Online Moving Camera Background Subtraction

Ali Elqursh and Ahmed Elgammal

Rutgers University

Abstract. Recently several methods for background subtraction from moving camera were proposed. They use bottom up cues to segment video frames into foreground and background regions. Due to this lack of explicit models, they can easily fail to detect a foreground object when such cues are ambiguous in certain parts of the video. This becomes even more challenging when videos need to be processed online. We present a method which enables learning of pixel based models for foreground and background regions and, in addition, segments each frame in an online framework. The method uses long term trajectories along with a Bayesian filtering framework to estimate motion and appearance models. We compare our method to previous approaches and show results on challenging video sequences.

1 Introduction

One may argue that the ultimate goal of computer vision is to learn and perceive the environment in the same way children learn. Without access to pre-segmented visual input, infants learn how to segment objects from background using low level cues. Inspired by this evidence, significant effort in the computer vision community has focused on bottom up segmentation of images and videos. It has become ever more important with the proliferation of videos captured by moving cameras.

Our goal is to develop an algorithm for foreground/background segmentation from freely moving camera in a online framework that is able to deal with arbitrary long sequences. Traditional video segmentation comes in different flavors depending on the application but falls short of achieving this goal. In background subtraction, moving foreground objects are segmented by learning a model of the background with the assumption of a static background. Alternatively, motion segmentation methods attempt to segment sparse point trajectories based on coherency of motion. However, they lack a model of the appearance of foreground or background. Video object segmentation attempts to segment an object of interest from the video with no model of the scene background. On the other hand there are several segmentation techniques that attempts to extend traditional image segmentation to the temporal domain. Such techniques are typically limited to segmenting a short window of time.

It is frequently the case however, that if one only considers a short window of frames, that low level cues may be ambiguous. Existing approaches either ignore this problem or resort to processing the whole video offline. Offline methods can

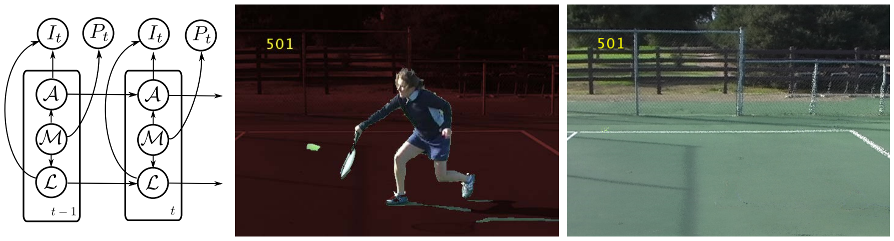

Fig. 1. Left:(a) Graphical model. Center: (b) Automatic segmentation obtained on long sequence with fast articulated motion. Our method is able to segment the tennis ball even when no trajectories exist on it. Right: (c) One sample from the background pixel based appearance model learned automatically.

typically produce good results on short sequences but the complexity increases rapidly as more frames need to be processed. The key to solving this problem is to recognize that to handle long sequences in an online way one has to learn and maintain models for the background and foreground regions. Such models serve the purpose of compactly accumulating the evidence over a large number frames and are essential for high level vision tasks.

The contribution of this paper is a novel online method that learns appearance and motion models of the scene (background and foreground) and produces segmentations of video frames. It uses long term point trajectories and thus is able to accumulate information over long sequences of frames while at the same time performs a constant computation per frame. To achieve this we describe a method to automatically update a low-dimensional representation of these trajectories and incrementally update a set of clusters in a novel coordinate free way. Rather than producing a single segmentation as an output, it uses Bayesian filtering to maintain a belief over different labellings, and appearances of the background and foreground. This enables our approach to recover from errors. By combining long term sparse trajectories and dense models of motion and appearance, our method is able to achieve superior results to the state of the art. We also devised an automatic method to determine foreground background labeling based on multiple static and dynamic cues.

Our method has several advantages over existing approaches. It processes frames online and thus can easily handle arbitrary long videos. The use of long term trajectories allows it to accumulate long term motion information and prevents merging of objects that were known to move differently. Unlike affine motion segmentation, we accomplish this without assuming an affine camera model, thus enabling automatic segmentation of articulated and non-rigid motion. Our approach is able to handle multiple moving objects and maintain motion and appearance models. Such models enables our approach to learn the appearance of the foreground and background even if they are partially occluded. For example, Fig 1. shows a frame of tennis sequence and the corresponding background model learned. Notice, that we are able to detect the tennis ball even when there are no trajectories on it. Finally, our appearance models can be used by a higher level reasoning framework to learn richer models of objects.

2 Related Work

Background Subtraction from Moving Camera. Traditionally, background subtraction methods [24,17,5] attempt to achieve foreground segmentation by assuming that any motion in the video data is due to moving objects. The success of these algorithms has led to their ubiquitous use in surveillance systems where this assumption is always satisfied. Due to the assumption of static camera, this severely limits their use in videos shot by a moving camera. Several attempts to relax this assumption has led to methods which compensates for the motion of the camera [7,14,10]. These methods use a homography or a 2D affine transform to compensate for the motion. This allows them to handle scenes that can be approximated by a plane or when the camera rotates but does not translate.

Recently, several methods were devised for background subtraction from moving camera. [15] uses orthographic motion segmentation over a sliding window to segment a set of trajectories. This is followed by sparse per frame appearance modeling to densely segment images. The use of orthographic motion segmentation means that motion information outside the sliding window is lost and the sparse appearance modeling fails to capture object boundaries when appearance information is ambiguous. Due to the dependence on long term trajectories only, regions with no trajectories may be disregarded as background altogether. Finally, the method fails if the assumption of orthographic projection is not satisfied. In [9], a method is proposed that maintains block based appearance models in a Bayesian framework. To update the appearance models, the method iterates between estimating the motion of the blocks and inferring the labels of the pixels. Once converged, the appearance models are used as a prior for the next frame and the process continues. Due to the iterative nature of the approach the method is susceptible to local minimum. Although motion information between successive frames is estimated, it is only used to estimate the labels in the current frame and does not carry on to future frames. In contrast, we maintain motion information via long term trajectory and maintain a belief over different labellings in a Bayesian filtering framework.

Motion Segmentation. Motion segmentation techniques relies on motion cues to segment the images [20]. They can roughly be divided into two groups; one that relies on sparse point trajectories and those that performs dense segmentations. Multi-body factorization [4,8] uses sparse point trajectories and the affine camera model to segment the trajectories belonging to multiple rigid objects. However, they suffer from major drawbacks. First, they assume that all trajectories are visible over all frames which severely limits their application to short video sequences. A few recent methods [13,6] have tried to overcome this limitation by assuming that only some trajectories span the whole sequence, but as the problem is inherent to factorization this can be successful only to a certain degree. Second, they assume that the objects are rigid which limits their applicability. Finally, they produce a sparse segmentation and therefore requires further post-processing to segment the image. [3,11], computes distances defined over long term trajectories followed by spectral clustering to group trajectories by object. This is followed by applying a hierarchical variational approach to segment regions belonging to different objects. Though the method does not rely on having trajectories span the entire sequence, their approach is not online. In comparison, our method incrementally tracks and segment the foreground while learning models of their appearance and motion. Dense motion segmentation methods[16] use optical flow and normalized graph cut to segment a graph formed from the video. Layered motion segmentation methods [22,19,25,12] segment videos into segments with consistent motion based on some notion of consistency.

3 Overview

We begin by introducing some notation. Our Bayesian filtering framework is represented by the graphical model in Fig 1(a). At time t our state consists of a tuple $s_t = (\mathcal{A}_t, \mathcal{M}_t, L_t)$. $\mathcal{A}_t = \{A_{b,t}, A_{f,t}\}$ represents the appearance models of the background and foreground respectively. Similarly, $\mathcal{M}_t = \{M_{b,t}, M_{f,t}\}$ are the motion models for the background and foreground. Finally, $L_t = \{l_t^i : l_t^i = \{b, f\}, i = 1 \ldots N\}$ where N is the number of pixels, is a pixel wise labeling of the image pixels at time t. For convenience, let $\phi(i) = (x, y)$ be the function that transforms pixel indexes to coordinates.

We adopt a camera centric representation for both the appearance and motion models. Let $k = \{b, f\}$ denote which layer the variable belongs to. Let $a_{k,t}^i$ denote a random variable representing the appearance of the ith pixel of layer k at time t. Let $m_{k,t}^i = [u_{k,t}^i \; v_{k,t}^i]^T$ denote a random variable representing the reverse motion of pixel i of the k^{th} layer between frames t and $t-1$. By grouping these random variables by layer we get $A_{k,t} = \{a_{k,t}^i : i = 1 \ldots N\}$ and $M_{k,t} = \{m_{k,t}^i : i = 1 \ldots N\}$.

We have two sources of observations, frames I_t and sparse labeled motion vectors P_t. Let I_t^i denote the color of the i^{th} pixel. The labeled sparse motion vectors at frame t is a set $P_t = \{p_{j,t} : j = 1 \ldots M\}$ of tuples $p_{j,t} = (q_{j,t}, w_{j,t}, l_{j,t})$, where q is the pixel location, $w_{j,t} = [u \; v]^T$ denotes the motion vector and $l_{j,t}^p = \{f, b\}$ denotes its layer.

In our model, foreground and background are represented as two layers of pixels. Each layer moves according to the motion vectors in the corresponding motion model. To generate a frame, pixels from the background and foreground are selectively selected based on the labels. Formally, the dynamics of the system can be described by the following equations.

$$L_t = \Omega(M_{f,t}, L_{t-1}), \; \mathcal{A}_t = g(\mathcal{A}_{t-1}, \mathcal{M}_t), \tag{1}$$

$$\Omega_i(M_{f,t}, L_{t-1}) = l_{t-1}^{j(i,f)}, \; g_k^i(A_{k,t}, M_{k,t}) = a_{k,t-1}^{j(i,k)}, \tag{2}$$

where $j(i,k) = \phi^{-1}(\phi(i) + m_{k,t}^i)$, $i = 1 \ldots N$. The function $\Omega = [\Omega_i], i = 1, \ldots, N$ takes as input the motion model of the foreground and labels at time $t-1$ and produces the labels at time t. Simply put, the label of i^{th} pixels at time t is the same as the label of a pixel $j(i,f)$ which is $m_{f,t}^i$ pixels away in the previous frame. Similarly the function $g = [g_k^i], i = 1 \dots N,\ k = \{f, b\}$ takes as input the motion model of each layer and moves the appearance models accordingly. Together, these functions describes a process by which the new appearances of the foreground and background are generated given the motion of each pixel of each layer. The observation model can be similarly described by

$$\begin{gathered} I_t = h(\mathcal{A}_t, L_t),\ P_t = z(\mathcal{M}_t), \\ h_i(\mathcal{A}_t, L_t) = a_{k,t}^i + \epsilon_I, \quad k = l_t^i, \epsilon_I \sim \mathcal{N}(0, \Sigma_I) \\ z_j(\mathcal{M}_t) = \{j, m_k^j + \epsilon_P, k\} for\ j \in 1 \dots N, \quad \epsilon_P \sim \mathcal{N}(0, \Sigma_P) \end{gathered} \tag{3}$$

The function $h = [h_i]$, $i = 1 \dots N$ describes how the image is generated given the appearance models and the labels. If the label of pixel i is f, then the observed appearance is the same as the appearance of pixel i in the foreground model $a_{f,t}^i$ plus some white noise with covariance Σ_I and vice versa. The function $z = [z_j]$, $j = 1 \dots N$ describes how labeled sparse motion vectors P_t are generated given the motion model. The observed sparse motion vector is simply the pixel index, the motion vector from the corresponding motion model with an added white noise with covariance Σ_P, and the label of the vector. Without loss of generality, we observe P_t for a subset of pixels only. Since our Bayesian filtering framework assumes that the observation P_t is available, we describe later how to compute a sparse set of motion trajectories and their associated labels.

Our algorithm can be summarized in the following steps. Using the first few frames in the video sequence we perform initialization (Subsection 7). For each consecutive frame afterwards, we maintain a low dimensional representation of the sparse trajectories and cluster them (Section 4). Next, we use multiple cues to label each cluster as foreground or background and thus obtain P_t (Section 5). From the clustered sparse trajectories, the motion models $\mathcal{M}_t$ are inferred. Finally, given the frame I_t, the previous frame appearance models $\mathcal{A}_{t-1}$ and labels L_{t-1}, and the inferred motion model $\mathcal{M}_t$ compute the updated appearance model and labels $\mathcal{A}_t$ and L_t (Section 6).

4 Long-Term Trajectory Modeling

We model each trajectory up to time t as a single point in a low dimensional embedding space. The coordinate of each trajectory in this space is continuously updated. Trajectories belonging to the background are expected to lie on a low-dimensional manifold (background manifold) while trajectories belonging to the foreground objects lie on separate manifolds. This arises due to the similarity in spatial location and motion between neighboring trajectories. Fig 2. shows a set of trajectories in the image space and their corresponding manifolds in the embedding space. In the embedding space we model each trajectory manifold using a Gaussian Mixture Model(GMM) where each patch of the manifold is modeled with a multivariate Gaussian. This representation is continuously updated in a coordinate free manner.

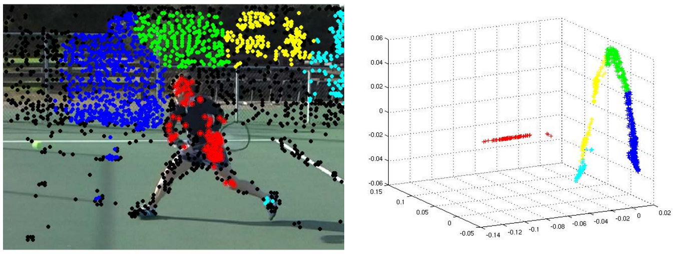

Fig. 2. Trajectories in the image and their corresponding embedding coordinates. Each cluster is marked with a different color. Black dots are short trajectories not yet added to the embedding. The foreground cluster is marked with two red concentric circles around its trajectories. (Best viewed in color).

Given two trajectories, we define two distance metrics $d^M(T_A, T_B)$ and $d^S(T_A, T_B)$ representing the difference in motion and spatial location respectively

$$d^M(T_A, T_B) = max_{t\in(A\cap B)} d^M_{t-4:t}(T_A, T_B), \quad d^S(T_A, T_B) = max_{t\in(A\cap B)} d^S_t(T_A, T_B), \tag{4}$$

where $d^M_{t-4:t}(T_A, T_B) = (u_A^{t-4:t} - u_B^{t-4:t})^2 + (v_A^{t-4:t} - v_B^{t-4:t})^2$, and $d^S_t(T_A, T_B) = \| q^t_A - q^t_B \|$. Note that both metrics can be computed incrementally as follows $d_{1:t}(T_A, T_B) = max(d_{1:t-1}\,(T_A, T_B), d_t(T_A, T_B))$. From the n trajectories, we define the two distance matrices $\mathbf{D}^M = [D_{ij} = d^M_{1:t}(T_i, T_j)]$,and $\mathbf{D}^S = [D_{ij} = d^S_{1:t}(T_i, T_j)]$ of all pairwise distances. It follows that we can compute $\mathbf{D}_{1:t}$ incrementally from $\mathbf{D}_{1:t-1}$ and $\varDelta\mathbf{D}$, where $\varDelta\mathbf{D} = \mathbf{D}_{t-1:t}$. We then form matrix $\mathbf{W}$ of affinities as

$$\mathbf{W}_{1:t} = exp(-\mathbf{D}^M_{1:t}/\lambda^2_M) \cdot exp(-\mathbf{D}^S_{1:t}/\lambda^2_S). \tag{5}$$

Given the affinity matrix, a lower dimensional representation is computed using Laplacian Eigenmaps [1]. Let D be the $n \times n$ matrix with elements $D_{ii} = \sum_j W_{ij}$. Laplacian eigenmaps is obtained by an eigen decomposition of the normalized Laplacian $\mathbf{\Gamma}^T\mathbf{\Lambda}\mathbf{\Gamma} = \mathbf{D}^{\frac{-1}{2}}(\mathbf{D} - \mathbf{W})\mathbf{D}^{\frac{-1}{2}}$ and keeping the eigenvectors $v_0, \ldots, v_m$ corresponding to the $m+1$ smallest eigenvalues and then ignoring v_0. The eigenvectors then defines the coordinates of the trajectories in a lower dimensional space.

This representation of the trajectories has two advantages. First, the distance of non-overlapping trajectories can be measured in the embedding space. Second, if part of an object goes out of view, its trajectories in the embedding space remain valid and can be used to enforce the existence of a cluster at that location.

For the first frame, each trajectory is assigned a cluster number by fitting a GMM of R components. In subsequent frames, after computing the new embedding coordinates, we associate trajectories extending from the previous frame with their old cluster assignment. Given this assignment, the parameters of a new GMM model can be estimated and optimized by running a few iterations of the EM algorithm on the whole set of trajectories. The intuition is that as the distance matrix is updated and the new embedding computed, the spatial relationship of trajectories in the embedding will not change abruptly. Due to the incremental nature of the algorithm, the good initialization prevents the algorithm from being stuck in local minima. The result of this step is a set of trajectories with their associated cluster labels.

As the embedding space is updated, the number of clusters may also need to be updated. This happens when an object enters or exits the field of view. During the EM iterations if a cluster ends up with zero trajectories, we simply remove the cluster and decrease the number of clusters by 1. To handle increasing the number of clusters, the probability of each trajectory is computed given the GMM parameters. If the number of trajectories with low probability is more than a threshold, we assign these trajectories to a new cluster and perform EM until convergence.

5 Figure/Ground Labeling

Due to the over-clustering of the embedding space, clusters obtained from the incremental trajectory clustering represents a part of an object or a part of the background. Labeling these clusters with foreground or background labels is, therefore, essential for the rest of the approach.

The problem of figure/ground separation is well studied in psychology of vision. In the case of a single image, the definition of figure/ground can be quite ambiguous. Among the most important factors psychologist studies pointed out to determine figure/ground are: Surroundedness, Size , Orientation, Contrast, Symmetry, Convexity, and Parallelism of the contour. Dynamic figure/ground processing seems to be far less ambiguous compared to single image figure/ground processing. The proposed approach uses motion grouping based on trajectory analysis (common fate), compactness, surroundedness, and spatial closeness. By combining multiple cues, the approach is much more robust than previous approaches which typically assumes that background is the cluster with largest number of trajectories.

Formally, an energy function is defined over labellings $L = \{l_1, \dots l_R\}$, where $l_i \in \{0, 1\}$ (0≡foreground, 1≡background) is the label given for each cluster. The energy function encapsulates the evidence from multiple cues.

$$E_l(L) = \alpha_C \sum_i \phi_C(l_i) + \alpha_A \sum_{(i,j)} \phi_A(l_i, l_j) + \alpha_B \sum_{(i,j)} \phi_B(l_i, l_j) + \alpha_S \phi_S(L)$$

The first term of the energy function $\phi_C(l_i) = (1 - l_i) \cdot (max(\frac{var(x)}{var(y)}, \frac{var(y)}{var(x)}) - 1.5)$ is a unary potential that measures the compactness of each cluster. Foreground objects are more likely to be elongated in the horizontal or vertical direction while background clusters are more spread. The remaining pairwise potentials are only defined for clusters with trajectories that are spatially close in the frame. We define a pairwise potential $\phi_A(l_i, l_j) = -l_i l_j \xi_{Affine} + (l_i(1 - l_j) + l_j(1 - l_i))\xi_{Affine}$ which encourages affine motion compatibility between two clusters with background labels and discourages compatibility between foreground and background clusters. The affinity term $\xi_{Affine} = exp(-var(AffErr))$ is computed by estimating an affine subspace out of the trajectories in both clusters and then measuring the variance of the error $AffErr$ in projecting these trajectories on the subspace. For this computation we only use trajectories over a small window of frames. $\phi_B(l_i, l_j) = -l_i l_j \xi_{Embed} + (l_i(1-l_j) + l_j(1-l_i))\xi_{Embed}$ tests for the existence of a clear boundary in the embedding space by penalizing distant clusters labeled as background, and close foreground and background clusters. The affinity ξ_{Embed} is defined as $exp(-min_{\forall x_i, x_j} \|x_i - x_j\|)$, where x_i is an embedding coordinate of a trajectory in cluster i and x_j is an embedding coordinate of a trajectory in cluster j. Finally, $\phi_S(L)$ computes a measure surroundedness of the foreground. Let F, and B denote the set of points belonging to the foreground and background clusters respectively. Then $\phi_S(L) = 1 - \frac{|F \in ConvexHull(B)|}{|F|}$. $\alpha_C, \alpha_A, \alpha_B, \alpha_S$ are coefficients the determine the relative importance of each term. Since the number of clusters is typically small < 10, the optimal assignment for the labeling can be found by evaluating all possible assignments and finding the minimum.

6 Motion Estimation and Bayesian Filtering

Motion Estimation

Given the sparse trajectories, we estimate the motion model for each layer by the marginal posterior probability $p(m^i_{k,t} | p_{j,t} : j = 1 \ldots M, l_{j,t} = k)$. However, not all pixels in a layer are associated with a trajectory and, without any assumption, inferring the motion is therefore an ill-posed problem. In reality however, background objects are rigid and foreground objects are articulated. This implies the motion of nearby pixels to be smooth.

For each layer, we construct a pairwise MRF with a set of vertices $\mathcal{V} = \{m^j_{k,t} : j = 1, \ldots, N\}$. The set of edges $\mathcal{E} = \{(i,j) : j \in \mathcal{N}(i)\}$ represents pairwise neighborhood relationships on a grid structure defined over the image. The joint posterior probability with respect to $M_{k,t}$ is given by

$$p(M_{k,t} | P_t) \propto \prod_{(i,j) \in \mathcal{E}} \Phi(m^i_{k,t}, m^j_{k,t}) \prod_{i : i = q_{j,t}} \Psi(m^i_{k,t}, P_{j,t}), \tag{6}$$

where

$$\Phi(m^i_{k,t}, m^j_{k,t}) = \mathcal{N}(m^i_{k,t} - m^j_{k,t} | 0, \Sigma_m), \quad \Psi(m^i_{k,t}, P_{j,t}) = \mathcal{N}(m^i_{k,t} | w_{j,t}, \Sigma_p) \tag{7}$$

Σ_m, Σ_p are two bandwidth matrix that represent the strength of the relationship between neighboring pixels and covariance of the observed motion vectors respectively. Since both unary and pairwise potentials are gaussians, this is an instance of Gaussian Belief Propagation (GaBP)[23]. It follows that the joint distribution can be written as $p(M_{k,t} | P_t) \propto e^{-\frac{1}{2} m^T V m + m^T b}$, where $m = [m^1_u, m^1_v \ldots, \ m^N_u, m^N_v]$ is the vector of random variables and V, b are the inverse covariance matrix and shift vector respectively. It is straightforward to write down V, b from (7). The marginal posterior probability $p(m^i_{k,t} | P_t) =$

$\mathcal{N}(\mu^i_{k,t}, \Sigma^i_{k,t})$ for $i = 1 \ldots N$, $k = \{b, f\}$ can be obtained in closed form as $\mu^i_{k,t} = \{V^{-1}b\}_i$, $\Sigma^i_{k,t} = \{V^{-1}\}_{ii}$.

Bayesian Filtering
Our ultimate goal is to estimate L_t given all observations. This can be accomplished by Bayesian filtering. In the first step we predict the appearance models and labels given the motion models. In the second step we update the prediction using the most recent observation. For brevity we will drop the dependence on past observations for the remainder of this section.

Prediction. We first predict the appearance model given the motion models. Since our model satisfies the Markov assumption, only the marginal appearance model from the previous time step is needed. The prediction step for the appearance models can be expressed as

$$p(a^i_{k,t}|P_t) = \sum_{j=1}^{N}[\sum_{m^i_{k,t} \in \mathbb{N}^2} p(a^i_{k,t}|m^i_{k,t}, a^j_{k,t-1})p(m^i_{k,t}|P_t)]\; p(a^j_{k,t-1}) \tag{8}$$

where (8)[1] is derived from marginalizing over the motion model. From (1),(2) we can derive $p(a^i_{k,t}|m^i_{k,t}, a^j_{k,t-1}) = \delta(a^{j(i,k)}_{k,t-1} - a^i_{k,t})$, where$j(i, m^i_{k,t}) = \phi^{-1}(\phi(i) + m^i_{k,t})$. Equation (8) thus becomes

$$p(a^i_{k,t}|P_t) = \sum_{m^i_{k,t} \in \mathbb{N}^2} p(m^i_{k,t}|P_t)p(a^{j(i,m^i_{k,t})}_{k,t-1}), \tag{9}$$

The summation in equation (9) will lead to an exponential increase in the number of samples needed to represent the appearance. Thus by approximating $p(m^i_{k,t}|P_t) = \mathcal{N}(\mu^i_{k,t}, \Sigma^i_{k,t})$ with $p(m^i_{k,t}|P_t) = \delta(\mu^i_{k,t} - m^i_{k,t})$ in (9) we can avoid this problem and obtain $p(a^i_{k,t}|P_t) \approx p(a^{j(i,\mu^i_{k,t})}_{k,t-1})$. Therefore, we use nonparametric density estimation to represent the probability density of the appearance models. Specifically, for each layer, the density of each pixel is represented by a set of N_{KDE} color samples in rgs[2] color space. At any instance, if the color samples for a pixel will exceed N_{KDE}, we discard the oldest color sample. The bandwidth of the KDE is chosen adaptively using the method in [5].

To predict the labels we similarly have

$$p(l^i_t = f|P_t) = \sum_{m^i_{f,t} \in \mathbb{N}^2} p(m^i_{f,t}|P_t)p(l^{j(i,m^i_{f,t})}_{t-1} = f). \tag{10}$$

This summation is evaluated exactly and takes into account the uncertainty in the motion vectors.

Update. In this step the new observation is incorporated by updating the marginal priors of the appearance model and labels. It turns out that if we

[1] Strictly speaking the summation in (8),(9), and (10) contains also an integration over the pixel areas which is neglected here for brevity.

[2] rgs can be computed from RGB by s = R+G+B,r=R/s ,g=G/s.

want to avoid maintaining a joint belief space of labels and appearance models, updating both the appearance and labels is a chicken and egg problem. Given the label at a pixel one can easily update the corresponding appearance model. On the other hand, if the appearance of a pixel is known, marginal posterior probability of the labels can be computed easily.

We address this by taking the former approach. Given the predicted label prior and appearance models a MAP estimate of the labels is inferred. Next the inferred labels are used to update the corresponding appearance model. In practice we have found that this approach yields excellent results while avoiding the complexity of maintaining a joint belief space. Note that the marginal posterior over the labels is maintained for the next prediction stage.

$$p(l_t^i = k|I_t^i, P_t) \propto p(I_t^i|l_t^i = k)p(l_t^i|P_t) = \int_{a_k^i} p(I_t^i|l_t^i, a_{k,t}^i)p(a_{k,t}^i|P_t)\, \mathbf{d}a_{k,t}^i\, p(l_t^i|P_t) \quad (11)$$

Since we previously approximated the posterior probability of motion model (by discarding the uncertainty) in the prediction of the appearance model, the correct appearance can be anywhere in a neighborhood around the current pixel. We therefore replace $p(I_t^i|l_t^i = k)$ in (11) with $\sum_j p(I_t^i|l_t^j) \cdot \mathcal{N}(\phi(j) - \phi(i)|0, \Sigma_{k,t}^i)$, where $\Sigma_{k,t}^i$is the uncertainty in the motion model at pixel i.

A pairwise MRF, defined over a grid structure over the pixels, is used to enforce smoothness on the labels

$$p(L_t|I_t) \propto \prod_{(i,j)\in\mathcal{E}} \Phi(l_t^i, l_t^j) \prod_i \Psi(l_t^i),$$

where $\Phi(l_t^i, l_t^j) = \mathcal{N}(I_t^i - I_t^j|0, \Sigma_l)(l^i l^j + (1-l^i)(1-l^j))$ and$\Psi(l_t^i) = l^i \cdot p(l_t^i|I_t^i, P_t) + (1 - l^i)(1 - p(l_t^i|I_t^i, P_t))$. Σ_l is a bandwidth matrix that represents the strength of the relation ship between neighboring pixels. The globally optimal solution l_{MAP}^i can be found efficiently via graph cuts[2].

Finally, updating the corresponding appearance model is done by simply adding the observed frame pixel color I_t^i to the set of samples of the corresponding pixel in layer l_{MAP}^i.

7 Continuous Initialization

In the first few frames, the appearance models of some pixels will not have the minimum number of samples needed to compute the appearance likelihood. Therefore a method must be devised to initialize the appearance models of these pixels. From the incremental trajectory clustering we obtain a set of sparse labels l_s^j at a subset of the pixels $j \in S$. To propagate these labels we construct a pairwise MRF with a set of vertices $\mathcal{V} = \{l_t^i : i = 1, \ldots, N\}$. The set of edges $\mathcal{E} = \{(i,j) : j \in \mathcal{N}(i)\}$ represents a grid structure defined over pixel neighborhood in the image. The joint posterior probability with respect to L_t is given by

$$p(L_t|I_t) \propto \prod_{(i,j)\in\mathcal{E}} \Phi(l_t^i, l_t^j) \prod_{i\in S} \Psi(l_t^i, l_s^i), \quad (12)$$

where $\Phi(l_t^i, l_t^j) = (l^i(1-l^j) + (1-l^i)l^j) \cdot \mathcal{N}(I_t^i - I_t^j|0, \Sigma_l)$ and $\Psi(l_t^i, l_s^i) = (l^i l_s^i + (1-l^i)(1-l_s^i))$,Σ_l is a bandwidth matrix that represents the strength of the relation ship between neighboring pixels. The MAP estimate can be computed efficiently by Graph Cut [2]. This labeling defines a segmentation of the image into foreground and background pixels. For each pixel which does not have the minimum number of KDE samples, depending on the inferred label l^i of each pixel, the color I^i of the pixel is added to the foreground or background appearance model.

8 Experiments

Feature Tracking
We use LDOF [18] to track dense feature points over pairs of frames. New trajectories are automatically incorporated up to a maximum number of trajectories per frame T^f_{max}. In addition, the total number of trajectories maintained is set to not exceed T^{memory}_{max} after with we drop trajectories with the oldest ending frame number. Note that at any instant of time we do not store the full trajectories but rather maintain a low dimensional representation.

Results. We evaluate our algorithm qualitatively and quantitatively on five challenging sequences[3]. Our results are compared to state-of-the-art algorithms that use dense point trajectories [15], and belief propagation and Bayesian filtering [9][4]. We also compare the results of our approach with and without the label prior. Parameter settings for all experiments are provided in the supplementary materials.

Table 1. Performance comparisons with other methods

	cars1			people1			people2			tennis			drive		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
Ours-1	0.84	**0.99**	**0.91**	0.94	0.85	0.89	0.69	0.88	0.77	0.86	**0.92**	**0.89**	0.55	**0.96**	**0.70**
Ours-2	0.85	0.97	0.90	**0.97**	0.88	0.92	**0.87**	0.88	**0.88**	**0.90**	0.81	0.85	**0.60**	0.67	0.63
[15]	0.63	**0.99**	0.77	0.78	0.63	0.70	0.73	0.83	0.78	0.27	0.83	0.40	0.02	0.66	0.04
[9]	**0.92**	0.84	0.88	0.95	**0.93**	**0.94**	0.85	**0.89**	0.86	-	-	-	-	-	-

The first three videos - cars1, people1 and people2 - comes from the Hopkins 155 dataset [21]. Manually annotated ground truth for a subset of frames, around one every 10 frames, is provided by Brox et al [3]. A characteristic of these sequences is that they are short and the objects are always on motion. These sequences are used as a benchmark to compare with other approaches.

[3] Project page: `http://www.cs.rutgers.edu/~elqursh/projects/bsmc/`

[4] We have requested the code and results from the authors but we did not receive a response, results for their approach are reported verbatim from the paper.

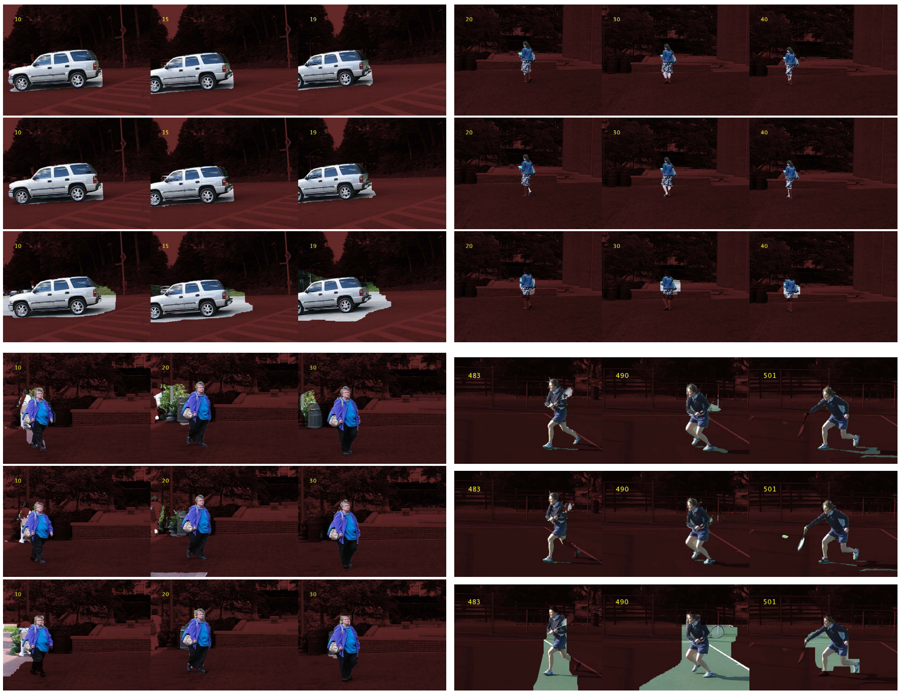

Fig. 3. Results on the cars1, people1, people2, and tennis sequences using our method (First row), our method without label prior (Second row), using [15](Third row). For visualization purposes, background regions are darkened while foreground regions maintain their colors. (Best viewed in color).

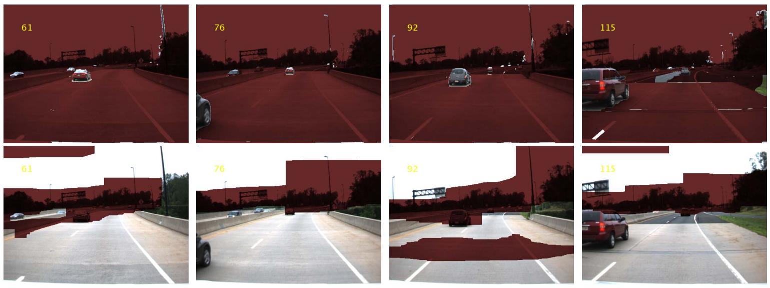

Fig. 4. Results for drive 1 sequence using our method (First row) and [15](Second row). [15] fails on this sequence since it highly deviates from the orthographic projection assumption. Our approach successfully segments the new car as soon as it enters the fields of view at frame 76. Notice also, how cars approaching in the opposite direction are successfully segmented.

Table 1. shows quantitative comparison on the first three benchmark sequences[5]. The row labeled ours-1 is our approach with the label prior, while ours-2 is our approach without the label prior. Without the label prior we do not rely on the predicted labels in inferring the new labels. We compare our methods to [15] and [9]. On cars1, people2 our approach ranks 1st in F1-score while in people1 we rank a close second. The reason our approach performs worser on the people1 sequence is that with at most 2000 trajectories there are no trajectories from the hips downwards and thus the motion of the legs is not captured by the trajectories. As can be seen from the first row of Fig. 3, after initialization the foreground appearance model captures the top portion and it takes a few more frames for it to recover from this error. Figs 3 shows qualitative results on these sequences.

To evaluate the performance of the proposed approach on long sequences with fast motion, two other sequences are used - tennis and drive. The tennis sequence is 466 frames long and also comes with ground truth from [3]. The tennis player pauses at times to wait for the ball, while at others moves fast to intercept it. Due to the fast motion and homogeneous ground color not all objects have trajectory points as seen in Fig.1(b). Finally, the drive sequence is 456 frames and was manually annotated with ground truth. This sequence is challenging since cars keep entering and exiting the field of view at different points in the video and due to the forward motion. Figs. 3, 4 shows qualitative results on the tennis and drive sequences. Thanks to the accurate background model our method is able to capture the cars on the other side of the road Fig. 4. More results are are available in the supplemental materials.

9 Conclusion

We introduce a methodthat accurately models appearance and motion to achieve robust moving camera background subtraction. Unlike previous approaches, it merges the best of both worlds, long term trajectories to accurately model long term motion dependencies and a Bayesian filtering framework to reason about pixel level appearance models for foreground and background regions. This is achieved in a online framework without sacrificing the accuracy and with a constant processing time per frame. The output is not only the final segmentation per frame but in addition the pixel based background and foreground model. Such models, we believe, are essential for high level reasoning. We evaluated the approach on benchmark sequences as well as on two challenging sequences and demonstrated that the method produces superior results.

References

1. Belkin, M., Niyogi, P.: Laplacian Eigenmaps for Dimensionality Reduction and Data. Neural Computation 15, 1373–1396 (2003)

[5] Let TP, FP, and FN denote true positives, false positives, and false negatives. Then $Prec = TP/(TP+FP)$, $Rec = TP/(TP+FN)$, $F1 = 2\cdot(Prec\cdot Rec)/(Prec+Rec)$.

2. Boykov, Y., Funka-Lea, G.: Graph Cuts and Efficient N-D Image Segmentation. International Journal of Computer Vision 70(2), 109–131 (2006)
3. Brox, T., Malik, J.: Object Segmentation by Long Term Analysis of Point Trajectories. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 282–295. Springer, Heidelberg (2010)
4. Costeira, J., Kanade, T.: A multi-body factorization method for motion analysis. In: ICCV, pp. 1071–1076 (1995)
5. Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proceedings of the IEEE 90(7), 1151–1163 (2002)
6. Elhamifar, E., Vidal, R.: Sparse subspace clustering. In: CVPR, pp. 2790–2797 (June 2009)
7. Irani, M., Rousso, B., Peleg, S.: Computing occluding and transparent motions. International Journal of Computer Vision 12(1), 5–16 (1994)
8. Kanatani, K.: Motion segmentation by subspace separation and model selection. In: ICCV, vol. 2, pp. 586–591 (2001)
9. Kwak, S., Lim, T., Nam, W., Han, B., Hee, J.: Generalized Background Subtraction Based on Hybrid Inference by Belief Propagation and Bayesian Filtering. In: ICCV (2011)
10. Mittal, A., Paragios, N.: Motion-based background subtraction using adaptive kernel density estimation. In: CVPR, vol. 2, pp. 302–309. IEEE (2004)
11. Ochs, P., Brox, T.: Object Segmentation in Video: A Hierarchical Variational Approach for Turning Point Trajectories into Dense Regions. In: ICCV (2011)
12. Pawan Kumar, M., Torr, P.H.S., Zisserman, A.: Learning Layered Motion Segmentations of Video. IJCV 76(3), 301–319 (2007)
13. Rao, S.R., Tron, R.: Motion Segmentation via Robust Subspace Separation in the Presence of Outlying, Incomplete, or Corrupted Trajectories. In: CVPR (2008)
14. Rowe, S., Blake, A.: Statistical mosaics for tracking. Image and Vision Computing 14(8), 549–564 (1996)
15. Sheikh, Y., Javed, O., Kanade, T.: Background Subtraction for Freely moving cameras. In: ICCV (2009)
16. Shi, J.: Motion segmentation and tracking using normalized cuts. In: ICCV, vol. 32(10), pp. 1832–1845 (October 2002)
17. Stauffer, C.: Learning patterns of activity using real-time tracking. PAMI 22(8), 747–757 (2000)
18. Sundaram, N., Brox, T., Keutzer, K.: Dense Point Trajectories by GPU-Accelerated Large Displacement Optical Flow. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 438–451. Springer, Heidelberg (2010)
19. Tao, H., Sawhney, H.S., Kumar, R.: Object Tracking with Bayesian Estimation of Dynamic Layer Representations. PAMI 24(1), 75–89 (2002)
20. Torr, P.H.S.: Outlier detection and motion segmentation. PhD thesis, University of Oxford (1995)
21. Tron, R., Vidal, R.: A Benchmark for the Comparison of 3-D Motion Segmentation Algorithms. In: CVPR (2007)
22. Weiss, Y.: Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In: CVPR, pp. 520–526 (1997)
23. Weiss, Y., Freeman, W.T.: Correctness of belief propagation in Gaussian graphical models of arbitrary topology. Neural Computation 13(10), 2173–2200 (2001)
24. Wren, C., Azarbayejani, A., Darrell, T., Pentland, A.: Pfinder: real-time tracking of the human body. PAMI 19(7), 780–785 (1997)
25. Xiao, J.: Accurate Motion Layer Segmentation and Matting. In: CVPR, pp. 698–703 (2005)