Contextualized Trajectory Parsing with Spatio-temporal Graph

Xiaobai Liu, Liang Lin, and Hai Jin, *Senior Member, IEEE*

Abstract—This work investigates how to automatically parse object trajectories in surveillance videos, that aims to jointly solve three subproblems: i) spatial segmentation, ii) temporal tracking, and iii) object categorization. We present a novel representation spatio-temporal graph (ST-Graph), in which: i) graph nodes express the motion primitives, each representing a short sequence of small-size patches over consecutive images; and ii) every two neighbor nodes are linked with either a positive edge or a negative edge to describe their collaborative or exclusive relationship of belonging to the same object trajectory. Phrasing the trajectory parsing as a graph multi-coloring problem, we propose a unified probabilistic formulation to integrate various types of context knowledge as informative priors. An efficient composite cluster sampling algorithm is employed in search of the optimal solution by exploiting both the collaborative and the exclusive relationships between nodes. The proposed framework is evaluated over challenging videos from public datasets, and results show that it can achieve state-of-the-art tracking accuracy.

Index Terms—Video Analysis, Visual Tracking, Spatio-temporal Graph, Composite Cluster Sampling

1 Introduction

Object tracking has long been an active research topic in computer vision. Recently, due to the increasing demands of industrial applications such as surveillance system, navigation, robotics and sports analysis, significant progress in object tracking have been made in terms of its scalability and reliability. However, tracking multiple objects of interests that move with significant occlusions remains challenging. Figure 1(a) shows a sequence of input images and Figure 1(b) shows the corresponding foreground regions (in black) which are generated by a background subtracting module (Gaussian Mixture Model based method [17] in this example). In the figures, the foreground blobs adhere together although they correlate with different objects of interests. In fact, one object (e.g. car) may enter into the view of camera with partial occlusions and never appear completely. Tracking multiple objects under persistent occlusions requires performing spatial segmentation for every single image, which give rises to the task of simultaneous temporal tracking and spatial segmentation.

- *X. Liu is with SCTS&CGCL, School of Computer Science&Technology, Huazhong University of Science Technology, Wuhan 430074, China, and with the Department of Statistics, University of California, Los Angeles, CA 90034. E-mail: xbliu.lhi@gmail.com.*
- *L. Lin is with Sun Yat-Sen University, Guangzhou 510275, China. E-mail: linliang@ieee.org.*
- *H. Jin is with SCTS&CGCL, School of Computer Science&Technology, Huazhong University of Science Technology, Wuhan 430074, China.*

This work was supported by National Science Foundation (no. 0917141), the Hi-Tech Research and Development (863) Program of China (no.2013AA013801), and the National Natural Science Foundation of China (no. 61173082). Corresponding author is Liang Lin.
Manuscript received 25 Apr. 2012; revised 18 Dec. 2012; accepted 30 Mar. 2013; published online 30 Apr. 2013.
Recommended for acceptance by I. Reid.
For more information on obtaining reprints of this article, please send e-mail to tpami@computer.org, and reference IEEECS Log Number TPAMI-2012-04-0313.
Digital Object identifier no. 10.1109/TPAMI.2013.84.

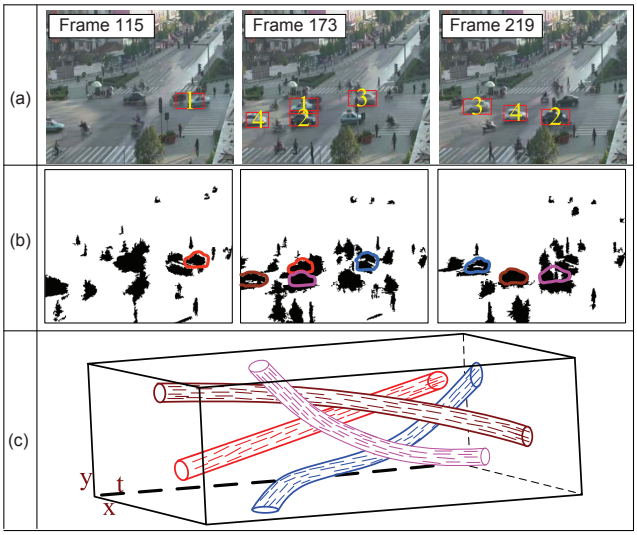

Fig. 1. Trajectory parsing via deferred observations. (a) Three images from a video sequence, and (b) the corresponding foreground masks. Object trajectories are shown in mask images where different colors indicate different objects of interest. (c) The parsed trajectories in perspective view. Each trajectory consists of a bundle of motion primitives.

In the past literature, Bugeau et al. [7] utilized the graph cut method to simultaneously track and segment multiple objects in videos. Zhao et al [45] proposed to segment and track multiple persons in crowded environments by a unified Bayesian model. Yu et al. [43] utilized temporal tracking (based on Gaussian Mixture Models) to assist video foreground/background segmentation. Although great successes obtained, these methods usually perform segmentation and tracking in alternation which may get stuck in a local minimum and requires good initialization to converge. In this work, we present unified framework for robust trajectory parsing.

The target is to automatically parse each input video into a set of object trajectories and obtain their category labels, e.g., sedans, pedestrians, or bicycles. In contrast with the past methods, our method can simultaneously solve three important tasks in video analysis, including foreground (moving) object segmentation, visual object tracking, and object categorization.

Our method starts with partitioning each image of the input video sequence into a set of fixed-size patches. Then, we match the patches over consecutive images to generate a number of cubic cells in the 3D coordinate (the 2D spatial coordinate plus the coordinate of time), called "*motion primitives*". One trajectory usually contains dozens of motion primitives on an observation period (e.g. a window of images) and one motion primitive usually consists of several (at most 8 patches in this work) perceptually similar and spatially smooth patches over consecutive images. Each motion primitive is likely to have one dominant moving direction and all the motion primitives belonging to the same trajectory are well connected with each other. Motion primitives are introduced as the intermediate-level representation of object trajectories. Figure 1(c) intuitively illustrates how motion primitives constitute the object trajectories. Our previous work [27] firstly used the motion primitives and demonstrated its superiority over the traditional representations [42] [3]. In this work, we introduce a new procedure to generate motion primitives that respects both appearance consistency and spatial smoothness.

Taking motion primitives as graph nodes, we link each node to its neighbor nodes in the 3D coordinate with one edge to construct an adjacent graph. One edge can be either positive or negative, indicating the two nodes either cooperatively or conflictingly belong to the same trajectory. We call this resulting adjacent graph as the spatio-temporal graph (*ST-graph*) because it can convey both the spatial and the temporal information. The negative (conflicting) constraints serve as important complements to the positive (cooperative) constraints, both of which should be satisfied with probability during inference. For example, if two motion primitives have significantly different moving directions, they are less likely to belong to the same object trajectory. Similar negative constraints could be obtained based on other cues, e.g. colors and shapes. In this work, we assign one edge to be positive or negative by examining the moving directions of the two motion primitives. One practical procedure for constructing ST-graph will be introduced in Section 2. Thus, the task of trajectory parsing is phrased as a graph multi-coloring (labeling) problem [3].

We present an efficient composite cluster sampling algorithm to infer the optimal coloring solution on ST-Graph by exploiting both the negative and the positive constraints. This algorithm bases on the typical cluster sampling algorithm [3]. The coloring procedure is essentially a series of reversible jumps between different solution states. Given one solution state (or the current coloring of ST-graph) at one step, there are two types of jumps moving to a new solution state. The first one is to generate clusters of graph nodes by turning on/off the edges with certain probabilities, select one cluster of nodes and recolor them (namely assigning these motion primitives to different trajectories) so that all internal constraints are satisfied. Two nodes linked with one positive edge of being "on" should stay the same color whereas two nodes linked with one negative edge of being "on" should be assigned to different colors. The other type of jump is to apply the object categorization method (introduced in later sections) to the graph nodes (motion primitives) with the identical color, which also leads to the change of the current solution state. The above two jumps alternate following the Metropolis Hastings [28] method until convergence.

Moreover, the proposed framework can integrate various scene context knowledge for parsing object trajectories in video surveillance. Particularly, we utilize both scene knowledge [16] and temporal consistency constraints to guide the inference over ST-graph. For example, scene surface annotations (once estimated) can be used for pruning false alarms in tracking, and camera viewpoint parameters can be used for predicting the potential object sizes and shapes by integrating the results of object categorization.

Our proposed algorithm does not require any manual initialization steps and thus can start and track objects of interest in a fully automatic manner.

1.1 Relation to Previous Works

In the literature of trajectory analysis, there are two main streams: **i) sequential inference** based on current observation only, and **ii) deferred inference** based on a period of observations.

The first category of methods usually learns object model from previously observed video sequence, and further applies the obtained model to predict the location of objects in the current image. Exemplar methods include Particle Filtering tracker [19], MeanShift tracker [9] and online boosting trackers [8], [2]. These methods can work well in video scenarios with little ambiguities. However, they need to make decisions immediately and update the learnt model on the fly, which may lead to the open problem of model drift [26]. This problem becomes even worse while there are long-time occlusions or mutual interactions of moving objects in video.

The second category of methods conducts inference based on a set of observations. To optimize, both deterministic inference algorithms, such as dynamic programming [18], multiple hypothesis tracker [31], joint probabilistic data-association filter [30], and stochastic sampling methods, e.g., Gibbs sampler [45] and Data-driven MCMC [42], have been widely studied and utilized for different models. There are strong evidences showing that deferred inference is usually more robust against various challenges than the sequential inference, and could reduce the effect of model drift problem in practice. Our work follows the methodology of deferred inference.

Graph based representation has been widely used in various problems, where both cooperative and conflicting relationships are modeled. In particular, Lin et al. [23] proposed to construct candidacy graph for robust layered graph matching, that can exploit the rich relationships between matching candidates. Porway et al. [29] presented a rich graphical model to generate multiple solutions for probabilistic inference. In contrast, the proposed spatio-temporal graph representation further extends

the rich graph representation to combine multiple different cues within one single probabilistic framework.

Motion primitive can be viewed as a type of tracklet, that depicts one moving object over a set of consecutive images [39], [36], [32]. One motion primitive usually comprises of small-size patches, and one trajectory may include dozens of motion primitives on a certain period (e.g. a window of 30 images), as illustrated in Fig. 1. In contrast, tracklet methods tend to detect the object of interest in every image and collect the detected foreground blobs (of relatively large-size) from a window of images to form one single trajectory segment, i.e. tracklet. This is not a trivial difference, because our method aims to over-segment the foreground regions of each image as the pre-step and then search for the optimal partition during inference. As a result, the spatial partition and the temporal tracking are jointly solved to handle the realistic challenges, e.g., persistent occlusions and object conglutinations. In the literature, other similar works include the method by Kompatsiaris et al. [20] which utilizes the spatio-temporal filters to separate the foreground objects from the background structures, and that by Basharat et al. [4] which proposes to construct motion segments based on the spatial and temporal analysis of interest point correspondence.

The remainder of this paper is arranged as follows. We first introduce the spatio-temporal graph representation in Section 2. In Section 3, we discuss the Bayesian treatment of trajectory parsing, and introduce an efficient inference algorithm which is further extended for multi-object tracking . In Section 4, we define the posterior probability used in Section 3 that integrates both the context prior models and the likelihood models. Last, we report the evaluations with comparisons to other methods in Section 5. We conclude this paper in Section 6.

2 Spatio-Temporal Graph

We first introduce the representation of this work. Let $\mathbf{I} = \{I^1, I^2, ..I^\tau\}$ denote the observed images in video sequence, t indexs the images and τ indicates the number of the recently observed images. Table 1 summarizes the main notations used in this paper. To separate the moving objects of interests from the background, we utilize the background modeling method based on Gaussian Mixture Model [17]. The derived foreground regions in each single image are evenly partitioned into a set of patches of fixed size (e.g. 12×12 pixels). Thus, our goal is to match these patches over consecutive images to generate motion primitives, and further collect them as graph nodes to construct the spatio-temporal graph.

2.1 Motion Primitive

We represent each object trajectory in a video sequence by a bundle of "motion primitive", as illustrated in Fig. 1. Each motion primitive consists of a short sequence of matched patches over consecutive images. Like the super-pixel in 2D image segmentation [25], motion primitives are introduced for dimensionality reduction and efficient inference. Figure 2(a-d) intuitively illustrate the proposed representation in the 3D coordinate and the 2D images.

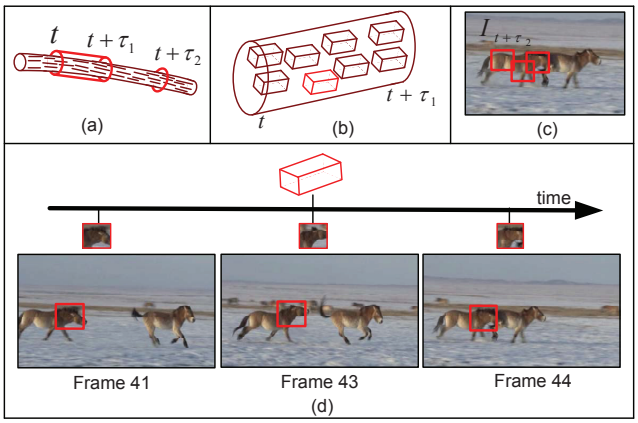

Fig. 2. Trajectory with motion primitives. (a) Illustration of one typical trajectory; (b) Illustration of one cropped trajectory from I^t to $I^{t+\tau_1}$; (c) A cross-section of the trajectory at time $t+\tau_2$ and 3 patches of interest (in red box); (d) Each motion primitive is a set of matched patches over consecutive images, represented as a "cubic cell" in $3D$ coordinate. Notice that in (c) and (d), image patches are depicted larger than its real size.

The ingredient step of constructing motion primitives is to find patch-level correspondences between two consecutive images. For one given image patch in I^t, our goal is to find the most similar patch in the next image I^{t+1}, and meanwhile, to encourage close-by patches to have similar displacements. These objectives are similar with the previous works on optical flow [24] [6] which assume the patch descriptors are constant with respect to the pixel displacement field, and utilize additional regularization terms to impose spatial smoothness. To address the particular problem here, we modify the optical flow assumptions in the following way. First, for one given patch, we restrict the search of the potential matches in the next image to a relatively small neighbor region, which will largely reduces the optimization time. Second, we allow one patch not to match any patches in the next image, accounting for occlusions or scene noises. These two assumptions allow establishing correspondences across images that encourage appearance consistency and spatial smoothness.

Let $\mathbf{x}_u^t$ denote the center location of the u^{th} patch in I^t, and $\mathbf{x}_u^t + \mathbf{d}_u^t$ denote the center location of its match in the image I^{t+1} where $\mathbf{d}_u^t$ is the two dimension displacement vector (the horizontal direction and the vertical direction), $\mathbf{d}_u^t \in [-M..M] \times [-M..M]$ where the constant M determines the size of the search window (i.e. $2M+1$). To describe the image patch at $\mathbf{x}_u^t$, we extract from it a 59-D LBP (local binary pattern) feature [15], a 39-D color histogram (RGB space, 13-D for each channel), and a 64-D histogram of oriented gradient (HOG) [10], normalized by respective summation, and concatenate them to form one single vector, denoted as $F(\mathbf{x}_u^t)$. We also introduce one binary variable $\phi_u^t \in [1, 0]$ for each patch to indicate whether it has a match in the next image. For every two consecutive images I^t and I^{t+1}, the correspondence search is formulated as a discrete optimization problem on the image lattice [24] [6] with the following cost

TABLE 1
Main notations used in this work.

$I = \{I^1, I^2, ..I^\tau\}$	I^t denotes the input image indexed by $t = 1..\tau$; τ is the number of the recently observed images in the video sequence.
u, v, i, j, k, l	u and v index the image patches in one image; i and j index the motion primitives (or graph nodes) collected for $\mathbf{I}$; k and l index the trajectories in $\mathbf{I}$.
$\mathbf{x}_u^t, M, F(\mathbf{x}_u^t), \phi_u^t \in [1, 0], \mathbf{d}_u^t, E(\mathbf{d}_u^t)$	For a patch in I^t, $\mathbf{x}_u^t$ denotes its location; M the size of the search window; $F(\mathbf{x}_u^t)$ the extracted appearance feature; ϕ_u^t the occlusion state variable; $\mathbf{d}_u^t$ the displacement vectors; $E(\mathbf{d}_u^t)$ the energy by the displacement vector $\mathbf{d}_u^t$.
$\mathcal{G} = (\mathbb{V}, \mathbb{E}), \mathbb{E}^+, \mathbb{E}^-, V_i$	$\mathcal{G}$ denotes the proposed spatio-temporal graph (ST-graph); $\mathbb{V}$ is the node set and $\mathbb{E}$ is the edge set; $\mathbb{E}^+$ the positive edge set; $\mathbb{E}^-$ the negative edge set; V_i is the i^{th} node in $\mathbb{V}$.
$e =< i, j >, \rho_e, q_e$	For an edge $e =< i, j >$ that connects V_i and V_j, ρ_e denotes its status variable ("on" or "off"), q_e the edge probability.
Δ_{ij}, κ, c_i	Δ_{ij} denote the cosine similarity between the moving directions of two nodes; κ is the constant threshold of cosine similarity used for determining whether an edge is positive or negative; c_i the color assigned to the node V_i;
$D^{apr}(\cdot, \cdot)$	the Euclidean distance between two appearance feature vectors
$W = \{N, C_{[1..N]}, L_{[1..N]}\}, W'$	W is the solution representation; W and W' are also used to indicate two different solution states during inference; N the number of trajectories; C_k the object trajectory indexed by k; L_k the category label of C_k.
$C_k = \{t_k^b, t_k^d, \Gamma_k, \mathbf{x}_k^b, \mathbf{x}_k^d\}$	t_k^b indicates the index of the image where C_k occurs in video, t_k^d the index of the image where C_k ends; Γ_k the skeleton shape of C_k; $\mathbf{x}_k^b$ the birth position in 2D spatial coordinate (where C_k occurs in the videos), $\mathbf{x}_k^d$ the death position (where C_k ends).
$cls(\mathcal{V}^c), cls(\mathcal{V}^c\|W)$	$cls(\mathcal{V}^c)$ denotes the current colors of the CCCP $\mathcal{V}^c$ (Composite Connected Component), and $cls(\mathcal{V}^c\|W)$ denotes the colors of $\mathcal{V}^c$ at the state W.
$Cut^+(\mathcal{V}^c\|W), Cut^-(\mathcal{V}^c\|W)$	$Cut^+(\mathcal{V}^c\|W)$ ($Cut^-(\mathcal{V}^c\|W)$) denote the set of positive (negative) edges that are turned off probabilistically around $\mathcal{V}^c$ at the state W;
$\Pi_k^t = \{\mathbf{y}_k^t, S_k^t\}, k = 1..N, \Pi_0^t$	Π_k^t denotes the foreground region in image I^t that belongs to the k^{th} trajectory; $\mathbf{y}_k^t$ the center location of Π_k^t in I^t; S_k^t the size of Π_k^t; Π_0^t indicates the regions in I^t which do not belong to any trajectories.
$Dir(\mathbf{h}^r, \alpha^r)$, $Dir(\mathbf{h}^t, \alpha^t)$	denote the Dirichlet prior on the input vector $\mathbf{h}^r$ (histogram of category occurrence frequencies) or $\mathbf{h}^t$ (histogram of trajectory lifespan) given the model parameter α^r or α^t.
$\mathcal{M}^f(\mathbf{y}_k^t\|L_k)$, $\mathcal{M}^a(\mathbf{y}_k^t\|L_k)$	$\mathcal{M}^f(\mathbf{y}_k^t\|L_k)$ returns the normalized occurrence frequency of the object category L_k at the position $\mathbf{y}_k^t$; $\mathcal{M}^a(\mathbf{y}_k^t\|L_k)$ returns the predicated area for L_k at $\mathbf{y}_k^t$ in images.
$\mathcal{M}^o(\mathbf{x}_k^b)$, $\mathcal{M}^p$	$\mathcal{M}^o(\mathbf{x}_k^b)$ returns the normalized frequencies of the death/birth of trajectories at position $\mathbf{x}_k^b$; $\mathcal{M}^p$ denotes a set of trajectories.
$D^{ske}(\Gamma_k, \Gamma_l^p), D^{geo}(\Gamma_k, \Gamma_l^p)$	Γ_k denotes the skeleton of the k^{th} trajectory in W; Γ_l^p the l^{th} skeleton in $\mathcal{M}^p$; $D^{ske}(\cdot, \cdot)$ returns the similarity distance between two skeletons; $D^{geo}(\cdot, \cdot)$ the geometric distance between two skeletons.
$\mathbb{S}(\Pi_k^t, L_k)$	returns the categorization confidence for Π_k^t of belonging to the category L_k, obtained by the categorization method [10].
$z_1, .., z_4$	constant parameters that control the weights of different energy terms of the posterior

function:

$$\min_{\{\mathbf{d}_u^t\}, \{\phi_u^t\}} \sum_u \begin{cases} \gamma, & \phi_u^t = 0; \\ E(\mathbf{d}_u^t), & \phi_u^t = 1. \end{cases} \tag{1}$$

with

$$E(\mathbf{d}_u^t) = \|F(\mathbf{x}_u^t) - F(\mathbf{x}_u^t + \mathbf{d}_u^t)\|^2 + \frac{1}{\sigma_1}\|\mathbf{d}_u^t\|^2 + \sum_{v \in \epsilon_u^t, \phi_v^t = 1} \min(\|\mathbf{d}_u^t - \mathbf{d}_v^t\|^2, T) \tag{2}$$

where $\|\|$ is the Forbenius norm of a vector, and ϵ_u^t indicates the spatial neighborhood of the patch at $\mathbf{x}_u^t$ (4-neighborhood structure is used). γ is a relatively small constant, accounting for the penalty of occlusions or large appearance variations. We set $\gamma = 0.05, \sigma_1 = 300, T = 4$ in this work. $E(\mathbf{d}_u^t)$ defines the energy to minimize for the patch $\mathbf{x}_u^t$, in which the ℓ_2 norm term is employed in the first two terms to account for feature matching and displacement respectively, and a ℓ_2 norm is used in the third term to model the spatial smoothness in the patch displacement field. The threshold T is introduced to allow for certain amount of discontinuities.

We adopt the efficient belief propagation (BP) method [12] to optimize Eq. (1), and the algorithm complexity is $O(M^2)$ if using the distance transform method [24]. We set $M = 2$ which works well in practice. The whole procedure for every two images, in the C++ platform, will converge in 0.2 second on a workstation (P-IV 2.2GHz CPU, 8GB RAM) without any code optimization.

Given the solved correspondences, we begin from the first image to group the matched patches sequentially. One image patch can be used for at most one motion primitive. To collect one motion primitive, we select one image patch as the seed and grow it to the consecutive images. This grouping proceeds until one of the following conditions is satisfied: i) there are no matched patches in the next image; ii) the length of the current group is larger than a threshold (fixed to be 8 patches in this work); iii) the cosine similarity between the displacement vectors of the firstly collected patch and the newly collected patch is larger than a threshold (fixed to be 0.2 in this work). Each group of patches that contains at least 3 matched patches is considered as one single motion primitive. The third condition is used to ensure a motion primitive has one dominant moving direction. Although we collect image patches sequentially to generate motion primitives, our method can work well in practice due to its following characterizes.

First, we densely collect patches to construct as many motion primitives and preserve the uncertainties for the later stage of inference. Second, the patch-level correspondences are solved to optimal. Last, the above three conditions ensure the generated motion primitives can well depict the object trajectories (in terms of appearance and motion).

2.2 Spatio-temporal Graph Representation

To represent the observed input video sequence, we propose to construct a **spatio-temporal graph** $\mathcal{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ indicates the set of graph nodes each representing one motion primitive, and $\mathbb{E}$ indicates the set of edges between nodes. Table 1 summarizes the major notations used in this paper.

The graph structure $\mathcal{G}$ is a 6-neighbor system, namely each node has 6 edges connecting to its neighbor nodes. To determine the neighbors for a node V_i in $\mathcal{G}$, we use a simple method as follows. First, we arrange all other nodes in $\mathbb{V}$ in a descending list according to their spatial distances (number of pixels) to V_i , denoted as R^1, arrange them in another descending list according to their temporal distances (number of frames) to V_i, denoted as R^2. Herein, every motion primitive is considered as a set of image patches. The spatial distance (or temporal distance) between two motion primitives are calculated as the minimum geometric distance (or the minimum difference of frame indices) between the two sets of image patches. Second, for each point $m = 1, 2, 3, ...$ of the ranked list R^1, append the m^{th} node in R^1 as the neighbor of V_i if it also occurs in the top m positions in R^2. We repeat this procedure until 6 neighbor nodes of V_i are found. Since the extracted motion primitives are usually densely distributed in the 3D coordinate, the neighbor system is usually fully connected. Once represented by the graph $\mathcal{G}$, the trajectory parsing problem is posed as an optimization problem that computes the most probable colorings with a posterior probability. The above ST-graph can convey both spatial and temporal structure between nodes. Our goal is to group these nodes so that both the spatial partitions on individual images and the temporal matching over consecutive images are solved to optimal.

2.3 Positive and Negative Edges

For every two neighbor nodes V_i and V_j in ST-graph, indexed by i and j, we link them with an edge $e =< i, j >$ for either a negative (conflicting) or positive (cooperative) relationship. A positive edge represents a cooperative constraint for two nodes having the same color in the graph. A negative edge, in contrast, requires the two nodes to have different colors, and thus describes a competitive or conflicting constraint. Thus, the edge set $\mathbb{E}$ contains two disjoint subsets, $\mathbb{E} = \mathbb{E}^+ \bigcup \mathbb{E}^-$. $\mathbb{E}^+$ is the set of positive edges and $\mathbb{E}^-$ is the set of negative edges.

We assign each edge to be positive or negative by examining the moving directions of the two nodes. The moving direction of one motion primitive is obtained by averaging over the displacement vectors of the contained patches (see Eq. (1)). For two motion primitives V_i and V_j, we calculate the cosine similarity between their moving directions, denoted as Δ_{ij}. If $\Delta_{ij} <= \kappa$ the edge is labeled to be negative; otherwise the edge be positive. κ is set to be relatively small so that two motion primitives that have significantly different moving directions will be assigned to different colors with high probability during inference. We set $\kappa = -0.2$ in this work.

These edges are turned on and off probabilistically to group nodes into clusters of nodes in a dynamic way so that nodes in every cluster are strongly coupled [3]. On each positive or negative edge, we define an edge probability for the coupling strength. That is, at each edge $e =< i, j >$, we define an auxiliary variable $\rho_e \in \{$"on","off"$\}$ that follows an independent edge probability q_e. At the present state of the ST-graph, for a positive edge $e =< i, j > \in \mathbb{E}^+$, if the two nodes have the same color, i.e., $c_i = c_j$, then the edge e is turned on with the probability q_e; if $c_i \neq c_j$, e is turned off deterministically (with probability 1).

On the other hand, for a negative edge $e =< i, j > \in \mathbb{E}^-$, if the two nodes have the same color $c_i = c_j$, e is turned off deterministically; otherwise, e is turned on with probability q_e to enforce the two nodes stay in different colors.

We define q_e as a statistical probability proportional to how perceptually compatible those two motion primitives are. Let $V_{i,u}$ denote the u^{th} patch of the i^{th} motion primitive, $\mathbf{x}(V_{i,u})$ the center location of $V_{i,u}$ in image. Let $F(\mathbf{x}(V_{i,u})) = F(V_{i,u})$ denote the appearance feature extracted for $V_{i,u}$ (see last subsection), $D^{apr}(\cdot, \cdot)$ denote the Euclidean distance of two appearance features, $|V_i|$ indicate the number of patches contained in V_i. We have,

$$q_e = \exp\left\{ -\frac{1}{\sigma_2 \times |V_i|} \sum_{u=1}^{|V_i|} \min_{v \in [1..|V_j|]} D^{apr}\left[F(V_{i,u}), F(V_{j,v})\right] \right\} \quad (3)$$

where σ_2 is the constant parameter (fixed to be 2). Thus, we define the edge probability using local appearance features. The edge probability is used to describe how strongly two motion primitives are coupled during inference. For a positive edge $e \in \mathbb{E}^+$, if the two nodes are perceptually similar (i.e. have similar appearance features), q_e should have a high value to ensure the two nodes remain the same color with high probability. On the other hand, if the two motion primitives are linked with one negative edge $e \in \mathbb{E}^-$ and they are similar in appearance, there is large ambiguity between these two nodes and q_e should be high to ensure the inference algorithm can exploit this informative conflicting constraint. Therefore, we measure q_e by the the same form in Eq. (3) for both positive edges and negative edges.

One major step in each iteration of our method is to sample ρ_e for each e independently following the edge probability q_e. Afterwards, the set of positive edges that remain "on" form several connected components (CCPs), in each of which every node is reachable from other nodes by the positive edges being "on". The set of negative edges that remain "on" form several Composite CCPs (CCCPs) in each of which every CCP is reachable from other CCPs by the negative edges being "on". Thus one CCCP is a set of isolated CCPs that are connected by negative edges. An isolated CCP is also treated as a CCCP. Different motion primitives in the same CCP will receive the

same color, whereas adjacent CCPs in the same CCCP will receive different colors.

The proposed ST-graph integrates both positive edges and negative edges, which is different from the commonly used adjacent graph representations that utilize positive edges only [42] [3]. Both positive and negative edges can impose soft constraints that should be satisfied during inference as shown in next section. To construct a ST-graph, there are two major steps: calculating the edge probability q_e in Eq. (3), and classifying edge types. One may employ multiple cues (e.g. color, shape, texture) to classify edge types. For example, we could examine multiple cues sequentially to generate for each cue a set of negative edges and assemble these edges to form $\mathbb{E}^-$. The edges not labeled to be negative in any exams form the positive edge set $\mathbb{E}^+$. Since the edges will be probabilistically turned on/off during inference, these different cues will be combined adaptively. In contrast, the existing feature fusion methods usually combine cues by linear voting schema [44] where voting weights are fixed during inference/testing. Although promising, it goes beyond the research scope of this paper to investigate how to combine multiple cues. To address the trajectory parsing problem, we simply examine the moving directions of motion primitives to classify edge types. We will show in experiments that the proposed ST-graph can significantly boost the system performance in comparisons to the typical adjacent graph representation [42].

3 Trajectory Parsing

Once the ST-graph constructed, we can formulate the problem of trajectory parsing as a graph multi-coloring task and define the following solution representation W,

$$W = \{N, C_{[0..N]}, L_{[1..N]}\}, \tag{4}$$

with

$$C_k = \{t_k^b, t_k^d, \mathbf{x}_k^b, \mathbf{x}_k^d, \Gamma_k\}, \quad k = [1..N] \tag{5}$$

where C_k denotes the k^{th} object trajectory consisting of a set of motion primitives, N is the total number of trajectories, and $L_k \in \{''sedan'', ''pedestrian'', ''bicycle''\}$ is the category label of C_k. Each trajectory is parameterized by the following variables: t_k^b indicates the index of the image where C_k occurs in video, t_k^d indicates the index of the image where C_k ends (i.e. left the view of camera), $\mathbf{x}_k^b$ denotes the birth position in 2D spatial coordinate (where C_k occurs in the videos), $\mathbf{x}_k^d$ denotes the death position (where C_k ends), and Γ_k denotes the skeleton shape of C_k. Γ_k is a curve consisting of a series of center positions (in 2D spatial coordinate) of motion primitives that belong to the same trajectory. C_0 represents the foreground blobs not contained in any trajectories. For ease of presentation, we use the notation system that assumes one trajectory C_k appears in every image between t_k^b and t_k^d. However, in practice, we allow one object trajectory C_k not associating with any foreground regions in $I^t, t \in [t_k^b..t_k^d]$ to account for occlusions.

Algorithm 1 . Procedure for Composite Cluster Sampling on ST-graph

1: **Input:** ST-graph $\mathcal{G} =< \mathbb{V}, \mathbb{E} >, \mathbb{E} = \mathbb{E}^+ \bigcup \mathbb{E}^-$; current solution state;
2: For each $e =< i, j > \in \mathbb{E}^+$
   - if $c_i = c_j$ then set $\rho_e =$"on" with probability q_e;
   - else set $\rho_e =$"off";
3: For each $e =< i, j > \in \mathbb{E}^-$
   - if $c_i \neq c_j$ then set $\rho_e =$"on" with probability q_e;
   - else set $\rho_e =$"off";
4: Collect CCPs based on positive edges of being "on";
5: Collect CCCPs based on negative edges of being "on";
6: Select one CCCP and assign colors to its CCPs to change the solution state;
7: Accept the new solution state with the acceptance probability;
8: Go to Step 2, until convergence;

Thus, we can solve the problem of trajectory parsing by maximizing a posterior (MAP) probability in the Bayesian framework,

$$\begin{aligned} W^* &= \arg\max_W P(W|\mathbf{I}; \beta, \theta) \\ &= \arg\max_W P(\mathbf{I}|W; \beta) P(W; \theta), \end{aligned} \tag{6}$$

where β and θ are the parameters for the likelihood and prior models respectively. The likelihood model and prior models are defined in Section 4.

In this section, we first present an efficient composite cluster sampling algorithm to address the multi-label graph coloring problem, and then introduce the categorization method used in this work to obtain L_i. Last, we extend the inference algorithm for multi-object tracking in video sequence.

3.1 Composite Cluster Sampling

Cluster sampling algorithm is first proposed by Swendson and Wang [33] (SW) and refined by Edwards and Sokal [11] for simulating Ising/Potts models in physics. It works iteratively following the MCMC design. At each single step, it flips the colors of multiple nodes, called a "cluster" or a connected component in the Ising Potts model. In contrast with the single-site samplers, e.g., Gibbs sampler [14], that only flip the color of one single node at each step, SW method can move much more efficiently in the solution space. SW was extended to general posterior probabilities in computer vision by Barbu and Zhu [3], called Swendson-Wang Cut (SW-Cut), which however only considers the cooperative relationships between graph nodes. In this work, we further extend the SW-cut method to explore both the cooperative and the conflicting relationships in ST-graph.

Algorithm 1 summarizes the whole composite sampling algorithm. In each iteration, it generates CCCPs, selects one CCCP, and reassign labels to its CCPs such that all internal constraints are satisfied.

The recoloring procedure is actually a MCMC jump that drives the current solution state W to a new solution state,

Algorithm 2 Procedure for Trajectory Parsing.

1: **Input:** currently observed images $\{\mathbf{I}\}$ in the video sequence;
Output: parsed object trajectories and their category labels;
2: Extract motion primitives as graph nodes $\mathbb{V}$;
3: Link every two neighbor motion primitives with either a positive edge or a negative edge to construct the ST-graph $\mathcal{G} =< \mathbb{V}, \mathbb{E} >$;
4: Calculate edge probability q_e, $\forall e \in \mathbb{E}^+ \bigcup \mathbb{E}^-$;
5: Initialize solution state;
6: Iterate until convergence,
- Call Algorithm 1;
- Call the recognition algorithm [10] to classify each obtained CCP;

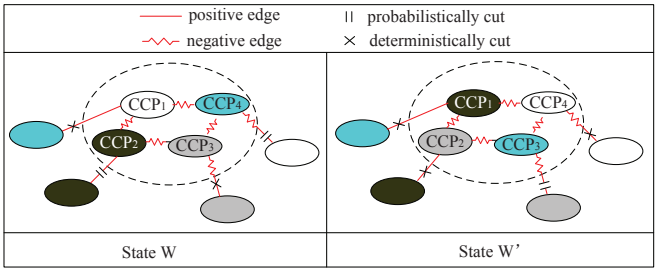

Fig. 3. Two typical solution states during coloring ST-Graph. The edges will be turned "off" either probabilistically or deterministically. The solid ellipses indicate the CCPs and the dashed ellipses indicate the CCCPs. Edges between two CCPs are depicted as one single edge.

denoted as W'. Figure 3 illustrates two solution states W and W' during inference. Each solid ellipse indicates one CCP. Three CCPs connected by the negative edges form a CCCP, depicted by the dashed ellipse. The symbol $\|$ indicates the edge that are turned "off" probabilistically, and the cross in black indicates the edge that are turned "off" deterministically.

We implement the above jump between W and W' according to the Metropolis-Hastings [28] method, which accepts the jump of solution state with an acceptance probability. Let $\mathcal{V}^c$ denote the selected CCCP, $Q\ (\ W \rightarrow W'\ )$ denote the proposal probability of moving from state W to state W', and $Q(W' \rightarrow W)$ denote the proposal probability from W' to W. The acceptance probability of the new state W' is defined based on the proposal probability and the posterior probability,

$$\min\left(1, \frac{Q(W' \rightarrow W)P(W'|\mathbf{I})}{Q(W \rightarrow W')P(W|\mathbf{I})}\right) \tag{7}$$

The proposal probability $Q(W \rightarrow W')$ includes two parts: (i) the probability of generating $\mathcal{V}^c$ at state W, denoted as $Q(\mathcal{V}^c|W)$, and (ii) the probability of recoloring $\mathcal{V}^c$ that moves W to W', denoted as $Q(cls(\mathcal{V}^c) = cls(\mathcal{V}^c|W')|\mathcal{V}^c, W)$. Herein, $cls(\mathcal{V}^c)$ denotes the new colors of $\mathcal{V}^c$, $cls(\mathcal{V}^c|W)$ and $cls(\mathcal{V}^c|W')$ denotes the colors of $\mathcal{V}^c$ at the states W and W' respectively. Therefore, we have the proposal probability ratio defined as follows,

$$\frac{Q(W'\rightarrow W)}{Q(W\rightarrow W')}=\frac{Q(\mathcal{V}^c|W')}{Q(\mathcal{V}^c|W)}\frac{Q(cls(\mathcal{V}^c)=cls(\mathcal{V}^c|W)|\mathcal{V}^c,W')}{Q(cls(\mathcal{V}^c)=cls(\mathcal{V}^c|W')|\mathcal{V}^c,W)}, \tag{8}$$

Given the chosen CCCP $\mathcal{V}^c$ in both states, the assignment of new colors is independent of the surrounding neighbors of $\mathcal{V}^c$ and is often assigned by uniform among all valid assignments. Thus they cancel out and we have $\frac{Q(cls(\mathcal{V}^c)=cls(\mathcal{V}^c|W)|\mathcal{V}^c,W')}{Q(cls(\mathcal{V}^c)=cls(\mathcal{V}^c|W')|\mathcal{V}^c,W)} = 1$.

We further assume $\mathcal{V}^c$ is selected with the uniform probability from all possible CCCPs, and simplify the proposal probability ratio for selecting $\mathcal{V}^c$ at states W and W' as:

$$\frac{Q(\mathcal{V}^c|W')}{Q(\mathcal{V}^c|W)}=\frac{\prod_{e\in Cut(\mathcal{V}^c|W')}(1-q_e)}{\prod_{e\in Cut(\mathcal{V}^c|W)}(1-q_e)}, \tag{9}$$

where $Cut(\mathcal{V}^c|W) = Cut^+(\mathcal{V}^c|W) \bigcup Cut^-(\mathcal{V}^c|W)$. $Cut^+(\mathcal{V}^c|W)$ denotes the set of positive edges that are turned off probabilistically around $\mathcal{V}^c$ at the state W,

$$Cut^+(\mathcal{V}^c|W) = \{e =< i,j >\in \mathbb{E}^+ : V_i \in \mathcal{V}^c, V_j \notin \mathcal{V}^c, c_i = c_j \text{ at } W\}. \tag{10}$$

$Cut^-(\mathcal{V}^c|W)$ denotes the set of negative edges that are turned off probabilistically around $\mathcal{V}^c$ at the state W,

$$Cut^-(\mathcal{V}^c|W) = \{e =< i,j >\in \mathbb{E}^- : V_i \in \mathcal{V}^c, V_j \notin \mathcal{V}^c, c_i \neq c_j \text{ at } W\}. \tag{11}$$

$Cut(\mathcal{V}^c|W')$ has the similar definition as $Cut(\mathcal{V}^c|W)$. The derivation of the above simplification is directly from the previous work in [3] which considers the positive edges only.

In summary, there are two major steps in the proposed inference method: 1) generate a set of CCCPs by turning on/off the edges in $\mathbb{E}$, either probabilistically or deterministically; 2) randomly select one of the formed CCCPs, and recolor its nodes to drive the solution from one state to the other state, while preserving both the positive and negative constraints. The move of states will be accepted with the probability defined in Eq. (7). We conduct the above iterative procedure until convergence or the reach of the maximum iteration numbers (fixed experimentally).

3.2 Object Categorization

During inference, we utilize certain categorization method to classify every obtained CCP. For the CCPs with the identical color (i.e. belonging to the same trajectory), we accumulate the categorization results and assign them to the category label that achieves the highest accumulated confidence. This step of object categorization serves as another MCMC jump which moves the current solution state to a new one. In this way, the object recognition task is integrated within our unified framework for trajectory parsing.

We choose to use the SVM based method in [10] in this work, which extracts the histograms of oriented gradients (HOGs) as the appearance features, and extracts the histograms of oriented optical-flow as the motion features, to describe the object of interests in videos. Results of categorization are also used for estimating various types of context priors which shall

be introduced in Section 4. Integrating the scene geometry knowledge and the predicted object category can provide strong prior about the possible object locations and sizes in the image. As an independent module, other recognition algorithms can be alternately integrated into our framework.

3.3 Procedure of Trajectory Parsing

Algorithm 2 summarizes the proposed trajectory parsing procedure. we adopt the sliding window method [40] [2] to handle the realtime video sequence. In particular, for a window of observed images, we call Algorithm 2 to parse the trajectories and then move to the next window (with a fixed step, e.g. 15 images). Only the graph nodes falling inside the current window can be recolored to change the solution state, i.e. the ST-graph is constructed from the current window of images. The parsed results from the previous windows (the number of windows is up to the storage limitation) are used for initializing the ST-graph of the current observation window, and calculating the posterior probability which shall be discussed in Section 4. This sliding-window strategy, however, will lead to the risk of model-drift that roosts in the ill-posed nature of tracking[26]. However, our method works well in practice due to the following characterizations: 1) there are significant overlappings between two consecutive windows, which experimentally improves the system robustness; and 2) our method works on a batch of deferred images, rather than one single image, which has shown to be successful in the past literature.

4 Bayesian Formulation

This section introduces the definition of the posterior probability in Eq. (6) used in Section 3. These probabilities are calculated from all the observations, instead of the current observation window. We first introduce the notations used in this section. Let Π_k^t denote the foreground region in I^t that belongs to the k^{th} trajectory (Π_k^t may correlate to more than one motion primitives), $k \in [1..N]$. Π_0^t indicates the regions in I^t that do not belong to any trajectories. Π_k^t is parameterized to be $\{\mathbf{y}_k^t, S_k^t\}$ where $\mathbf{y}_k^t$ denotes its center location in I^t and S_k^t its region size.

4.1 Prior Model

We impose three types of priors for parsing trajectories in surveillance videos, including the recognition prior, the scene context prior and the temporal prior.

4.1.1 Recognition Prior

We utilize a mixture of multi-nominal distributions on the category occurrence frequencies to build recognition prior. Let $\mathbf{h}^r$ denote the histogram of category occurrence frequencies in the currently observed images, and α^r denote the corresponding histogram pooled from the training data. Each bin of $\mathbf{h}^r$ or α^r represents the occurrence frequency of one object category in videos. Both $\mathbf{h}^r$ and α^r are normalized to be unit one. Thus, the recognition prior has the following form,

$$P(L_{[1..N]}) = Dir(\mathbf{h}^r; \alpha^r), \tag{12}$$

where $Dir(\mathbf{h}^r; \alpha^r)$[1] denotes the Dirichlet prior on $\mathbf{h}^r$ given the model parameter α^r. In this work, the Dirichlet model is region-wise: we partition the whole image into several regions and pool a parameter vector (i.e., α^r) for each region. In contrast with one global Dirichlet model, the mixture of multiple Dirichlet models is more effective because the model parameters are adaptive for different semantic regions, e.g. sky and buildings.

4.1.2 Scene Context Prior

Objects of interest in surveillance systems, e.g., pedestrians, vehicles and bicycles, have strong priors on their potential locations and sizes in images [16]. Let $\mathbf{y}_k^{[t_k^b..t_k^d]} = \{\mathbf{y}_k^{t_k^b}, \mathbf{y}_k^{t_k^b+1}, \ldots, \mathbf{y}_k^{t_k^d}\}$, $S_k^{[t_k^b..t_k^d]} = \{S_k^{t_k^b}, S_k^{t_k^b+1}, \ldots, S_k^{t_k^d}\}$, we define the scene context prior as follows:

$$P(\Pi^{[1..\tau]}|L_{[1..N]}) = \prod_{k=1}^{N} P(\mathbf{y}_k^{[t_k^b..t_k^d]}|L_k) P(S_k^{[1..\tau]}|\mathbf{y}_k^{[t_k^b..t_k^d]}, L_k) \tag{13}$$

with,

$$P(\mathbf{y}_k^{[t_k^b..t_k^d]}|L_k) = \frac{1}{t_k^d - t_k^b + 1} \sum_{t=t_k^b}^{t_k^d} \mathcal{M}^f(\mathbf{y}_k^t|L_k) \tag{14}$$

$$P(S_k^{[t_k^b..t_k^d]}|\mathbf{y}_k^{[t_k^b..t_k^d]}, L_k) = \exp\left[-\frac{1}{t_k^d - t_k^b + 1}\frac{1}{\sigma_3}\sum_{t=t_k^b}^{t_k^d} \|S_k^t - \mathcal{M}^a(\mathbf{y}_k^t|L_k)\|^2\right] \tag{15}$$

where $\mathcal{M}^f(\mathbf{y}_k^t|L_k)$ returns the occurrence frequency of the object category L_k at the position $\mathbf{y}_k^t$, and $\mathcal{M}^a(\mathbf{y}_k^t|L_k)$ returns the predicated area of the object category L_k at $\mathbf{y}_k^t$. σ_3 is a constant (fixed to be 400). Once camera calibrated [16], both $\mathcal{M}^f(\mathbf{y}_k^t|L_k)$ and $\mathcal{M}^a(\mathbf{y}_k^t|L_k)$ can be directly pooled from the training data. $\mathcal{M}^f(\mathbf{y}_k^t|L_k)$ is normalized to be unit one. The location-size map $\mathcal{M}^a(\mathbf{y}_k^t|L_k)$ indicates the minimal blob size for each category at certain position in image. This map can be pre-computed from the training data when system is initialized, and thus almost no additional cost is introduced in the inference stage.

4.1.3 Trajectory Temporal Prior

We define the temporal prior to be the product of two independent priors. The first one is the prior distribution on trajectory lifespan in the video, namely the duration (or number of images) spanned by individual trajectories (from birth to death). Like the recognition prior, we assume it follow with a Dirichlet distribution. Let $\mathbf{h}^t$ denote the histogram each bin indicating the average lifespan of one category. Denote $Dir(\mathbf{h}^t; \alpha^t)$ as the probabilistic predication on $\mathbf{h}^t$ given the model parameter α^t. $\mathbf{h}^t$ is pooled from the currently observed images and α^t is pooled from the training data, both normalized by respective summation.

The second temporal prior is related to the probability distribution on the global properties of an object trajectory,

1. We have, $Dir(\mathbf{h}; \alpha^r) = \frac{1}{B(\alpha^r)}\Pi_{i=1}^{K}(\mathbf{h}_i)^{\alpha_i^r}$ where $\mathbf{h}$ is an K-dimensionality vector and $B(\alpha^r)$ is the normalizing constant which is a multi-nominal beta function.

including birth/death position and the moving direction. Given the training data of annotated trajectories, we aggregate the times of the birth or death positions of objects to generate a 2D frequency histogram, called *birth/death map*. We normalize the frequencies in the birth/death map by their summarization. Figure 4(b) visualizes the birth/death map generated for the scenario in Figure 4(a). We do not distinguish the birth and the death positions. Moreover, we collect all the trajectories in training data, and extract for each trajectory the center positions in each image to build the skeleton line. We further collect all the skeleton lines to construct the *trajectory map*. This idea is originally proposed by Wang et al. in [34], and we use it as a global prior for trajectory parsing. Figure 4(c) shows one trajectory map built for the scenario in Figure 4(a).

Formally, let $\mathcal{M}^o(\mathbf{x}_k^b)$ return the normalized frequency of birth/death at the position $\mathbf{x}_k^b$, $\mathcal{M}^p$ indicate a set of object trajectories, $|\mathcal{M}^p|$ indicate the number of trajectories in $\mathcal{M}^p$, Γ_l^p denote the skeleton of the l^{th} trajectory in $\mathcal{M}^p$. We define the temporal prior as the product of three terms:

$$P(C_{[1..N]})=Dir(\mathbf{h}^t;\alpha^t)\prod_{k=1}^{N}\mathcal{M}^o(\mathbf{x}_k^b)\mathcal{M}^o(\mathbf{x}_k^d)P(\Gamma_k|\mathcal{M}^p) \quad (16)$$

$P(\Gamma_k|\mathcal{M}^p)$ is a mixture model:

$$P(\Gamma_k|\mathcal{M}^p)\propto\sum_{l=1}^{|\mathcal{M}^p|}D^{geo}(\Gamma_k,\Gamma_l^p)\exp\left\{-\mathcal{K}\left[D^{ske}(\Gamma_k,\Gamma_l^p)\right]\right\}, \quad (17)$$

where $D^{geo}(\Gamma_k,\Gamma_l^p)$ denotes the geometric distance between Γ_k and Γ_l^p (calculated by averaging over the minimum distances from the points in Γ_k to the points in Γ_l^p). $\mathcal{M}^o(\cdot)$ is directly estimated from training data. $\mathcal{K}$ is the Gaussian function with kernel size 1. $D^{ske}(\cdot,\cdot)$ returns the similarity distance between two skeleton shapes. Here we define $D^{ske}(\cdot,\cdot)$ using the squared Procrustes distance, as in [23],

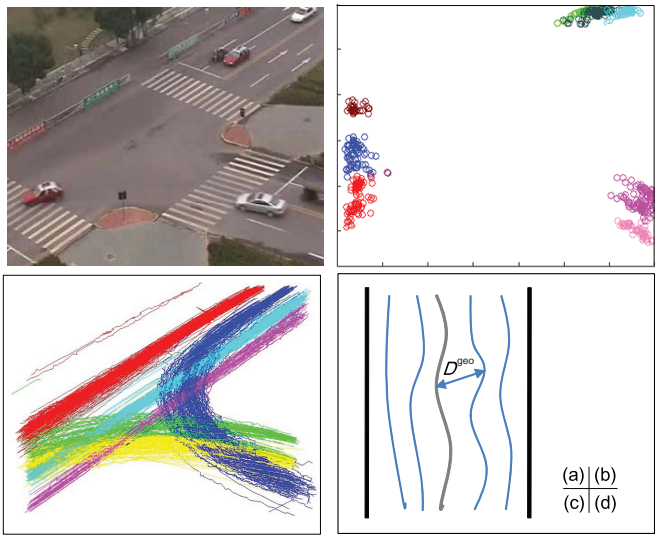

Fig. 4. Scene context modeling. (a) One observed scene; (b) birth/death map; (c) trajectory map; and (d) trajectory skeletons where the blue lines indicate the trajectories in the trajectory map and the gray line indicates one trajectory at the current solution state.

4.2 Likelihood Model

In this work, the likelihood model takes the following form,

$$P(\mathbf{I}|W)=\prod_{k=1}^{N}P(\mathbf{I}|L_k)\prod_{t=t_k^b}^{t_k^d-1}P(\Pi_k^{t+1}|\Pi_k^t,C_k) \quad (18)$$

with,

$$P(\Pi_k^{t+1}|\Pi_k^t,C_k)=\exp\left[-\frac{1}{\sigma_4}D^{apr}\left(F(\Pi_k^{t+1}),F(\Pi_k^t)\right)\right] \quad (19)$$

$$P(\mathbf{I}|L_k)=\exp\left[-\frac{1}{t_k^d-t_k^b+1}\sum_{t=t_k^b}^{t_k^d}\frac{1}{\sigma_5}\mathbb{S}(\Pi_k^t,L_k)\right] \quad (20)$$

where $F(\Pi_k^t)$ indicates the appearance features extracted from Π_k^t, and $\mathbb{S}(\Pi_k^t,L_k)$ indicates the confidence of classifying Π_k^t as L_k by the method [10]. σ_4 and σ_5 are constant parameters (fixed to be 2). $P(\Pi_k^{t+1}|\Pi_k^t,C_k)$ is used to preserve the appearance consistency of individual trajectories over consecutive images. The other term $P(\mathbf{I}|L_k)$ is introduced to combine the outputs of the object categorization method [10].

In summary, we can rewrite the posterior in Eq. (6) to combine the prior terms and the likelihood terms using the form of exponential family [35],

$$P(W|\mathbf{I})\propto\exp\{-z_1\log P(L_{[1..N]})-z_2P(\Pi^{[1..\tau]}|L_{[1..N]}) \quad (21)$$
$$-z_3\log P(C_{[1..N]})-z_4\log P(\mathbf{I}|W)\}$$

where $z_1,..,z_4$ are constants that control the weights of different terms in the final decision. We will introduce an effective method to determine the proper values of these parameters in Section 5.

5 Experiments

In this section, we evaluate the proposed method on public datasets and compare with other popular methods.

TABLE 2
Details of datasets used in this work.

	LHI [41]	PETS [13]	I-80
No. of Clips	8	8	8
No. of images	8644	6455	7920
No. of Objects	241	112	104

5.1 Evaluation Protocol

Parameter Setting. In order to build various prior models as introduced in Section 4.1, we develop an interactive toolkit, which provides three major functions: camera viewpoint calibration, surface property estimation and parameters learning. The parameters in prior models include α^r (used for recognition prior), location-size map, α^t (used for prior on trajectory lifespan), birth/death map and trajectory map. In addition, there are several free parameters used in our method, including $z_1,\ldots,z_4$. In order to determine the proper parameter values, we adopt the method in [42] which bases on Linear Programming (LP). Given the training data, this method begins with

degrading the optimal solution state W^* obtained from the groundtruth to another solution state W' by turning off/on edges in the ST-graph probabilistically. For each W', we define a constraint, namely the posterior probability ratio $\frac{P(W^*)}{P(W')} \geq 1$, that provides a linear inequality in terms of the parameters. We repeat the above procedure to collect multiple constraints, and then use Linear Programming to find a solution of positive parameters with a minimum sum. More details are referred to the literature [42]. We generate 3000 constraints in total and fix the solved parameters in all experiments.

The other parameters of our method are set as follows. The size of image patches is fixed to be 12×12 pixels and the maximum number of patches in one single motion primitive is set to be 8. The size of observation window is fixed to be 30 images. For each observation window, we conduct Algorithm 2 until convergence, after that we move the current observation window with the step size of 10 images, and so on.

Evaluation setting. As aforementioned, our proposed solution consists of several components, including various types of priors, the trajectory representation of ST-graph (with motion primitives), the composite cluster sampling method, and the object categorization module. For comparisons, we introduce two more components. 1) An adjacent graph which takes image patches (extracted from the foreground regions in individual images) as graph nodes. Every node has four neighbor nodes in the same image and two neighbor nodes in the consecutive images. We use the strategies introduced in Section II to construct the graph edges and edge variables except that only positive edges are built between graph nodes. Similar graph structure has also been used in [42]. We compare this adjacent graph with the proposed ST-graph. 2) We also implement the SW-cut sampling algorithm [3] for inference, which ignores the conflicting interactions between graph nodes.

In order to demonstrate the benefit of every individual component, we implement and evaluate several variants of the proposed method. Table 5.1 summarizes the implementation details. The first column shows the abbreviation names of algorithms (TP denotes Trajectory Parsing), and the rest columns show Y if the algorithm (e.g. TP-1) adopts the component (e.g. ST-graph), or blank otherwise. Among these, algorithm TP-1 uses the typical adjacent graph for representation and the SW-cut method [3] for inference, while algorithm TP-2 instead uses the ST-graph and the presented composite cluster sampling method. Algorithm TP-3 extends algorithm TP-2 by additionally integrating the spatial prior (including the Dirichlet prior on categorization and the location-size constraint) into the proposed framework. Algorithm TP-4 further extends TP-3 by imposing the temporal context (including Dirichlet prior on trajectory lifespan, birth/death map and trajectory map), and algorithm TP-5 implements the unified solution proposed in this work. For algorithms TP-1,..TP-4, the likelihood model defined in Eq. (18) only contains the term $P(\Pi_k^{t+1}|\Pi_k^t, C_k)$. In the evaluations, we use the same parameter settings for the above variants.

Metric: We evaluate the proposed method from two aspects: multi-object tracking and object categorization. To quantitatively evaluate tracking performance, We adopt the metrics in [39].

- **Recall** , *number of correctly matched detections / total number of ground-truth detections*;
- **Precision**, *number of correctly matched detections / total number of output detections*;
- **FAF**, *average false alarms per image* (smaller is better);
- **MT**, mostly tracked, percentage of ground truth trajectories which are covered by tracker output for more than 80% in length;
- **ML**, mostly lost, percentage of ground-truth trajectories which are covered by tracker output for less than 20% in length (the smaller the letter).
- **IDS**, ID Switch, the number of times that an object trajectory changes its matched id.
- **MOTP**, Multi Object Tracking Precision, the average ratio of the spatial intersection divided by the union of an estimated object bounding box and the ground-truth bounding box. This metric indicates the position precisions of algorithms' tracks.

We use the toolkit provided by Yang et al. [39] to calculate above metrics. We utilize ROC curve to evaluate object categorization.

5.2 Experiments on LHI, PETs and I-80

We integrate the proposed framework into a surveillance system (refer to the details in [26]), which also includes a background modeling module [17] and an object recognition module [10]. The system is capable of processing $10 \sim 15$ images per second on a Pentium-IV 2.2GHZ computer (with 8GB RAM) after code optimization in the C++ platform. Notice that the BP based optimization procedure for Eq. (1) is parallelized at thread-level to fully take advantage of the CPU-RAM architecture. Figure 5 shows a plot of energy versus iteration number for the following algorithms: 1) algorithm TP-2 which uses ST-graph for representation and the proposed composite cluster sampler for inference; 2) algorithm TP-1 which uses the adjacent graph for representation and the SW-cut method [3] for inference; and 3) algorithm MCMCDA [42] which uses the adjacent graph for representation and the Gibbs Sampler [14] for inference. The data used here is one video clip selected from the LHI dataset and the parameter settings are the same as introduced above. From the curves, we can observe that the composite cluster sampler can converge in about 50 iterations, the fastest convergence of all algorithms. SW-cut achieves the second fastest convergence as it updates larger space at each iteration than the single-site Gibbs sampler (which usually converges in more than $10,000$ iterations). In the following experiments, we set the maximum iterations in Algorithm 1 and 2 to be 40 and 80, respectively.

In this experiment, we use video clips from three public datasets: LHI [41], PETs [13] and I-80 [2] and manually annotate the object bounding boxes within each image as the ground truths of object trajectories. Table 2 depicts the details of each dataset.

We compare our proposed algorithms, namely TP-1, TP-2,...,TP-5, with four popular tracking algorithms. i) The

2. http://ngsim.fhwa.dot.gov/

TABLE 3
Evaluation setting of the proposed solution to trajectory parsing. Dir-C: Dirichlet Prior on object categorization; L-S: location-size constraint; Dir-L: Dirichlet prior on trajectory lifespan; B-D Map: birth/death map; Trj-Map: trajectory map. ST-graph: spatio-temporal graph with nodes of motion primitives; Adj-graph: adjacent graph with nodes of image patches. C-Cluster: the proposed composite-cluster sampling algorithm; SW-cut: the SW-Cut sampling method [3].

	Prior Terms						Representation		Inference	
Alg-Name	Dir-C	L-S	Dir-L	B-D Map	Trj-Map	Obj-Categorization	ST-graph	Adj-graph	C-Cluster	SW-cut
TP-1								Y		Y
TP-2							Y		Y	
TP-3	Y	Y					Y		Y	
TP-4	Y	Y	Y	Y	Y		Y		Y	
TP-5	Y	Y	Y	Y	Y	Y	Y		Y	

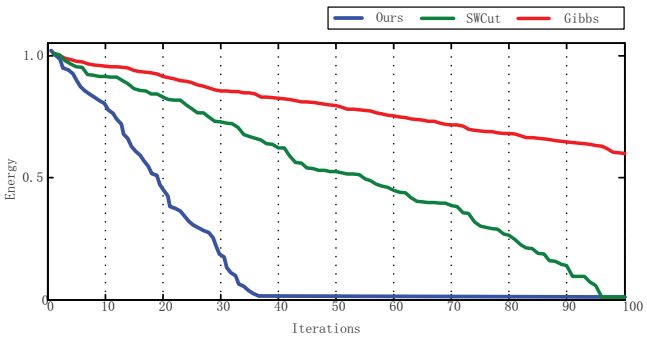

Fig. 5. Energy-vs-Iteration for various inference methods, including the Composite-Cluster sampler, the SW-cut method [3] and the Gibbs sampler [42]. Given the posterior probability $P(W)$ at the current step, the energy is calculated as $-\log(P(W))$.

MCMC-based Data Association (**MCMCDA**) by Yu et al. [42]. ii) The method proposed by Birchfield et al. [5] which combines the ideas of Lucas-Kanade and Horn-Schunck to jointly track sparse interest points and edges (**JLK**). JLK has several parameters, e.g. regularization weight, which are set to be constant empirically as in [5]; iii) The spatial selection algorithm for attentional visual tracking (**AVT**) proposed by Yang et al. [40]. The key parameter is the number of attentional regions (ARs) used for representing one object of interest. Following the suggestions in [40], we use $60 \sim 70$ ARs for the large-size objects and $30 \sim 40$ ARs for small-size objects (the expected size of one object is up to the category and location in the image). We set other parameters the same as in the original article [40]. The inputs to the above baseline methods contain the foreground regions, which are extracted using a background model [17], and the detected foreground blobs from which tracking is performed. To avoid insufficient training of the background model, for each video sequence, we manually annotate the objects of interest for the first 30 images and use these annotations to initialize the above tracking algorithms. It is worth noting that algorithm TP-1 can be viewed as a special implementation of the algorithm MCMCDA [42], whereas the differences are two folds: 1) TP-1 does not use the prior model as in [42][3]; and 2) TP-1 uses the SW-Cut method for inference rather than the Gibbs sampler in [42]. It is interesting to compare these two different implementations on the same dataset.

3. The prior model in [42] is to prefer long trajectories, which are not applicable in surveillance environment, because the surveillance systems usually have much stronger priors about the locations, lengths and density of object trajectories in video.

TABLE 4
Quantitative tracking results on the LHI database [41]. R: recall rate; P: precision rate.

	R(%)	P(%)	FAF	MT(%)	ML(%)	MOTP(%)	IDS/GT
MCMCDA [42]	79.4	81.0	0.458	78.9	11.2	73.5	54/241
JLK [5]	82.1	84.7	0.257	87.5	6.3	79.4	31/241
AVT [40]	85.3	86.1	0.219	89.6	4.1	83.9	27/241
TP-1	80.1	81.2	0.467	81.2	9.1	81.3	52/241
TP-2	88.6	89.1	0.313	89.5	3.8	87.9	43/241
TP-3	89.2	91.2	0.268	91.4	3.5	88.5	37/241
TP-4	91.3	92.4	0.202	92.1	2.9	90.4	36/241
TP-5	**92.1**	**93.7**	**0.192**	**93.4**	**2.1**	**92.6**	29/241

TABLE 5
Quantitative tracking results on the PETs database [13].

	R(%)	P(%)	FAF	MT(%)	ML(%)	MOTP(%)	IDS/GT
MCMCDA [42]	83.1	81.4	0.797	76.8	9.3	82.5	21/112
JLK [5]	85.2	86.4	0.378	79.2	4.9	84.9	17/112
AVT [40]	86.6	84.1	0.457	81.3	4.5	87.2	16/112
TP-1	83.4	82.5	0.754	80.1	8.7	84.7	18/112
TP-2	88.1	86.3	0.405	84.3	3.7	89.5	9/112
TP-3	89.3	88.3	0.323	85.6	3.5	90.3	9/112
TP-4	90.2	88.7	0.314	86.1	3.3	91.4	7/112
TP-5	**91.1**	**89.6**	**0.273**	**87.4**	**3.1**	**93.1**	6/112

TABLE 6
Quantitative tracking results on the I-80 database.

	R(%)	P(%)	FAF	MT(%)	ML(%)	MOTP(%)	IDS/GT
MCMCDA [42]	82.4	83.4	0.926	74.3	7.5	79.8	21/104
JLK [5]	89.1	85.6	0.318	85.2	5.1	86.4	17/104
AVT [40]	88.1	86.8	0.227	85.7	4.9	87.7	13/104
TP-1	83.2	84.7	0.717	79.5	6.9	85.2	18/104
TP-2	89.3	89.2	0.332	86.2	4.2	88.6	10/104
TP-3	91.4	89.9	0.253	87.4	3.6	90.3	7/104
TP-4	92.6	90.1	0.231	87.6	2.7	91.8	5/104
TP-5	**93.2**	**92.8**	**0.104**	**88.5**	**2.3**	**92.7**	2/104

Figure 6 shows several video sequences overlaid with the tracking results by algorithm TP-5. Most of the videos are very challenging due to the crowded objects, scale changes, severe occlusions and low resolutions. Each cell includes the mask images of foreground regions (top row) proposed by the background modeling module and the results of tracking (bottom row). For the last sequence from I80 dataset, we only show three numbered objects of interest for the ease of

display. From the results, we can observe that algorithm TP-5 can successfully recover the object trajectories in all the video clips. Particularly, it is interesting to observe that, while there are severe occlusions (half-occlusions or full-occlusions) in videos (e.g., the top-right video clip), our method can still work robustly. This is due to several characterizations of our method. First, while constructing motion primitives, we allow one patch does not match with any patches in the next image. Second, we aim to solve the optimal spatial partition for every image while inferring the temporal matching at the same step. Third, the prior models are very helpful to predict the potential object positions in incoming images given the current tracking results. In practice, we impose a boolean status variable for each object of interest indicating whether it is occluded (determined according to the tracking results of other objects and scene knowledge, e.g., whether two objects move to the same location, or whether one object is moving behind a vertical surface). If yes, we will increase the weight of temporal prior in the final decision. This simple strategy has shown great success in practice.

Tables 4, 5 and 6 report the quantitative comparisons of our method and other baseline algorithms on the LHI [41], PETs and I-80 datasets. From the results, we can have the following observations.

- Among all the algorithms, TP-5 achieves the best recall and precision rates on all the three datasets. Specially, algorithm AVT [40] is known as the stat-of-the-art tracking algorithm, and although its performance is already good, our method remarkably outperforms it with the margins of 6.8 percentages in terms of recall rate and 7.6 percentages in terms of precision rate on the LHI database. Also, algorithm TP-5 clearly outperforms the traditional sampling based algorithms, including MCMCDA [42] and JPDA [30].
- Our method achieves the FAF of 0.192, 0.273 and 0.104, on the LHI, PETs and I-80 datasets, respectively. In contrast, the corresponding best results of other four baselines are 0.219, 0.378 and 0.227, which are much higher than the proposed solution. Similar improvements can be achieved in terms of the metric of MT. These comparisons on the above two metrics show that our method bears higher robustness which favors the practical applications.
- Algorithm TP-2, which does not use any prior terms, achieves better performance than MCMCDA [42], JLK [5] and AVT [40] in terms of both recall rate and precision. These comparisons well demonstrate the advantages of our proposed ST-graph that integrates motion primitives and the ST-graph representation.
- The usage of spatial prior and temporal prior consistently boosts the tracking performance on all datasets. This observation comes from the comparisons between the algorithms TP-2,TP-3,TP-4 and TP-5. Although the prior terms used here are usually limited to certain surveillance environment, they are valuable for practical applications which require robust trajectory parsing.

In addition, algorithm TP-1 achieves the comparable performance with MCMCDA [42], while it is worthy noting that TP-1 is much more efficient in computation (SW-Cut usually converges much faster than Gibbs sampler) and easier in implementation (the only MCMC jump is to turn on/off edges probabilistically).

From the comparisons between algorithms TP-5 and TP-4 in Tables 4, 5 and 6, we can observe that involving object categorization in inference can clearly improve the robustness of parsing trajectories. In order to demonstrate how the unified formulation takes effect of object categorization, we compare the proposed unified solution with the separate categorization algorithm [10] which is performed independently and sequentially for each observed object/trajectory. The data are from LHI dataset [41]. For each trajectory, once the categorization results in individual images obtained, we use the majority voting strategy to determine the final category label of individual trajectories. Thus, we can calculate the false positive rate and the true positive rate for each method. We plot the ROC curves of three categories: pedestrians, sedans, and bicycles in Figure 7(a), Figure 7(b) and Figure 7(c), respectively. The solid curves represent the recognition performance by our framework, and the dashed ones represent the results by conducting object categorization independently . The Area Under Curve (AUC) for each method is also shown in the figure. We can observe that significant improvements on categorization performance are obtained due to the integration of our framework.

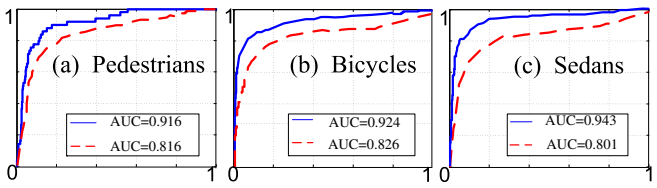

Fig. 7. ROC curves of object categorization; (a) pedestrians, (b) bicycles, and (c) sedans. Horizontal direction indicates the false positive rate and the vertical direction indicates the true positive rate. The solid curves represent the recognition with trajectory parsing and the dashed curves represent the recognition of being executed independently without our framework.

5.3 Experiments on PETs'09 and TRECVID'08

We further evaluate the proposed algorithm TP-2 on PETS'09 dataset. The clip is selected from the S2L1 subset (first viewpoint). It contains 795 images. This set has been used in the previous works [1] [38]. We compare with three recently proposed tracking algorithms, i) EMM [1], that formulates multi-object tracking as the continuous energy minimization task; ii) PRIMPT [22], that addresses the multi-person tracking problem particularly; and iii) MIL [38] that employs a multiple instance learning method to associate tracklets based on both appearance and motion models. Table 7 reports the quantitative tracking results on PETS'09. The figures of the three baseline methods are directly from their respective papers. For our method TP-2, all parameters are the same as in the last subsection. From the table, we could observe that all these algorithms

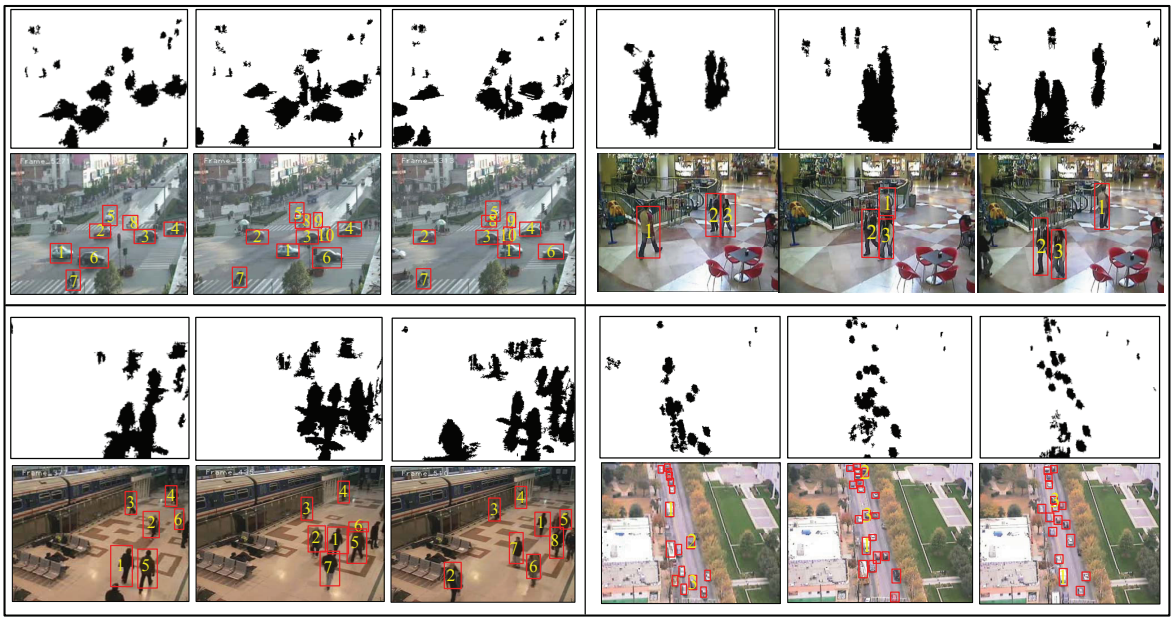

Fig. 6. Exemplar results of trajectory parsing on challenging scenes. Each plot shows three images and their foreground masks. Each recovered trajectory can be identified by the bounding boxes (in red) with the numbers in images. The clips in the first row come from the LHI dataset [41], the down-left clip from the PETs [13] and the down-right clip from the I-80.

TABLE 7
Quantitative tracking results on PETS'09 .

	Recall(%)	Precision(%)	FAF	MT(%)	IDS/GT
EMM [1]	-	-	-	82.6	15/23
PRIMPT [22]	89.5	99.6	0.020	78.9	1/19
MIL [38]	91.8	99.0	0.053	89.5	0/19
Ours	95.8	99.4	0.031	92.6	1/19

TABLE 8
Quantitative tracking results on TRECVID'08 .

	Recall(%)	Precision(%)	FAF	MT(%)	ML(%)	IDS/GT
OffLineCRF [37]	79.2	85.8	0.996	78.2	4.9	253/919
OLDAMs [21]	80.4	86.1	0.992	76.1	4.6	224/919
PRIMPT [22]	79.2	86.8	0.920	77.0	5.2	171/919
OnlineCRF [39]	79.8	87.8	0.857	75.5	5.8	147/919
Ours	81.3	87.2	0.831	80.3	5.1	142/919

can achieve high Precision rates whereas our method achieves the highest Recall rate.

We also evaluate our method on the challenging TRECVID'08 dataset. We use the 9 video clips selected by Yang et al. in [39], each of which has 5000 images. These clips are filed in a busy airport, and have a high density of people with frequent occlusions. We compare with four tracking algorithms: i) OffLineCRF [37] which tracks objects of interest by training an offline CRF model on pre-labeled groundtruth data; ii) OLDAMs [21] that proposes an online learned discriminative appearance models for tracking; iii) PRIMPT [22] and iv) OnlineCRF [39] that applies the online-learned Conditional Random Field model for multi-target tracking. Table 8 reports the comparison results by our method (TP-2) and other popular tracking algorithms. The figures of those baseline methods are directly from their respect papers. Our method achieves comparable tracking performance as other baseline algorithms. These comparisons clearly demonstrate our method, even without scene context knowledge, can still achieve robust tracking in challenging videos.

6 Conclusion and Future Work

In this paper, we proposed a unified framework for jointly solving object segmentation, tracking and categorization in surveillance videos. We presented a novel spatio-temporal graph representation, which takes motion primitives as graph nodes, and describes both cooperative and conflicting relationships between graph nodes by positive and negative edges respectively. The Bayesian treatment of trajectory parsing problem enables naturally integrating various types of context information. To optimize, an efficient composite cluster sampling method was utilized to overcome the problem of combinatorial search of the optimal solution by constructing large MCMC jumps. Our

method can perform object tracking automatically, without the requirement of manual initialization. The comparisons with the state-of-the-art tracking approaches on challenging datasets demonstrated its advantages in achieving high-quality tracking as well as wide applicability in practice.

In the future research, we plan to investigate the proposed method in the following two aspects. First, the proposed ST-graph can be used to combine multiple diverse cues adaptively. In consideration of the widely use of graph representation, it is easily to extend ST-graph for other image tasks, i.e. image segmentation, and object detection. Second, the presented motion primitives and related generation algorithm provide a general way to represent video sequences, and thus can be applied for other video tasks, e.g. event analysis, super-resolution, and video classification.

7 Acknowledgment

The dataset used in this work are provided by the Lotus Hill Institute. We thanks Professor Song-chun Zhu (UCLA) and Professor Alan Yuille (UCLA) for their constructive comments.

References

[1] A. Andriyenko and K. Schindler. Multi-target tracking by continuous energy minimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[2] S. Avidan. Ensemble tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 29(2):261–271, 2007.

[3] A. Barbu and S.C. Zhu. Generalizing swendsen-wang for image analysis. *Journal of Computational and Graphical Statistics*, 16(4):877–900, 2007.

[4] A. Basharat, Y. Zhai, and M. Shah. Content based video matching using spatiotemporal volumes. *Computer Vision and Image Understanding*, 110(3):360–377, 2008.

[5] S. Birchfield and S.Pundlik. Joint tracking of features and edges. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1–6, 2008.

[6] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory of warping. In *European Conference on Computer Vision*, volume 3024, pages 25–36, 2004.

[7] A. Bugeau and P. Pérez. Track and cut: simultaneous tracking and segmentation of multiple objects with graph cuts. *EURASIP Journal on Image and Video Processing - Special Issue on Video Tracking in Complex Scenes for Surveillance Applications*, 317278:1–14, 2008.

[8] R. Collins, Y. Liu, and M. Leordeanu. On-line selection of discriminative tracking features. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(10):1631–1643, 2005.

[9] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[10] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *European Conference on Computer Vision*, volume 1, pages 7–13, 2006.

[11] R. Edwards and A.Sokal. Generalization of the fortuin-kasteleyn-swendsen-wang representation and monte carlo algorithm. *Physical Review*, 38(6):2009–2012, 1998.

[12] P. Felzenszwalb and D. Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision*, 70(1):41–54, 2006.

[13] R. Fisher. The pets04 surveillance ground-truth data sets. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2004.

[14] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, (6):721–741, 1984.

[15] A. Hadid and M. Pietikainen. Face recognition with local binary patterns: Application to recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.

[16] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2137–2144, 2006.

[17] W. Hu, H. Gong, S.C. Zhu, and Y. Wang. An integrated background model for video surveillance based on primal sketch and 3d scene geometry. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1–8, 2008.

[18] Imran N. Junejo and Hassan Foroosh. Trajectory rectification and path modeling for video surveillance. In *International Journal of Computer Vision*, volume 1, pages 1–7, 2007.

[19] Z. Khan, T. Balch, and F. Dellaert. Mcmc-based particle filtering for tracking a variable number of interacting targets. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(11):1805–1819, 2005.

[20] I. Kompatsiaris and M. Strintz. Spatiotemporal segmentation and tracking of objects for visualization of videoconference image sequences. *IEEE Transactions on Circuits and Systems for Video Technology*, 10:1388–1402, 2000.

[21] C.-H. Kuo, C. Huang, and R. Nevatia. Multi-target tracking by on- line learned discriminative appearance models. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[22] C.-H. Kuo and R. Nevatia. How does person identity recognition help multi-person tracking. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[23] L. Lin, K. Zeng, X. Liu, and S.C. Zhu. Layered graph matching by cluster sampling with collaborative and competitive interactions. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1351–1358, 2009.

[24] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. Freeman. Sift flow: Dense correspondence across different scenes. In *European Conference on Computer Vision*, volume 5304, pages 28–42, 2008.

[25] X. Liu, J. Feng, S. Yan, L. Lin, and H. Jin. Segment an image by looking into an image corpus. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 2249–2256, 2011.

[26] X. Liu, L. Lin, S. Yan, and H. Jin. Adaptive tracking via learning hybrid template online. *IEEE Transactions on Circuits and Systems for Video Technology, in press*, 2011.

[27] X. Liu, L. Lin, S.C. Zhu, and H. Jin. Trajectory parsing by cluster sampling in spatio-temporal graph. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 739–746, 2009.

[28] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):85–111, 1953.

[29] J. Porway and S.C. Zhu. Computing multiple solutions in graphical models by cluster sampling. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 33(9):1713–1727, 2011.

[30] C. Rasmussen and G. Hager. Joint probabilistic techniques for tracking multi-part objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 16–21, 1998.

[31] D. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):84–90, 1979.

[32] B. Song, T. Jeng, E. Staudt, and A. Roy-Chowdhury. A stochastic graph evolution framework for robust multi-target tracking. In *European Conference on Computer Vision*, 2010.

[33] R. Swendsen and J. Wang. Nonuniversal critical dynamics in monte carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987.

[34] X. Wang, K. Tieu, and E. Grimson. Learning semantic scene models by trajectory analysis. In *European Conference on Computer Vision*, volume 3, pages 110–123, 2006.

[35] Y. Wu, S.C. Zhu, and C. Guo. From information scaling of natural images to regimes of statistical models. *Quarterly of Applied Mathematics*, 66(1):81–122, 2008.

[36] Z. Wu, M. Betke, and T. Kunz. Efficient track linking methods for track graphs using network-flow and set-cover techniques. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[37] B. Yang, C. Huang, and R. Nevatia. Learning affnities and dependencies for multi-target tracking using a crf model. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[38] B. Yang and R. Nevatia. Multi-target tracking by online learning of non-linear motion patterns and robust appearance models. In *IEEE Computer Vision and Pattern Recognition*, 2012.

[39] B. Yang and R. Nevatia. An online learned crf model for multi-target tracking. In *IEEE Computer Vision and Pattern Recognition*, 2012.

[40] M. Yang, J. Yuan, and Y. Wu. Spatial selection for attentional visualtracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1–8, 2007.

[41] B. Yao, X. Yang, and S.C. Zhu. Introduction to a large scale generalpurpose groundtruth dataset: Methodology, annotation tool, and benchmarks. In *Energy Minimization Methods in Computer Vision and Pattern Recognition, Springer LNCS*, volume 4697, pages 169–183, 2007.

[42] Q. Yu, G. Medioni, and I. Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[43] T. Yu, C. Zhang, M. Cohen, Y. Rui, and Y. Wu. Monocular video foreground/background segmentation by tracking spatial-color gaussian mixture models. In *IEEE Workshop on Motion and Video Computing*, 2007.

[44] X. Yuan, X. Liu, and S. Yan. Visual classisfication with multitask joint sparse representation. In *IEEE Transaction on Image Processing*, volume 21, pages 4349–4360, 2012.

[45] T. Zhao, R. Nevatia, and B. Wu. Segmentation and tracking of multiple humans in crowded environments. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 30(7):1198–1211, 2008.

Xiaobai Liu is currently a Postdoctoral Research Scholar in the Department of Statistics, University of California at Los Angeles (UCLA). He received the Ph.D from the school of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China in 2011. Before that, he had been working in the Dept. Electric & Computer Engineering, National University of Singapore and working as the Research Associate for Professor Shuicheng Yan. He also worked as Research Associate at Lotus Hill Institute under the supervision of Professor Song-Chun Zhu from 2007 to 2008. He has published more than 20 peer-reviewed articles over a series of research topics and now his research interests fall in computer vision, machine learning and large scale image retrieval.

Liang Lin received the B.S. and Ph.D. degrees from the Beijing Institute of Technology (BIT), Beijing, China, in 1999 and 2008, respectively. From 2006 to 2007, he was a joint Ph.D. student with the Department of Statistics, University of California, Los Angeles (UCLA). He was a Post-Doctoral Research Fellow with the Center for Image and Vision Science of UCLA. From 2007 to 2009, he was a Senior Research Scientist with the Lotus Hill Research Institute, China. He is currently an Associate Professor with the Software School of Sun Yat-Sen University, Guangzhou, China. He was awarded by the "Hundred Talents Program" of the University in 2009 and by the "Program for New Century Excellent Talents in University" in 2012. His current research interests include but are not limited to computer vision, pattern recognition, machine learning, and multimedia technology. He has authored or co-authored over 60 academic papers over a wide range of research topics. Dr. Lin has received a number of academic honors, including several scholarships while pursuing Ph.D. degree, the Beijing Excellent Students Award in 2007, China National Excellent PhD Thesis Award Nomination in 2010, Best Paper Runners-Up Award in ACM NPAR 2010, and Google Faculty Award in 2012.

Hai Jin is a professor of computer science and engineering at the Huazhong University of Science and Technology (HUST) in China. He is now Dean of the School of Computer Science and Technology at HUST. Jin received his PhD in computer engineering from HUST in 1994. In 1996, he was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz in Germany. Jin worked at The University of Hong Kong between 1998 and 2000, and as a visiting scholar at the University of Southern California between 1999 and 2000. He was awarded Excellent Youth Award from the National Science Foundation of China in 2001. Jin is the chief scientist of ChinaGrid, the largest grid computing project in China.

Jin is a senior member of the IEEE and a member of the ACM. Jin is the member of Grid Forum Steering Group (GFSG). He has co-authored 15 books and published over 400 research papers. His research interests include computer architecture, virtualization technology, cluster computing and grid computing, peer-to-peer computing, network storage, and network security

Jin is the steering committee chair of International Conference on Grid and Pervasive Computing (GPC), Asia-Pacific Services Computing Conference (APSCC). Jin is a member of the steering committee of the IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid), the IFIP International Conference on Network and Parallel Computing (NPC), and the International Conference on Grid and Cooperative Computing (GCC), International Conference on Autonomic and Trusted Computing (ATC), and International Conference on Ubiquitous Intelligence and Computing (UIC).