browsing all the image patches of the object in chronological order. Compare Figure 3 with Figure 8 to see the effects.

- By hovering in a single instance, the response is similar to clicking, excerpt for all instances corresponding to the same object will be highlighted with thicker stroke, in Video Viewer where no response will happen and in Object Viewer the corresponding image will show in the left viewer. When hovering out, the corresponding data in other views will recover to former state. Compare Figure 3 with Figure 7 to see the effects.

- By brushing on the axes in parallel coordinates, signatures which contains property value that are out of the brush range will become transparent as well as corresponding instances in two scatterplots, in the labeling panel the rows of these objects will be marked with a dark filling. Refer to Figure 13, after brush, data outside the brushed range are faded in S-P Volume and two scatterplots.

- When a flawed tracking object is found, the user can mark it as flawed, corresponding instances in Image Scatterplot will be faded and the corresponding polyline in Parallel Coordinates will become dashed. See Figure 9.

6. Annotation

Using aforementioned interactions, the user can select single or multiple objects and set label(s) in the Labeling Panel ((Figure 3(f))). The label(s) given by the user will be recorded and display in the object list. The automatic tracking relieves the burden of drawing bounding on object frame

by frame, and simplifies the process into selecting object of interest using the visual tools. Result in an improvement on cost-effectiveness. The visualizations enable the user to quickly identify (action of) object and set proper label(s) without browsing the frames. Although we give the freedom to the user that they can set label in whatever scope, it can be as general as "person walking" or as detailed as "a person slowly wandering, head to southwest", the user is recommended to take advantage of perceptible information provided by the visualizations and generate detailed annotations.

The annotated output data (by default) consist of image patches(a set of images named in chronological order) , spatio-temporal data(a text file, including trajectory sequences, the signature, the given label etc) of detected objects in an organized directory. For further use of the there needs a respective parser to parse the annotated file in target projects.

7. Usage Scenario

In this section we present exemplary scenarios to demonstrate the effectiveness of this system.

Case 1, detection of flawed tracking. (i)See Figure 10. When we look at the Image Scatterplot, we can easily find that instances of the selected (thicker stroke) object separates into (mainly) two clusters, by monitoring the corresponding image in the Object Viewer, we figure out that it is actually images of two different objects, but the tracker thought they are the same object. So we should exclude this sample. This is how we can spot flawed results of the tracker with the help of Image Scatterplot.

(ii)An analogous example can be seen in Figure 11, where two (different-in-color) clusters are very close to each other on the Image Scatterplot. By selecting instances in the two clusters and monitoring the image, we find that these two clusters are actually images of the same object, but the tracker regarded them as different objects. This is another type of flawed result of the tracker.

(iii)A third type of flawed tracking is noise tracked as object (false alarm). See Figure 12, the selected pink instances are scattered in Image Scatterplot, according to the Parallel Coordinates its every property value is low, in S-T volume and Trajectory Scatterplot we find that its trajectory are quite different from others'. Any of the visual representations suggests the "object" is problematic, we can easily spot noise like this.

Case 2, identify object/action. See Figure 13, brushing on Speed axis we easily screen out two fastest moving objects in the scene. Then by comparing their accelerations, we can identify that Object 3 (yellow-colored) whose acceleration is lower moves more uniformly than Object 6 (blue-colored). This can hardly be quickly identified by browsing the video. The visualizations fasten the identification and gives subtle information.

Case 3, incomplete object. See Figure 14, using the brush we screen out a small-size object with rather high speed, namely a person-size object moves as far as car. So we retrieve to frames and figure out what happened and it turns out to be a piece of a truck. This suggests that the images, and bounding box information are incorrect. Nonetheless, by looking at its trajectory in S-P Volume and Video Viewer, we find that its trajectory completely comforts to the track of the truck. Thus we can still annotate the trajectory of the truck, although the images should be excluded. Our system can make use of incomplete objects.

Case 4, identify similar motion. See Figure 15, we easily find two objects with similar trajectories through the Trajectory Scatterplot, but their image patches are not clustered in the Image Scatterplot which indicates their images are different, the properties in Parallel Coordinates gives affirmative information (two spatial (bounding box) properties are different, the rest properties relate to trajectory show similar in value). Then by browsing in the Video and Object Viewer we can figure out that one is a motorcycle, the other is a pair of bikers pass the same route. Due to the tracking problem, the bikers are regarded as a single object. As a result they show different in appearance (images) while similar in motion(trajectories). Our system enables the user to categorize objects/actions based on trajectories.

8. Evaluation

To assess more quantified effectiveness of the proposed annotation in terms of time, we ourselves performed tests. We tested the process of video annotation using our system on three videos and recorded the time it takes. Table 1,2 and 3 shows the time for annotating video in Figure 4, 5 and 6 respectively. Consider that there is no other annotation tool that is capable of spatio-temporal annotation, the cost-effectiveness of our work is not directly comparable with other existing tools. The key quantity is *time per bounding box*, which can be compared with the time needed to manually draw each bounding box. According to our test, the system costs (in average) less than 0.2 second to draw a bounding box, which is unattainable in manual drawing frame-by-frame. Even compare with tools that only collect images, e.g., *iVAT* and *ViPER-GT* which take 0.43" and 1.15"[6] to draw a bounding box respective, our system shows superiority in effectiveness.

Note that when computing average time e.g. *time per bounding box*, we only consider time consumed in annotation, not include time consumed in video processing. Since our video processing is totally automatic, the user are free to do any other task during video processing.

Table 1. Result of Test 1

Tracker	Frames	Processing time	Objects tracked	Objects flawed	Tracking accuracy
KCF	310	135”	7	1	85.7%
Annotation time	Images annotated	Trajectories annotated	Time per object av.	Images per object av.	Time per boundingbox av.
45”	456	6	7.5”	76	0.099”

Figure 4. Video used in Test 1. A 3-channel thermal video, resolution 720*540, 310 frames.

Table 2. Result of Test 2

Tracker	Frames	Processing time	Objects tracked	Objects flawed	Tracking accuracy
KCF	200	394”	14	7	50%
Annotation time	Images annotated	Trajectories annotated	Time per object av.	Images per object av.	Time per boundingbox av.
151”	736	7	21.6”	105	0.205”

Figure 5. Video used in Test 2. A 3-channel grayscale video, resolution 476*316, 200 frames.

Table 3. Result for Test 3

Tracker	Frames	Processing time	Objects tracked	Objects flawed	Tracking accuracy
KCF	200	1426”	18	6	66.7%
Annotation time	Images annotated	Trajectories annotated	Time per object av.	Images per object av.	Time per boundingbox av.
219”	2114	12	18.3”	176	0.104”

Figure 6. Video used in Test 3. A 3-channel color video, resolution 720*480, 722 frames.

9. Conclusion and Future Work

In this paper we, for the first time, bring up the importance of spatio-temporal video annotation. And present our integrated semi-automatic system for video data mining, visual analytics and annotation. We display the visual design and illustrate the scheme and explain the functionality. We also give several usage scenarios to demonstrate the potential use of our system.

Our system faces below limitations and future work can be conducted in: (i) We deal with the flawed tracking results by simply excluding those from annotation (without considering correcting them). This results in a waste of data and heavily rely on the tracking performance, in videos where the tracker performance poorly, we can hardly annotation proper samples. The future work can direct to the scheme to correct the flawed results. (ii) Although ignored in theory, the scaling effect of video affects the accuracy of many properties in practice. In the future work, approaches to compensate the error caused by scaling would be a significant improvement. (iii) As product, the application and further study of the spatio-temporal data annotated by our system will also be much expected.

References

[1] K. All, D. Hasler, and F. Fleuret. Flowboostappearance learning from sparsely annotated video. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1433–1440. IEEE, 2011.

[2] A. Ambardekar, M. Nicolescu, and S. Dascalu. Ground truth verification tool (gtvt) for video surveillance systems. In *Advances in Computer-Human Interactions, 2009. ACHI'09. Second International Conferences on*, pages 354–359. IEEE, 2009.

[3] A. Basharat, A. Gritai, and M. Shah. Learning object motion patterns for anomaly detection and improved object de-

tection. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[4] G. W. Bassel, E. Glaab, J. Marquez, M. J. Holdsworth, and J. Bacardit. Functional network construction in arabidopsis using rule-based machine learning on large-scale data sets. *The Plant Cell*, 23(9):3101–3116, 2011.

[5] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[6] S. Bianco, G. Ciocca, P. Napoletano, and R. Schettini. An interactive tool for manual, semi-automatic and automatic video annotation. *Computer Vision and Image Understanding*, 131:88–99, 2015.

[7] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR '98, pages 232–, Washington, DC, USA, 1998. IEEE Computer Society.

[8] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1395–1402. IEEE, 2005.

[9] A. Bolbol, T. Cheng, I. Tsapakis, and J. Haworth. Inferring hybrid transportation modes from sparse gps data using a moving window svm classification. *Computers, Environment and Urban Systems*, 36(6):526–537, 2012.

[10] R. P. Botchen, S. Bachthaler, F. Schick, M. Chen, G. Mori, D. Weiskopf, and T. Ertl. Action-based multifield video visualization. *IEEE transactions on visualization and computer graphics*, 14(4):885–899, 2008.

[11] Y. Caspi, A. Axelrod, Y. Matsushita, and A. Gamliel. Dynamic stills and clip trailers. *The Visual Computer*, 22(9):642–652, 2006.

[12] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[13] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 65–72. IEEE, 2005.

[14] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[15] B. Höferlin, R. Netzel, M. Höferlin, D. Weiskopf, and G. Heidemann. Inter-active learning of ad-hoc classifiers for video visual analytics. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 23–32. IEEE, 2012.

[16] B. Howe, P.-s. Lee, M. Grechkin, S. T. Yang, and J. D. West. Deep mapping of the visual literature. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, pages 1273–1277, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee.

[17] A. Inselberg. The plane with parallel coordinates. *The visual computer*, 1(2):69–91, 1985.

[18] M. Jain, H. Jegou, and P. Bouthemy. Better exploiting motion for better action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2555–2562, 2013.

[19] I. Kavasidis, S. Palazzo, R. Di Salvo, D. Giordano, and C. Spampinato. A semi-automatic tool for detection and tracking ground truth generation in videos. In *Proceedings of the 1st International Workshop on Visual Interfaces for Ground Truth Collection in Computer Vision Applications*, page 6. ACM, 2012.

[20] I. Kavasidis, S. Palazzo, R. Di Salvo, D. Giordano, and C. Spampinato. An innovative web-based collaborative platform for video annotation. *Multimedia Tools and Applications*, 70(1):413–432, 2014.

[21] T. Kawahara, H. Nanjo, T. Shinozaki, and S. Furui. Benchmark test for speech recognition using the corpus of spontaneous japanese. In *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[22] A. Klaser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association, 2008.

[23] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[24] R. Laxhammar and G. Falkman. Online learning and sequential anomaly detection in trajectories. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1158–1173, 2014.

[25] Q. Li, Z. Qiu, T. Yao, T. Mei, Y. Rui, and J. Luo. Action recognition by learning deep multi-granular spatio-temporal video representation. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, ICMR '16, pages 159–166, New York, NY, USA, 2016. ACM.

[26] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: Dense correspondence across different scenes. In *European conference on computer vision*, pages 28–42. Springer, 2008.

[27] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[28] S. Mascaro, A. E. Nicholso, and K. B. Korb. Anomaly detection in vessel tracks using bayesian networks. *International Journal of Approximate Reasoning*, 55(1):84–98, 2014.

[29] D. Mihalcik and D. Doermann. The design and implementation of viper. *University of Maryland*, 2003.

[30] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. Aggarwal, H. Lee, L. Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *Computer vision and pattern recognition (CVPR), 2011 IEEE conference on*, pages 3153–3160. IEEE, 2011.

[31] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proceedings of the 7th European*

Conference on Computer Vision-Part I, ECCV '02, pages 661–675, London, UK, UK, 2002. Springer-Verlag.

[32] K. Schoeffmann and L. Boeszoermenyi. Video browsing using interactive navigation summaries. In *Content-Based Multimedia Indexing, 2009. CBMI'09. Seventh International Workshop on*, pages 243–248. IEEE, 2009.

[33] K. Schoeffmann, M. Lux, M. Taschwer, and L. Boeszoermenyi. Visualization of video motion in context of video browsing. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 658–661. IEEE, 2009.

[34] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

[35] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360. ACM, 2007.

[36] T. Shon and J. Moon. A hybrid machine learning approach to network anomaly detection. *Information Sciences*, 177(18):3799–3821, 2007.

[37] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep fisher networks for large-scale image classification. In *Advances in neural information processing systems*, pages 163–171, 2013.

[38] C. Vondrick and D. Ramanan. Video annotation and tracking with active learning. In *Advances in Neural Information Processing Systems*, pages 28–36, 2011.

[39] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[40] H. Wang, H. Su, K. Zheng, S. Sadiq, and X. Zhou. An effectiveness study on trajectory similarity measures. In *Proceedings of the Twenty-Fourth Australasian Database Conference-Volume 137*, pages 13–22. Australian Computer Society, Inc., 2013.

[41] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4305–4314, 2015.

[42] E. J. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 85(411):664–675, 1990.

[43] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. *Computer Vision–ECCV 2008*, pages 650–663, 2008.

[44] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[45] A. Yao, J. Gall, C. Leistner, and L. Van Gool. Interactive object detection. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3242–3249. IEEE, 2012.

[46] B. Z. Yao, B. X. Nie, Z. Liu, and S.-C. Zhu. Animated pose templates for modeling and detecting human actions. *IEEE transactions on pattern analysis and machine intelligence*, 36(3):436–452, 2014.

[47] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang. T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International conference on advances in geographic information systems*, pages 99–108. ACM, 2010.

[48] X. Yuan, W. Lai, T. Mei, X.-S. Hua, X.-Q. Wu, and S. Li. Automatic video genre categorization using hierarchical svm. In *Image Processing, 2006 IEEE International Conference on*, pages 2905–2908. IEEE, 2006.

[49] J. Yuen, B. Russell, C. Liu, and A. Torralba. Labelme video: Building a video database with human annotations. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1451–1458. IEEE, 2009.

[50] J. Yuen and A. Torralba. A data-driven approach for event prediction. *Computer Vision–ECCV 2010*, pages 707–720, 2010.

[51] L. Zelnik-Manor and M. Irani. Event-based video analysis. 2001.

[52] Y. Zheng, L. Wang, R. Zhang, X. Xie, and W.-Y. Ma. Geolife: Managing and understanding your past life over maps. In *Mobile Data Management, 2008. MDM'08. 9th International Conference on*, pages 211–212. IEEE, 2008.

[53] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 28–31. IEEE, 2004.

[54] Z. Zivkovic and F. Van Der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006.

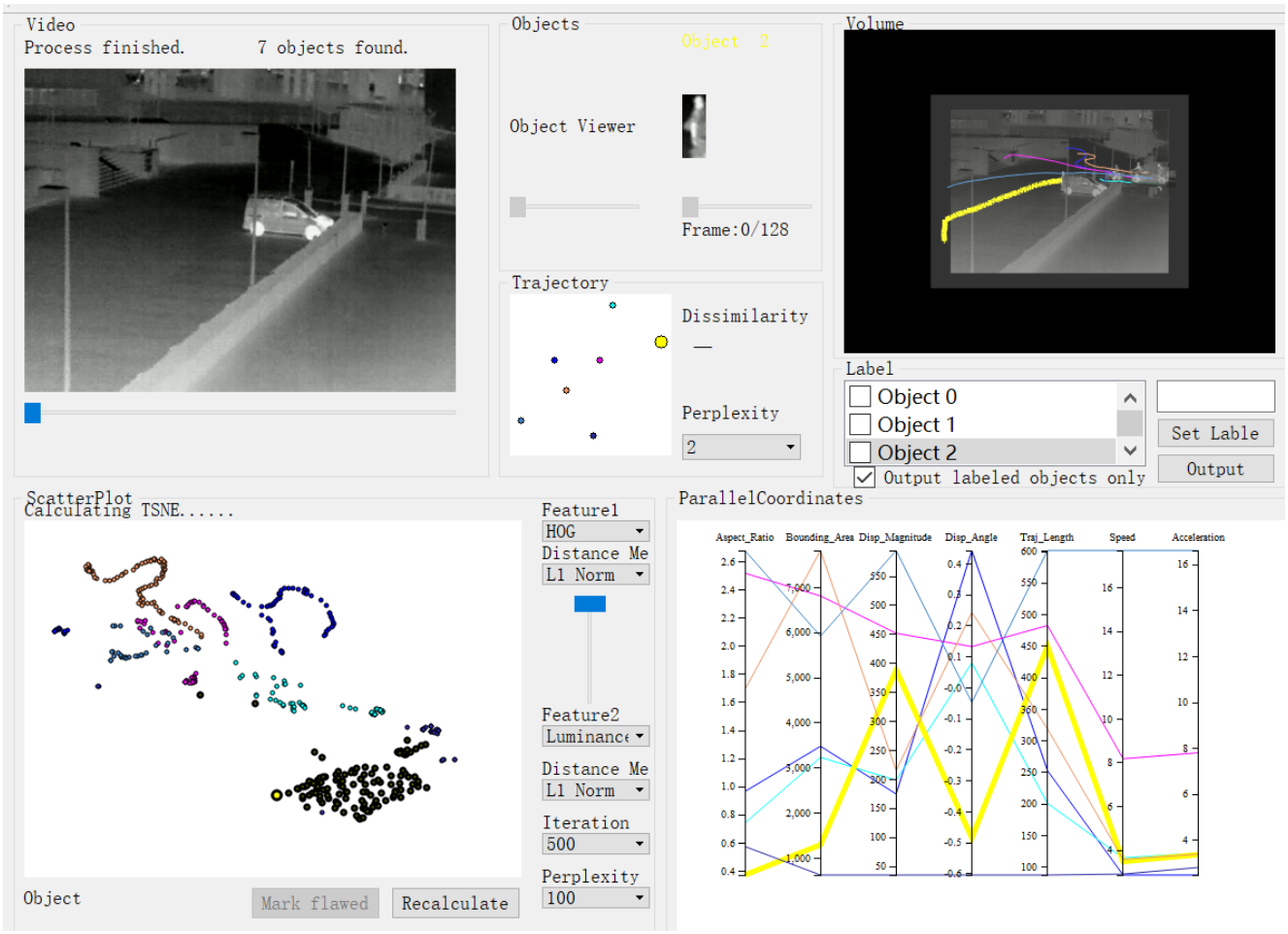

Figure 7. Hover in an instance of object 2 (in yellow) in Image Scatterplot.

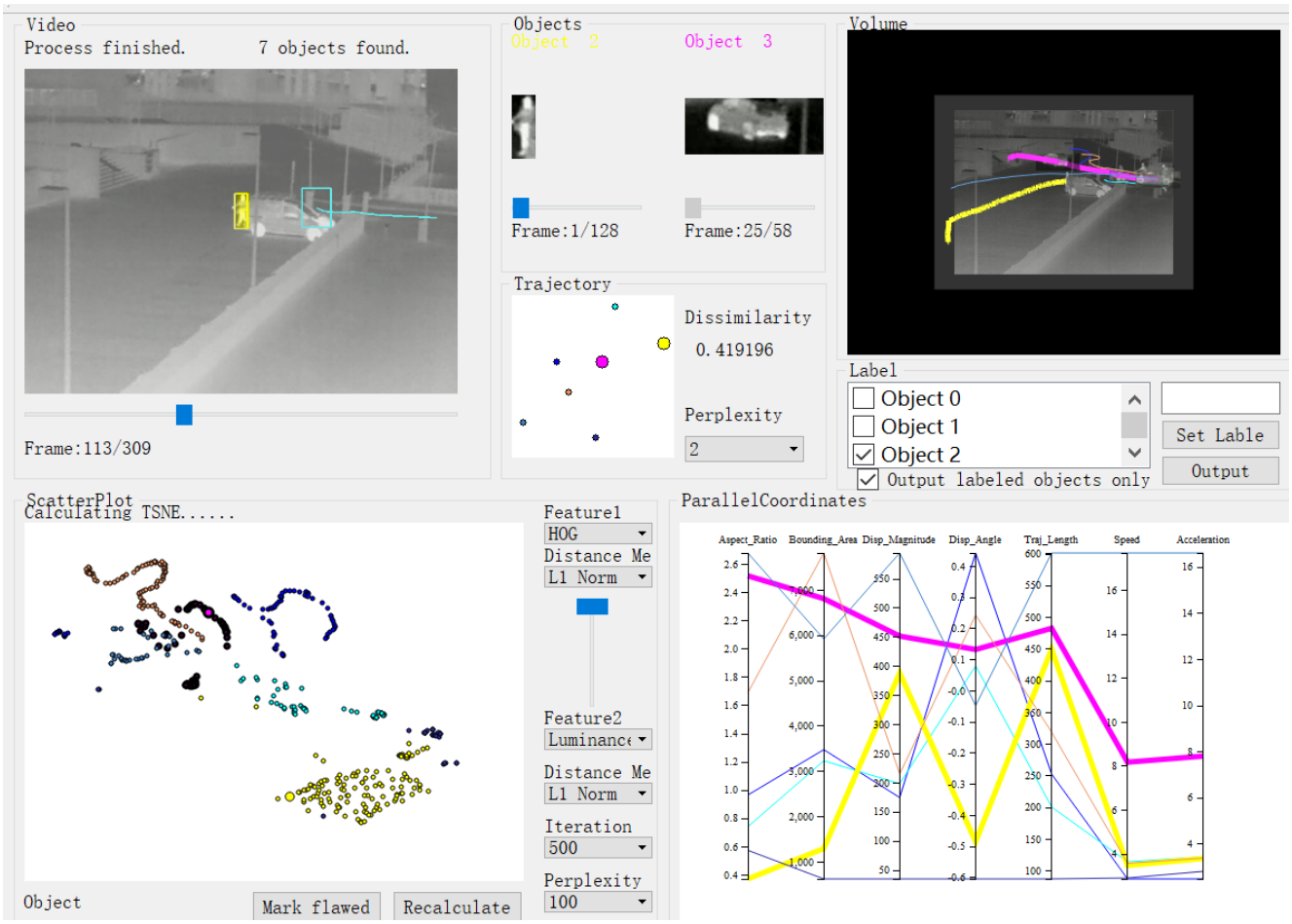

Figure 8. Click an instance of Object 2 (in yellow) in Image Scatterplot. while hovering in Object 3 (in pink).

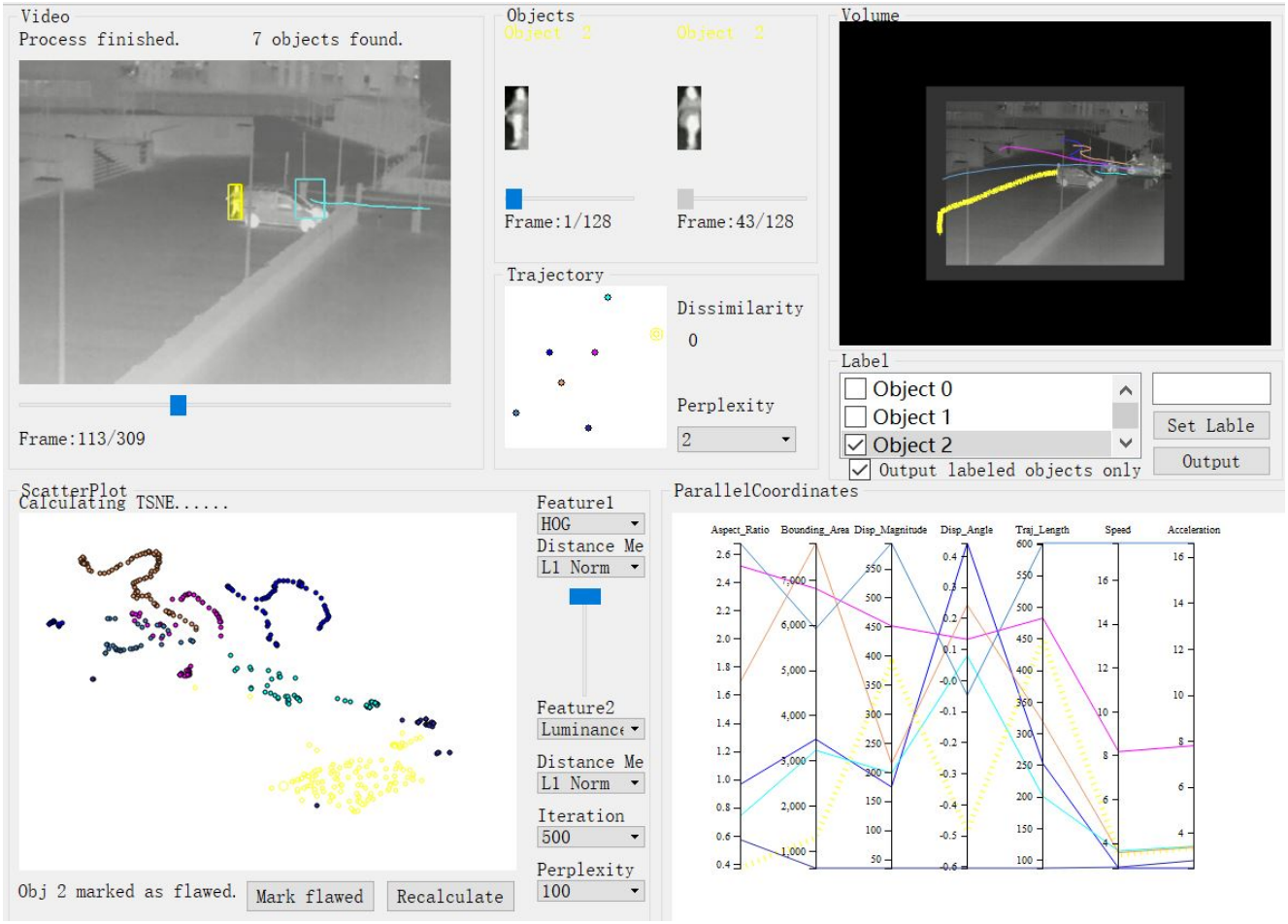

Figure 9. Mark Object 2 (in yellow) as flawed.

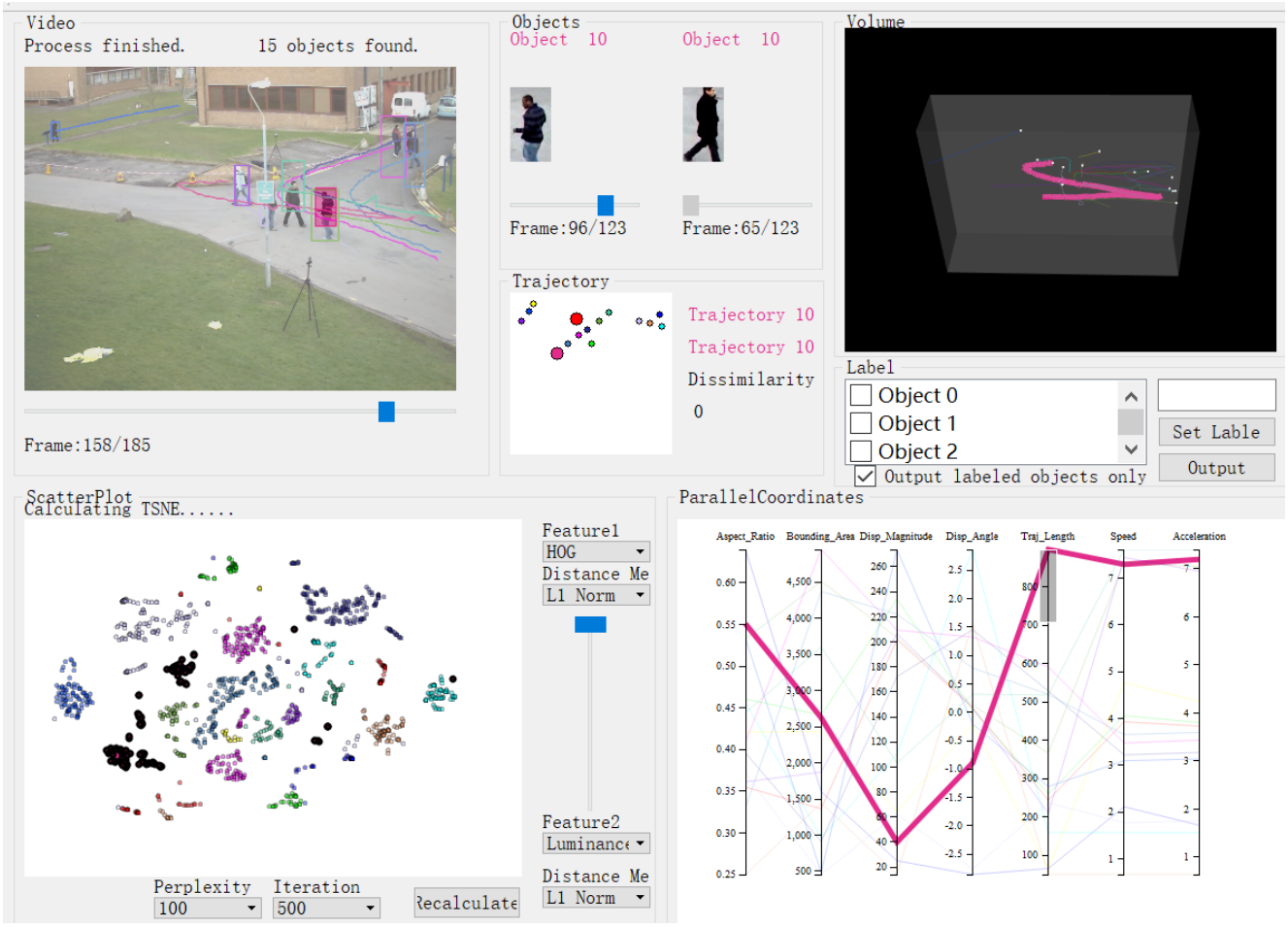

Figure 10. Case 1. Spot flawed result of tracking using Image Scatterplot. Instances in same color scatter into different clusters.

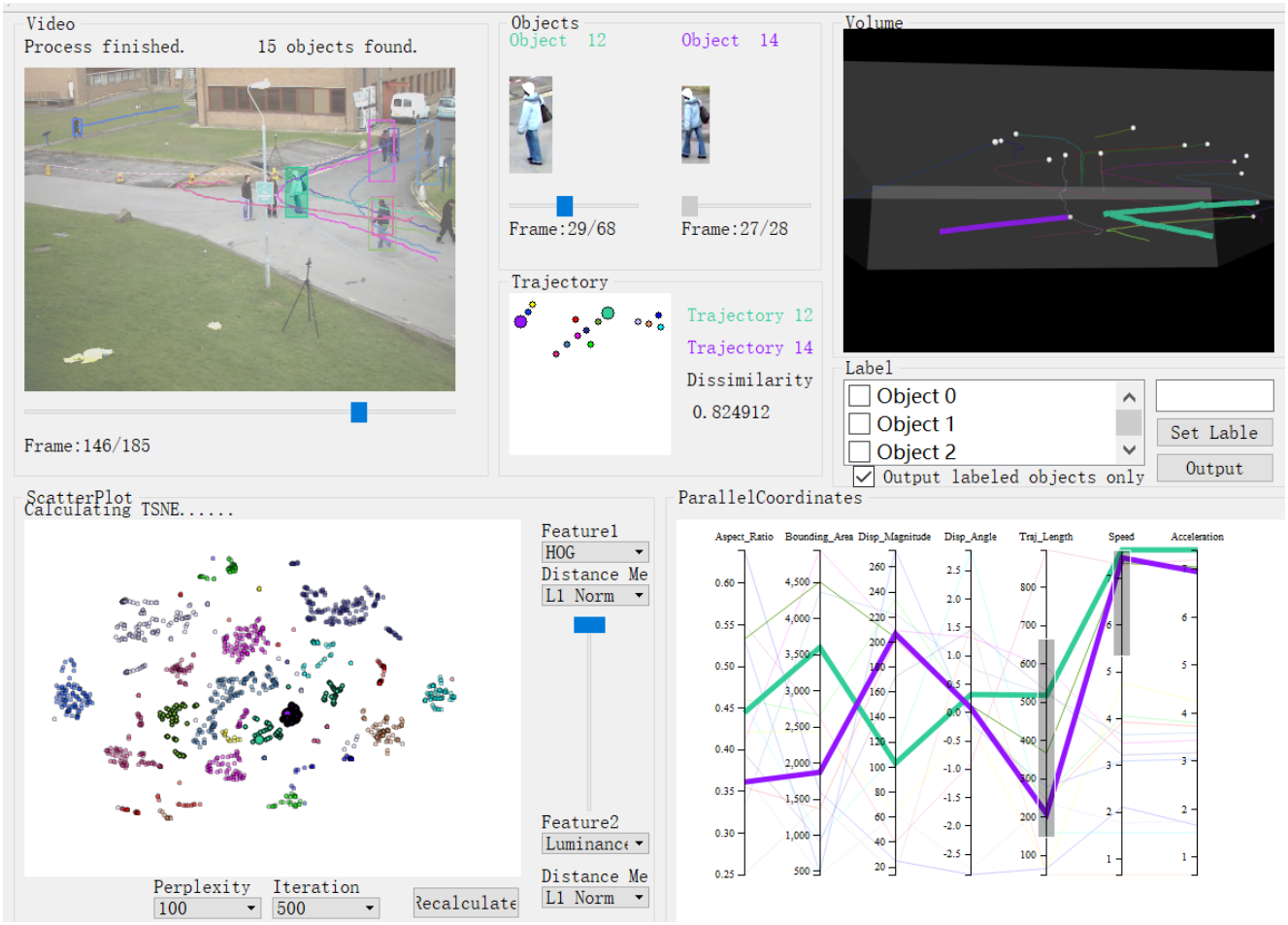

Figure 11. Case 1. Spot flawed result of tracking using Image Scatterplot. Instances in different colors rather close.

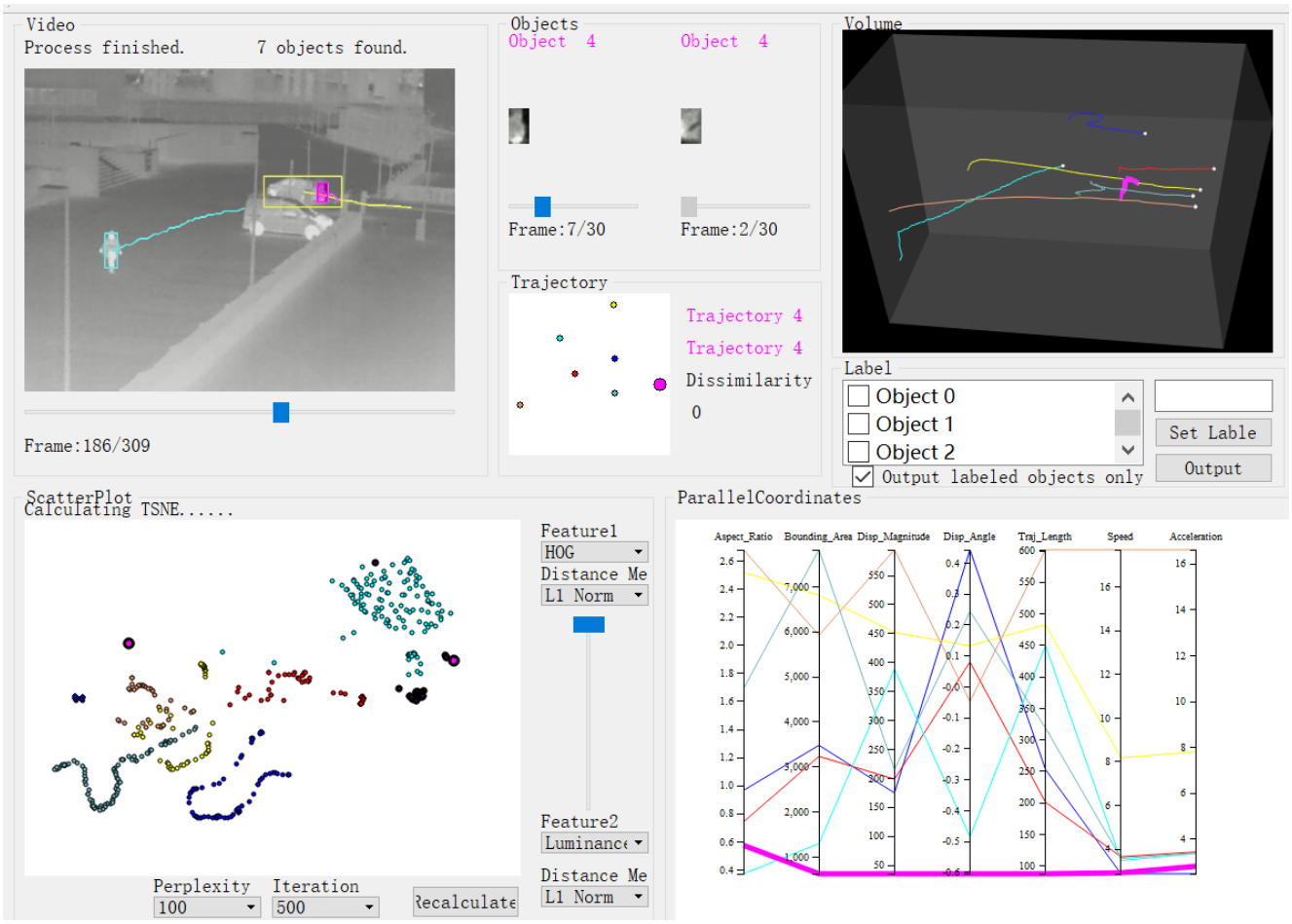

Figure 12. Case 2. Spot flawed result of tracking using Image Scatterplot. Noise tracked as object.

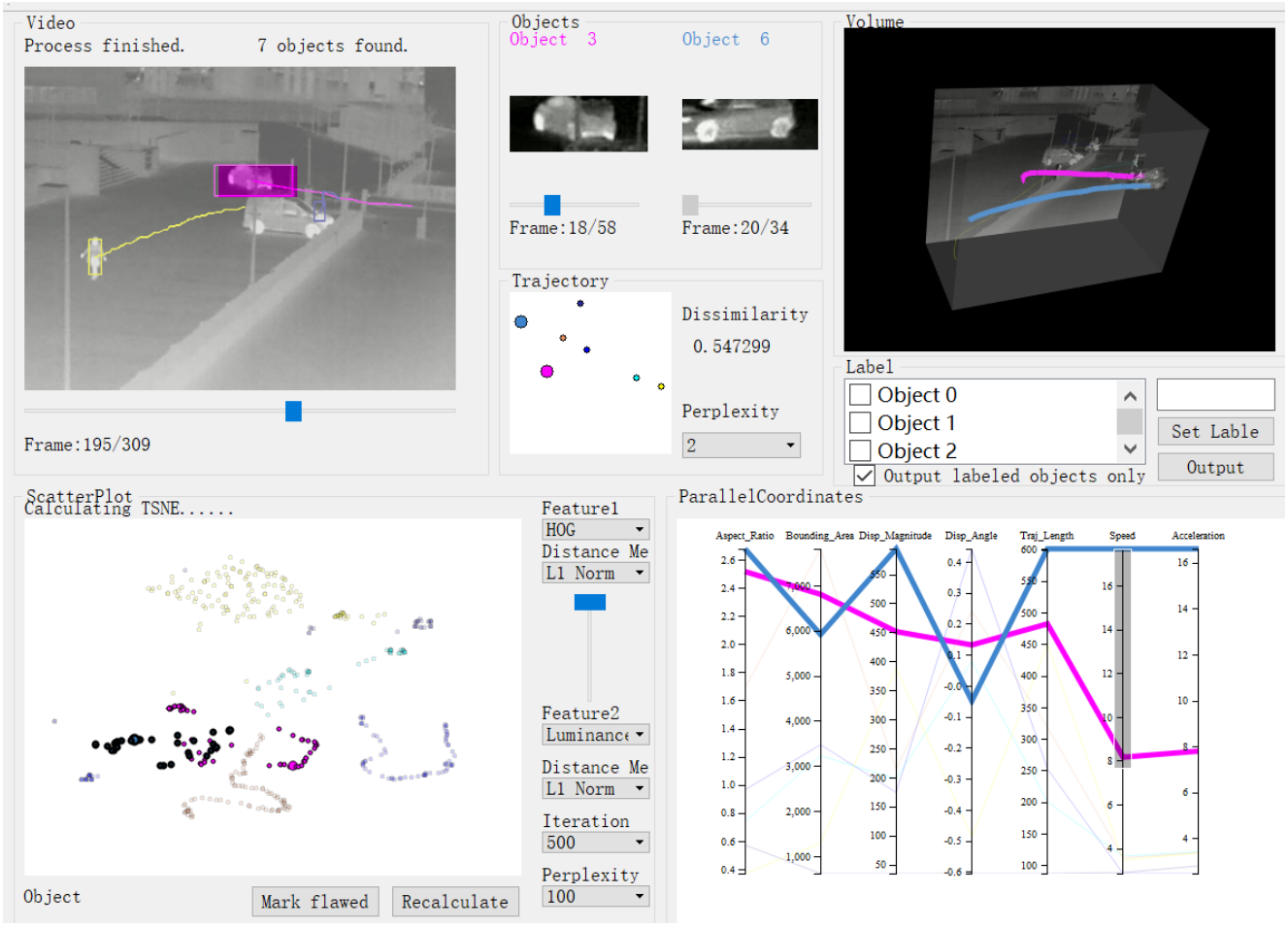

Figure 13. Case 2. Search for a uniform moving car.

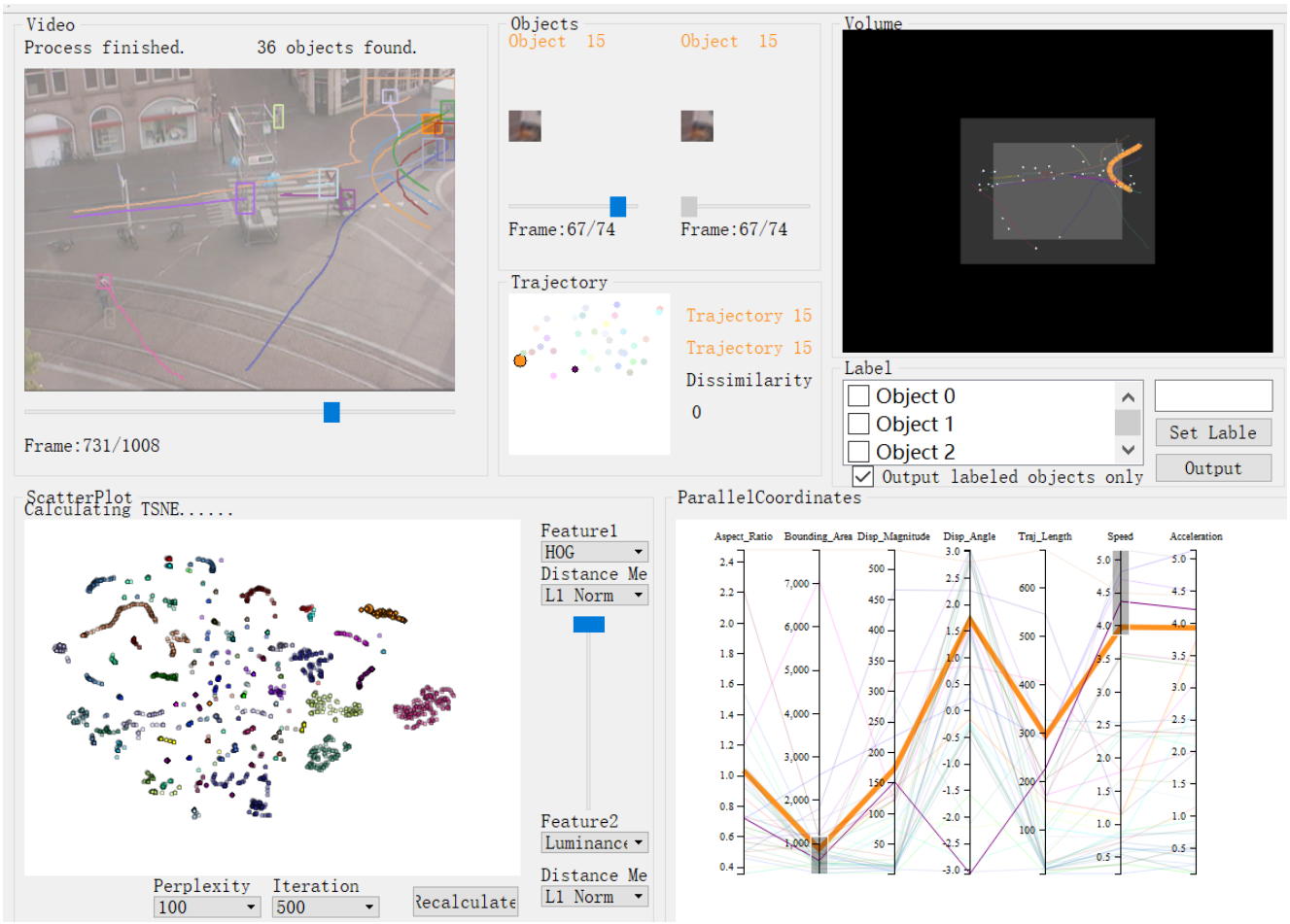

Figure 14. Case 3. Part of an object tracked.

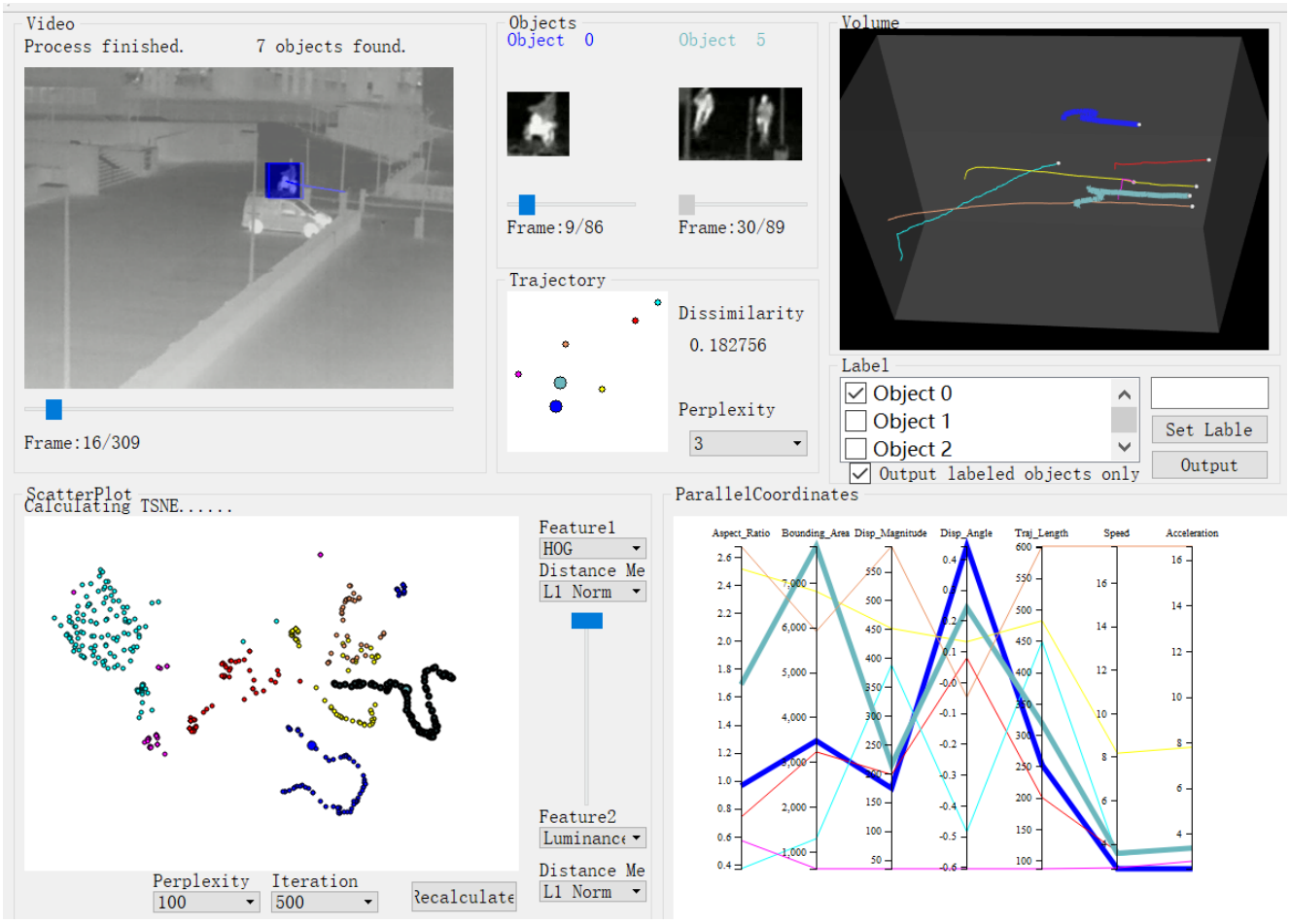

Figure 15. Case 4. Objects with unsimilar images but similar trajectories.