The Dependent Dirichlet Process Mixture of Objects for Detection-free Tracking and Object Modeling

Willie Neiswanger
Machine Learning Department
Carnegie Mellon University

Frank Wood
Department of Engineering
Oxford University

Eric P. Xing
Machine Learning Department
Carnegie Mellon University

Abstract

This paper explores how to find, track, and learn models of arbitrary objects in a video without a predefined method for object detection. We present a model that localizes objects via unsupervised tracking while learning a representation of each object, avoiding the need for pre-built detectors. Our model uses a dependent Dirichlet process mixture to capture the uncertainty in the number and appearance of objects and requires only spatial and color video data that can be efficiently extracted via frame differencing. We give two inference algorithms for use in both online and offline settings, and use them to perform accurate detection-free tracking on multiple real videos. We demonstrate our method in difficult detection scenarios involving occlusions and appearance shifts, on videos containing a large number of objects, and on a recent human-tracking benchmark where we show performance comparable to state of the art detector-based methods.

1 Introduction

Algorithms for automated object detection and tracking in video have found application in a wide range of fields, including robotic vision, cell tracking, sports analysis, video indexing, and video surveillance [28, 33]. The goal of these algorithms is to find the sequences of positions held by each object of interest in a video. A majority of modern methods require a pre-trained object detector or make use of prior knowledge about the objects' physical characteristics (such as their color or shape) to perform detection [9]. Often, these methods will apply the detector in each frame of a video, and then use the detection results in tracking or data association algorithms. Other algorithms use heuristics to find, or require manual initialization of, object positions and then search for similar image patches in consecutive frames to perform tracking [24]. Both techniques require some predefined detection strategy for each type of object they intend to find and track.

When the objects to be tracked have highly variable appearance, if one wishes to track many different types of objects, or if one simply does not know the types of objects in advance, it is often hard to find a suitable detection strategy [5]. Furthermore, common video conditions such as variable lighting, low quality images, non-uniform backgrounds, and object occlusions can all reduce detection accuracy [31].

Cases such as these, where it is difficult to construct an object detector in advance, prompt the need for a method to automatically localize and track arbitrary objects. Some methods towards this end have involved background subtraction and blob tracking, which segment foreground patches to localize objects, and optical flow-based tracking, which separate objects based on their relative motion. Both have trouble consistently and accurately segmenting objects and tracking through occlusion [29, 6]. A recent work introduced the term "detection-free tracking" for this task, and proposed a method based on spectral clustering of trajectories [18].

Bayesian models have also been employed to capture the components of a video, and a number of recent works have incorporated nonparametric Bayesian priors for finding the patterns of motion in scenes [30, 17]. However, there has been little work towards building Bayesian models of arbitrary objects in order to perform detection-free tracking.

In this paper, we develop a nonparametric Bayesian model for jointly learning a representation of each object and performing unsupervised tracking, thereby allowing for accurate localization of arbitrary objects. We combine a dependent Dirichlet process mixture with object and motion models to form the dependent Dirichlet process mixture of objects (DDPMO). The advantages of our model are that it can (a) accurately localize and track arbitrary video objects in a fully unsupervised fashion, (b) jointly learn a time-varying model for each object and use these models to

Appearing in Proceedings of the 17^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33.

increase the localization/tracking performance, (c) infer a distribution over the number of distinct objects present in a video, (d) incorporate a model for the motion of each object, and (e) begin tracking as objects enter the video frame, stop when they exit, and track through periods of partial or full occlusion.

2 Dependent Dirichlet Process Mixture of Objects

To find and track arbitrary video objects, the DDPMO models spatial and color features that are extracted as objects travel within a video scene (described in Section 2.1). The model isolates independently moving video objects and learns object models for each that capture their shape and appearance. The learned object models allow for tracking through occlusions and in crowded videos. The unifying framework is a dependent Dirichlet process mixture, where each component is a (time-varying) object model. This setup allows us to estimate the number of objects in a video and track moving objects that may undergo changes in orientation, perspective, and appearance.

2.1 Preliminaries

Dependent Dirichlet process prior. Dirichlet process (DP) priors for component weights in mixture models have long been used as nonparametric Bayesian tools to estimate the number of clusters in data [3]. Dependent Dirichlet process (DDP) mixtures extend this by allowing cluster parameters to vary with some covariate [23]. In our case, a DDP object mixture lets us estimate, and capture the uncertainty in, the number of objects while modeling their time-varying parameters.

A DDP known as a generalized Polya urn (GPU) [11] has the desired properties that, when used in a mixture model, clusters can be created and die off and cannot merge or split. In this model, the n^{th} data point at time t, $\mathbf{x}_{t,n}$, has an assignment $c_{t,n}$ to a cluster $k \in \{1, \ldots, K_{t,n}\}$ (where $K_{t,n}$ denotes the total number of assigned clusters after reaching $\mathbf{x}_{t,n}$). Each assignment increases the cluster's size $m^k_{t,n}$ by one. After each time step, cluster sizes may decrease when observations are uniformly "unassigned" in a deletion step. The generative process for the GPU, at each time step t, is

1. For $k = 1, \ldots, K_{t-1,N_{t-1}}$
   (a) Draw $\Delta m^k_{t-1} \sim \text{Binom}(m^k_{t-1,N_{t-1}}, \rho)$
   (b) Set $m^k_{t,0} = m^k_{t-1,N_{t-1}} - \Delta m^k_{t-1}$
2. For $n = 1, \ldots, N_t$
   (a) Draw $c_{t,n} \sim \text{Cat}\left(\frac{m^1_{t,n-1}}{\alpha+\sum_k m^k_{t,n-1}}, \ldots, \frac{m^{K_{t,n-1}}_{t,n-1}}{\alpha+\sum_k m^k_{t,n-1}}, \frac{\alpha}{\alpha+\sum_k m^k_{t,n-1}}\right)$
   (b) If $c_{t,n} \leq K_{t,n-1}$: set $m^{c_{t,n}}_{t,n} = m^{c_{t,n}}_{t,n-1} + 1$, $m^{\setminus c_{t,n}}_{t,n} = m^{\setminus c_{t,n}}_{t,n-1}$, and $K_{t,n} = K_{t,n-1}$
   (c) If $c_{t,n} > K_{t,n-1}$: set $m^{c_{t,n}}_{t,n} = 1$, $m^{\setminus c_{t,n}}_{t,n} = m^{\setminus c_{t,n}}_{t,n-1}$, and $K_{t,n} = K_{t,n-1} + 1$.

where Cat is the categorical distribution, $m^{\setminus c_{t,n}}_{t,n}$ is the set $\{m^1_{t,n}, \ldots, m^{K_{t,n}}_{t,n}\} \setminus \{m^{c_{t,n}}_{t,n}\}$, Binom is the binomial distribution, α is the DP concentration parameter, and ρ is a deletion parameter that controls temporal dependence of the DDP. We will refer to this process as GPU(α, ρ).

Data. At each frame t, we assume we are given a set of N_t foreground pixels, extracted via some background subtraction method (such as those detailed in [33]). These methods primarily segment foreground objects based on their motion relative to the video background. For example, an efficient method applicable for stationary videos is frame differencing: in each frame t, one finds the pixel values that have changed beyond some threshold, and records their positions $\mathbf{x}^s_{t,n} = (x^{s_1}_{t,n}, x^{s_2}_{t,n})$. In addition to the position of each foreground pixel, we extract color information. The spectrum of RGB color values is discretized into V bins, and the local color distribution around each pixel is described by counts of surrounding pixels (in an $m \times m$ grid) that fall into each color bin, denoted $\mathbf{x}^c_{t,n} = (x^{c_1}_{t,n}, \ldots, x^{c_V}_{t,n})$. Observations are therefore of the form

$$\mathbf{x}_{t,n} = (\mathbf{x}^s_{t,n}, \mathbf{x}^c_{t,n}) = (x^{s_1}_{t,n}, x^{s_2}_{t,n}, x^{c_1}_{t,n}, \ldots, x^{c_V}_{t,n}) \quad (1)$$

Examples of spatial pixel data extracted via frame differencing are shown in Figure 1 (a)-(g).

2.2 DDPMO

Our object model $\text{F}(\theta^k_t)$ is a distribution over pixel data, where θ^k_t represents the parameters of the k^{th} object at time t. We wish to keep our object model general enough to be applied to arbitrary video objects, but specific enough to learn a representation that can aid in tracking. In this paper, we model each object with

$$\mathbf{x}_{t,n} \sim \text{F}(\theta^k_t) = \text{Normal}(\mathbf{x}^s_{t,n}|\mu_t, \Sigma_t)\text{Mult}(\mathbf{x}^c_{t,n}|\delta_t) \quad (2)$$

where object parameters $\theta_t = \{\mu_t, \Sigma_t, \delta_t\}$, and $\sum_{j=1}^V \delta^j_t = 1$. The object model captures the objects' locus and extent with the multivariate Gaussian and color distribution with the multinomial. We demonstrate in Section 4 that this representation can capture the physical characteristics of a wide range of objects while allowing objects with different shapes, orientations, and appearances to remain isolated during tracking.

We would also like to model the motion of objects. Assuming as little as possible, we take each object's parameters θ^k_t to be a noisy version of the previous parameters θ^k_{t-1} (if the object existed at the previous time step) and define

$$\theta^k_t|\theta^k_{t-1} \sim \begin{cases} \text{T}(\theta^k_{t-1}) \text{ if } k \leq K_{t-1,N_{t-1}} \\ \mathbb{G}_0 \text{ if } k > K_{t-1,N_{t-1}} \end{cases} \quad (3)$$

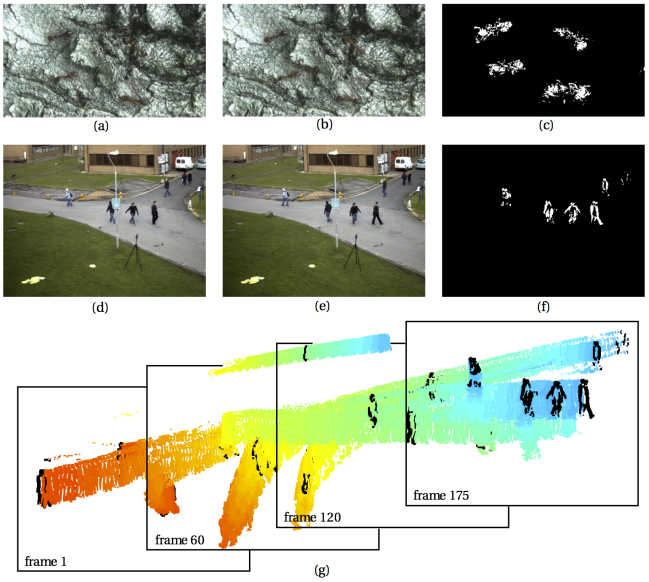


Figure 1: (a - f) Two pairs of consecutive frames and the spatial observations $\mathbf{x}^s_{t,n}$ extracted by taking the pixel-wise frame difference between each pair. (g) The results of frame differencing over a sequence of images (from the PETS2010 dataset).

where T denotes a transition kernel, the $k > K_{t-1,N_{t-1}}$ case is when a new cluster has been created at time t, and $\mathbb{G}_0$ is the base distribution of the dependent Dirichlet process, which represents the prior distribution over object parameters. We define $\mathbb{G}_0$ to be

$$\mathbb{G}_0(\theta^k_t) = \text{NiW}(\boldsymbol{\mu}^k_t, \Sigma^k_t | \boldsymbol{\mu}_0, \kappa_0, \nu_0, \Lambda_0)\text{Dir}(\delta^k_t | q_0) \quad (4)$$

where NiW denotes the normal-inverse-Wishart distribution and Dir denotes the Dirichlet distribution; these act as a conjugate prior to the object model. We can therefore write the generative process of the DDPMO as, for each time step $t = 1, \ldots, T$:

1. Draw $\{c_{t,1:N_t}, K_{t,N_t}, m^{1:K_{t-1,N_{t-1}}}_{t,0}\} \sim \text{GPU}(\alpha, \rho)$
2. For $k = 1, \ldots, K_{t,N_t}$:
   draw $\theta^k_t \sim \begin{cases} \text{T}(\theta^k_{t-1}) \text{ if } k \leq K_{t-1,N_{t-1}} \\ \mathbb{G}_0(\boldsymbol{\mu}_0, \kappa_0, \nu_0, \Lambda_0, q_0) \text{ if } k > K_{t-1,N_{t-1}} \end{cases}$
3. For $n = 1, \ldots, N_t$: draw $\mathbf{x}_{t,n} \sim \text{F}(\theta^{c_{t,n}}_t)$

where the notation $c_{1,1:N_1} = \{c_{1,1}, \ldots, c_{1,N_1}\}$. A graphical model for the DDPMO is shown in Figure 2.

To meet technical requirements of the GPU, the transition kernel T must satisfy

$$\int \mathbb{G}_0(\theta^k_{t-1})\text{T}(\theta^k_t | \theta^k_{t-1}) d\theta^k_{t-1} = \mathbb{G}_0(\theta^k_t) \quad (5)$$

or, equivalently, its invariant distribution must equal the base distribution [19]. One way to satisfy this while providing

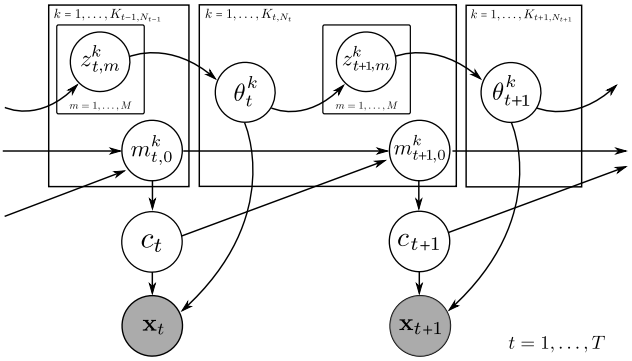


Figure 2: Graphical model of the dependent Dirichlet process mixture of objects (DDPMO). All observations at time t are denoted as $\mathbf{x}_t$ and their assignments as c_t.

a reasonable transition kernel is to introduce a set of M auxiliary variables $\mathbf{z}^k_t = (z^k_{t,1}, \ldots, z^k_{t,M})$ for cluster k at time t such that

$$P(\theta^k_t | \theta^k_{t-1}) = \int P(\theta^k_t | \mathbf{z}^k_t) P(\mathbf{z}^k_t | \theta^k_{t-1}) d\mathbf{z}^k_t \quad (6)$$

With this addition, object parameters do not directly depend on their values at a previous time, but are instead dependent through an intermediate sequence of variables. This allows the cluster parameters at each time step to be marginally distributed according to the base distribution $\mathbb{G}_0$ while maintaining simple time varying behavior. We can therefore sample from the transition kernel using $\theta^k_t \sim T(\theta^k_{t-1}) = T_2 \circ T_1(\theta^k_{t-1})$, where

$$\begin{aligned} z^k_{t,1:M} &\sim T_1(\theta^k_{t-1}) \\ &= \text{Normal}(\boldsymbol{\mu}^k_{t-1}, \Sigma^k_{t-1})\text{Mult}(\delta^k_{t-1}) \quad (7) \\ \mu^k_t, \Sigma^k_t, \delta^k_t &\sim T_2(z^k_{t,1:M}) \\ &= \text{NiW}(\mu_M, \kappa_M, \nu_M, \Lambda_M)\text{Dir}(q_M) \quad (8) \end{aligned}$$

where $\mu_M, \kappa_M, \nu_M, \Lambda_M$ and q_M are posterior NiW and Dir parameters, given the auxiliary variables $z_{t,1:M}$ (formulas given in Section 3.1.1).

3 Inference

We describe two inference algorithms for the DDPMO: sequential Monte Carlo (SMC) with local Gibbs iterations, and Particle Markov Chain Monte Carlo (PMCMC).

3.1 Sequential Monte Carlo

We first derive an SMC (particle filter) inference algorithm where we draw samples from a proposal distribution by iterating through local Gibbs updates (detailed in Section 3.1.1). SMC allows us to make a single pass through the data and draw posterior samples in an online fashion.

Algorithm 1 SMC for the DDPMO

Input: Extracted pixel data $\{\mathbf{x}_{1,1:N_1}, \ldots, \mathbf{x}_{T,1:N_T}\}$, number of particles L, number of local Gibbs iterations S

Output: Posterior samples $\left\{\theta_1^{1:K_{1,N_1}}, \ldots, \theta_T^{1:K_{T,N_T}}\right\}^{(1:L)}$ of the object model parameters

for $t = 1$ **to** T **do**
  for $l = 1$ **to** L **do**
    for iter $= 1$ **to** S **do**
      Sample $(c_{t,1:N_t})^{(l)} \sim Q_1$ and $(\theta_t^{1:K_{t,N_t}})^{(l)} \sim Q_2$
    end for
    for $k = 1$ **to** K_{t,N_t} **do**
      Sample $(\Delta m_t^k)^{(l)} \sim \text{Binom}((m_{t,N_t}^k)^{(l)}, \rho)$
      Set $(m_{t+1,0}^k)^{(l)} = (m_{t,N_t}^k)^{(l)} - (\Delta m_t^k)^{(l)}$
      Sample $(z_{t+1,1:M}^k)^{(l)} \sim T_1((\theta_t^k)^{(l)})$
    end for
    Compute particle weight $\tilde{w}_t^{(l)}$
  end for
  Normalize particle weights and resample particles
end for

3.1.1 Local Gibbs Updates

We perform Gibbs sampling on the assignments and object parameters (at a given t) to draw SMC proposals; this allows for the proposal of well-mixed samples given newly introduced data in a particular frame. For an assignment $c_{t,n}$, we can compute a value proportional to the posterior for each possible assignment value $1, \ldots, K_{t,n}$, and then sample from the resulting categorical distribution (after normalizing). The first proposal distribution Q_1 is the probability of an assignment $c_{t,n}$ given current cluster sizes, cluster parameters, and concentration parameter α, written

$$\begin{aligned} Q_1\left(c_{t,n}|m_{t,n-1}^{1:K_{t,n-1}}, \theta_t^{1:K_{t,n-1}}, \alpha\right) \propto \\ \text{Cat}(m_{t,n-1}^1, \ldots, m_{t,n-1}^{K_{t,n-1}}, \alpha) \\ \times \begin{cases} \text{F}(\mathbf{x}_{t,n}|\theta_t^{c_{t,n}}) & \text{if } c_{t,n} \leq K_{t,n-1} \\ \int P(\mathbf{x}_{t,n}|\theta)\mathbb{G}_0(\theta)d\theta & c_{t,n} > K_{t,n-1} \end{cases} \end{aligned} \tag{9}$$

where we set the number of clusters $K_{t,n}$ and their sizes $m_{t,n}^{1:K_{t,n}}$ appropriately as each $c_{t,n}$ is assigned, and assume $K_{1,0} = 0$ for consistency at $t = 1$. The integral in the case of a new cluster ($k > K_{t,n-1}$) has an analytic solution

$$\begin{aligned} \int P(\mathbf{x}_{t,n}|\theta)\mathbb{G}_0(\theta)d\theta = \\ t_{\nu_0-1}\left(\mathbf{x}_{t,n}^s \mid \mu_0, \frac{\Lambda_0(\kappa_0+1)}{\kappa_0(\nu_0-1)}\right) \\ \times \prod_{j=1}^{V} \frac{\Gamma(\mathbf{x}_{t,n}^c)}{\Gamma(q_0)} \times \frac{\Gamma(\sum_{j=1}^V q_0)}{\Gamma(\sum_{j=1}^V \mathbf{x}_{t,n}^c)} \end{aligned} \tag{10}$$

where t_{ν_0-1} denotes the multivariate t-distribution with $\nu_0 - 1$ degrees of freedom, where we follow the three-value parameterization [21], and Γ denotes the gamma function.

The conjugacy of appearance model and transition kernel allow us to sample from the second proposal distribution Q_2, which is the posterior distribution over the object parameters given current observations, auxiliary variables, and previous time object parameters, written

$$\begin{aligned} Q_2(\theta_t^k|\theta_{t-1}^k, \mathbf{x}_{t,1:N_t}^k, z_{t,1:M}^k) = \\ \text{F}(\mathbf{x}_{t,1:N_t}^k|\theta_t^k)T_2(\theta_t^k|z_{t,1:M}^k) \\ = \text{NiW}(\mu_t^k, \Sigma_t^k|\mu_N, \kappa_N, \nu_N, \Lambda_N) \\ \times \text{Dir}(\delta_t^k|q_N) \end{aligned} \tag{11}$$

where $\mathbf{x}_{t,1:N_t}^k = \{x_{t,n} \in x_{t,1:N_t} | c_{t,n} = k\}$ and the parameters for the NiW and Dir distributions are given when $\mathbf{x}_{t,1:N_t}^k$ and $z_{t,1:M}^k$ are taken to be the "observations" in the following posterior updates

$$\kappa_N = \kappa_0 + N \tag{12}$$

$$\nu_N = \nu_0 + N \tag{13}$$

$$\mu_N = \frac{\kappa_0}{\kappa_0 + N}\mu_0 + \frac{N}{\kappa_0 + N}\overline{\mathbf{x}}^s \tag{14}$$

$$\Lambda_N = \Lambda_0 + S_{\mathbf{x}^s} \tag{15}$$

$$q_N = q_0 + \sum_{i=1}^{N} \mathbf{x}_i^c \tag{16}$$

where N is the number of observations, $\{\mu_0, \kappa_0, \nu_0, \Lambda_0\}$ are the NiW prior parameters, q_0 is the Dir prior parameter, $\mathbf{x}^s$ and $\mathbf{x}^c$ respectively denote the spatial and color features of the observations, and $\overline{\mathbf{x}}$ and $S_{\mathbf{x}}$ respectively denote the sample mean and sample covariance of the observations.

3.1.2 Particle Weights

At each time, the particle weights are set to be

$$\tilde{w}_t^{(l)} = \frac{P\left((c_{t,1:N_t})^{(l)}, (\theta_t^{1:K_{t,N_t}})^{(l)}, \mathbf{x}_{t,1:N_t}|\Lambda\right)}{P\left((c_{t,1:N_t})^{(l)}, (\theta_t^{1:K_{t,N_t}})^{(l)}|\Lambda\right)} \tag{17}$$

where we've defined

$$\Lambda = \{(\theta_{t-1}^{1:K_{t-1,N_{t-1}}})^{(l)}, (m_{t,0}^{1:K_{t-1,N_{t-1}}})^{(l)}\} \tag{18}$$

Note that the numerator decomposes into

$$\begin{aligned} P\left(\mathbf{x}_{t,1:N_t}|(c_{t,1:N_t})^{(l)}, (\theta_t^{1:K_{t,N_t}})^{(l)}\right) \\ \times P\left((c_{t,1:N_t})^{(l)}, (\theta_t^{1:K_{t,N_t}})^{(l)}|\Lambda\right) \end{aligned} \tag{19}$$

which can be computed using the DDPMO local probability equations defined in Section 2.2, and the denominator can be computed using equations 9 and 11. After the particle weights are computed, they are normalized; particles are then redrawn based on their normalized weights in a multinomial resampling procedure [15].

3.1.3 Computational Cost

Assume N extracted pixels per frame, T frames, L particles, M auxiliary variables, S local Gibbs iterations, and fewer than K sampled objects ($K_{T,N_T}^{(1:L)} \leq K$). In the SMC inference algorithm, each local Gibbs iterations is $\mathrm{O}(KN + M)$ and evaluating each particle weight is $\mathrm{O}(K + N)$; the SMC algorithm therefore scales as $\mathrm{O}(TL(K(SN + M) + SM + N))$. If we neglect the number of auxiliary variables M, as we can usually fix this at a small value, the algorithm scales as $\mathrm{O}(TLKSN)$. We have empirically found that an SMC implementation in MATLAB, while not tuned for speed, usually requires 4-20 seconds for every 1 second of video, depending on the number of objects (after frame-rate has been subsampled to approximately 3 images/second in all cases). It is not unreasonable to believe that this could be scaled to real time tracking, given parallel computation and efficient image processing.

3.2 Particle Markov Chain Monte Carlo

SMC provides an efficient, online method for posterior inference, but can suffer from degeneracy; notably, a large majority of the returned particles correspond to a single, non-optimal tracking hypothesis. Ideally, we would like to infer a full posterior over object paths. MCMC methods are guaranteed to yield true posterior samples as the number of samples tends to infinity; however, we have found batch Gibbs sampling to be impractical for inference in the DDPMO, as samples tend to remain stuck in local posterior optima (often when a track begins on one object before switching to another) and cannot converge to a high accuracy tracking hypothesis in a reasonable amount of time.

PMCMC [2] is a Markov chain Monte Carlo method that attempts to remedy these problems by using SMC as an intermediate sampling step to move efficiently through high dimensional state spaces. We implement a specific case known as the Particle Gibbs (PG) algorithm, where we sample from the conditional distributions used in Gibbs sampling via a modified version of Algorithm 1 referred to as conditional sequential Monte Carlo.

3.2.1 Conditional SMC

Conditional SMC [2] allows for SMC to be used as a proposal distribution in a Gibbs sampling algorithm. We must first introduce the notion of a particle's lineage. Let $A_t^{1:L}$ denote the indices of the L particles chosen during the resampling step in time t (in Algorithm 1). The lineage $B_{1:T}^{(l)}$ of a particle is recursively defined as $B_T^{(l)} = l$ and for $t = (T-1), \ldots, 1,\ B_t^{(l)} = A_t^{B_{t+1}^{(l)}}$. More intuitively, for the l^{th} particle, which contains the variables $\Theta_{1:T}^{(l)}$ at the final time T, $B_t^{(l)}$ denotes the index of the particle that contained the variables $\Theta_{1:t}^{(l)} \subset \Theta_{1:T}^{(l)}$ at time t.

Conditional SMC uses lineages to ensure that a given particle will survive all resampling steps, whereas the remaining particles are generated as before. We define conditional SMC for the DDPMO in Algorithm 2. Note that computation of particle weights and resampling (for relevant particles) is performed in the same manner as in SMC inference (Algorithm 1).

Algorithm 2 Conditional SMC for the DDPMO

Input: Extracted pixel data $\{\mathbf{x}_{1,1:N_1}, \ldots, \mathbf{x}_{T,1:N_T}\}$, number of particles L, number of local Gibbs iterations S, condition particle $\Phi_{1:T}^{(\eta)}$ with lineage $B_{1:T}^{(\eta)}$ $(\eta \in \{1, \ldots, L\})$

Output: Particle-conditional posterior samples $\{\Theta_{1:T}\}^{(1:L)}$ of all latent model variables

for $t = 1$ **to** T **do**
  for $l = 1$ **to** L **do**
    if $l \neq B_t^{(\eta)}$ **then**
      for iter $= 1$ **to** S **do**
        Sample $(c_{t,1:N_t})^{(l)} \sim Q_1$ and $(\theta_t^{1:K_{t,N_t}})^{(l)} \sim Q_2$
      end for
      for $k = 1$ **to** K_{t,N_t} **do**
        Sample $(\Delta m_t^k)^{(l)} \sim \mathrm{Binom}((m_{t,N_t}^k)^{(l)}, \rho)$
        Set $(m_{t+1,0}^k)^{(l)} = (m_{t,N_t}^k)^{(l)} - (\Delta m_t^k)^{(l)}$
        Sample $(z_{t+1,1:M}^k)^{(l)} \sim T_1((\theta_t^k)^{(l)})$
      end for
      Set $\Theta_t^{(l)} = \Big\{(c_{t,1:N_t})^{(l)}, (\theta_t^{1:K_{t,N_t}})^{(l)}, (m_{t+1,0}^{1:K_{t,N_t}})^{(l)},\ (z_{t+1,1:M}^{1:K_{t,N_t}})^{(l)}\Big\}$
    else
      Set $\Theta_t^{(l)} = \Phi_t^{(\eta)}$
    end if
    Compute particle weight $\tilde{w}_t^{(l)}$
  end for
  for $l = 1$ **to** L **do**
    if $l \neq B_t^{(\eta)}$ **then**
      Normalize weights and resample particles
    end if
  end for
end for

3.2.2 Particle Gibbs

In the particle Gibbs (PG) algorithm [2], the model variables are first initialized, and then conditional SMC (Algorithm 2) is run for a number of iterations. More specifically, at the end of each iteration, a sample is drawn from the set of weighted particles returned by conditional SMC, and this sample is conditioned upon in the next iteration. The PG algorithm for the DDPMO is formalized in Algorithm 3.

As the PG inference requires all variables to be initialized, SMC inference (Algorithm 1) can be used as a quick way to provide near-MAP initialization of variables.

Algorithm 3 PMCMC (Particle Gibbs) for the DDPMO

Input: Extracted pixel data $\{\mathbf{x}_{1,1:N_1}, \ldots, \mathbf{x}_{T,1:N_T}\}$, number of global Gibbs iterations G, number of particles L, number of local Gibbs iterations S

Output: Posterior samples $\left\{\theta_1^{1:K_{1,N_1}}, \ldots, \theta_T^{1:K_{T,N_T}}\right\}^{1:G}$ of the object model parameters

1: Initialize all model variables to $\Phi_0^{(L)}$
2: **for** $g = 1$ **to** G **do**
3: Run conditional SMC (Algorithm 2) with input $\{\mathbf{x}_{1,1:N_1}, \ldots, \mathbf{x}_{T,1:N_T}\}$, L, S, and conditional on particle $\Phi_{g-1}^{(L)}$ to get particle set $\{\Theta_{1:T}^{(1)}, \ldots, \Theta_{1:T}^{(L)}\}$
4: Draw $\Phi_g^{(L)} \sim \text{Unif}(\{\Theta_{1:T}^{(1)}, \ldots, \Theta_{1:T}^{(L)}\})$
5: **end for**
6: Return $\left\{\theta_1^{1:K_{1,N_1}}, \ldots, \theta_T^{1:K_{T,N_T}}\right\}^{1:G} \in \Phi_{1:G}^{(L)}$

4 Experiments

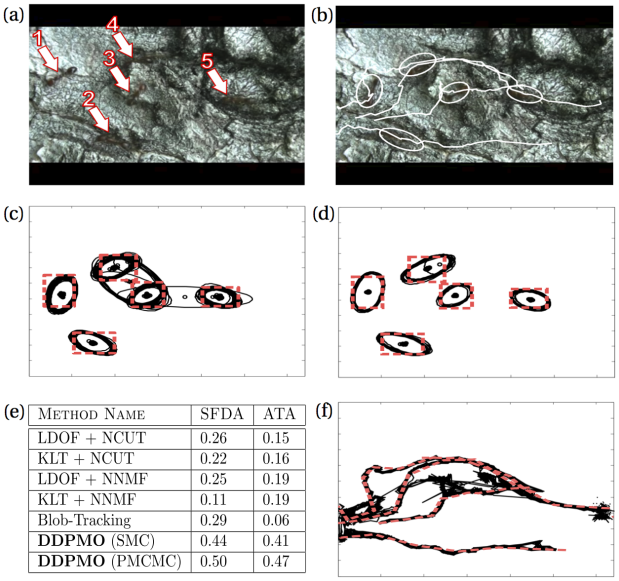
We demonstrate the DDPMO on three real video datasets: a video of foraging ants, where we show improved performance over other detection-free methods; a human tracking benchmark video, where we show comparable performance against object-specific methods designed to detect humans; and a T cell tracking task where we demonstrate our method on a video with a large number of objects and show how our unsupervised method can be used to automatically train a supervised object detector.

Detection-free comparison methods. Detection-free tracking strategies aim to find and track objects without any prior information about the objects' characteristics nor any manual initialization. One type of existing strategy uses optical flow or feature tracking algorithms to produce short tracklets, which are then clustered into full object tracks. We use implementations of Large Displacement Optical Flow (LDOF) [10] and the Kanade-Lucas-Tomasi (KLT) feature tracker [27] to produce tracklets [1]. Full trajectories are then formed using the popular normalized-cut (NCUT) method [25] to cluster the tracklets or with a variant that uses non-negative matrix factorization (NNMF) to cluster motion using tracklet velocity information [12] [2]. We also compare against a detection-free blob-tracking method, where extracted foreground pixels are segmented into components in each frame [26] and then associated with the nearest neighbor criterion [33].

Performance metrics. For quantitative comparison, we report two commonly used performance metrics for object detection and tracking, known as the sequence frame detection accuracy (SFDA) and average tracking accuracy (ATA) [22]. These metrics compare detection and tracking results against human-authored ground-truth, where SFDA$\in [0, 1]$ corresponds to detection performance and ATA$\in [0, 1]$ corresponds to tracking performance. We authored the ground-truth for all videos with the Video Performance Evaluation Resource (ViPER) tool [14].

4.1 Insect Tracking

In this experiment, we aim to demonstrate the ability of the DDPMO to find and track objects in a difficult detection scenario. The video contains six ants with a similar texture and color distribution as the background. The ants are hard to discern, and it is unclear how a predefined detection criteria might be constructed. Futher, the ants move erratically and the spatial data extracted via frame differencing does not yield a clear segmentation of the objects in individual frames. A still image from the video, with ant locations shown, is given in Figure 3(a). We compare the SMC and PMCMC



Method Name	SFDA	ATA
LDOF + NCUT	0.26	0.15
KLT + NCUT	0.22	0.16
LDOF + NNMF	0.25	0.19
KLT + NNMF	0.11	0.19
Blob-Tracking	0.29	0.06
DDPMO (SMC)	0.44	0.41
DDPMO (PMCMC)	0.50	0.47

Figure 3: The ants in (a) are difficult to discern (positions labeled). We plot 100 samples from the inferred posterior over object parameters (using SMC (c) and PMCMC (d)) with ground-truth bounding boxes overlaid (dashed). PMCMC proves to give more accurate object parameter samples. We also plot samples over object tracks (sequences of mean parameters) using PMCMC in (f) , and its MAP sample in (b). We show the SFDA and ATA scores for all comparison methods in (e).

inference algorithms, and find that PMCMC yields more accurate posterior samples (3(d)) than SMC (3(c)). Ground-truth bounding boxes (dashed) are overlaid on the posterior samples. The MAP PMCMC sample is shown in 3(b) and posterior samples of the object tracks are shown in 3(f), along with overlaid ground-truth tracks (dashed). SFDA

[1] The LDOF implementation can be found at `http://www.seas.upenn.edu/~katef/LDOF.html` and the KLT implementation at `http://www.ces.clemson.edu/~stb/klt/`.

[2] The NCUT implementation can be found at `http://www.cis.upenn.edu/~jshi/software/` and the NNMF implementation at `http://www.ornl.gov/~czx/research.html`.

and ATA performance metrics for all comparison methods are shown in 3(e). The DDPMO yields higher metric values than all other detection-free comparison methods, with PM-CMC inference scoring higher than SMC. The comparison methods seemed to suffer from two primary problems: very few tracklets could follow object positions for an extended sequence of frames, and clustering tracklets into full tracks sharply decreased in accuracy when the objects came into close contact with one another.

4.2 Comparisons with Detector-based Methods

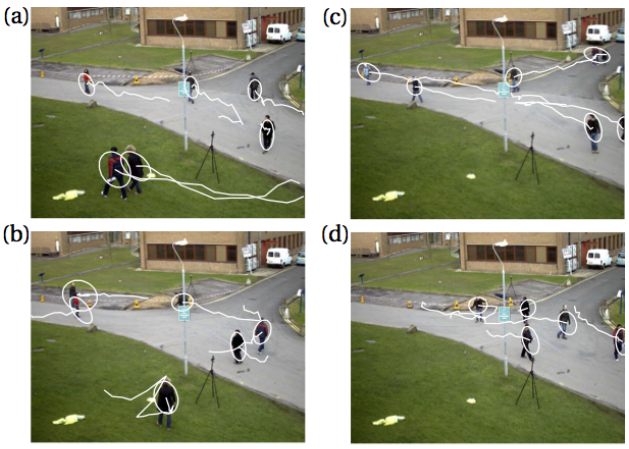
In this experiment we aim to show that our general-purpose algorithm can compete against state of the art object-specific algorithms, even when it has no prior information about the objects. We use a benchmark human-tracking video from the International Workshop on Performance Evaluation of Tracking and Surveillance (PETS) 2009-2013 conferences [16], due to its prominence in a number of studies (listed in Figure 4(f)). It consists of a monocular, stationary camera, 794 frame video sequence containing a number of walking humans. Due to the large number of frames and objects in this video, we perform inference with the SMC algorithm only.

The DDPMO is compared against ten object-specific detector-based methods from the PETS conferences. These methods all either leverage assumptions about the orientation, position, or parts of humans, or explicitly use pre-trained human detectors. For example, out of the three top scoring comparison methods, [9] uses a state of the art pedestrian detector, [32] performs head and feet detection, and [13] uses assumptions about human geometry and orientation to segment humans and remove shadows.

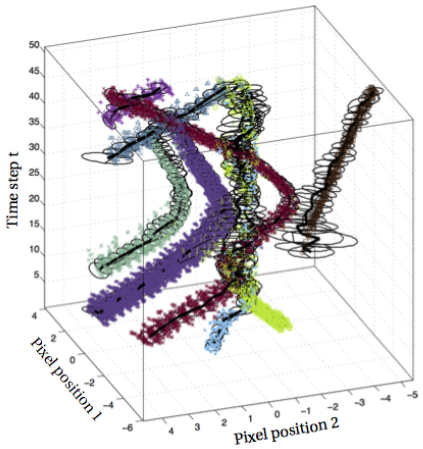
In Figure 4(a-d), the MAP sample from the posterior distribution over the object parameters is overlayed on the extracted data over a sequence of frames. The first 50 frames from the video are shown in 4(e), where the assignment of each data point is represented by color and marker type. We show the SFDA and ATA values for all methods in 4(f), and can see that the DDPMO yields comparable results, receiving the fourth highest SFDA score and tying for the second highest ATA score.

4.3 Tracking Populations of T Cells

Automated tracking tools for cells are useful for cell biologists and immunologists studying cell behavior. We present results on a video containing T cells that are hard to detect using conventional methods due to their low constrast appearance against a background (Figure 5(a)). Furthermore, there are a large number of cells (roughly 60 per frame, 92 total). In this experiment, we aim to demonstrate the ability of the DDPMO to perform a tough detection task while scaling up to a large number of objects. Ground-truth bounding boxes for the cells at a single frame are shown in



(e)

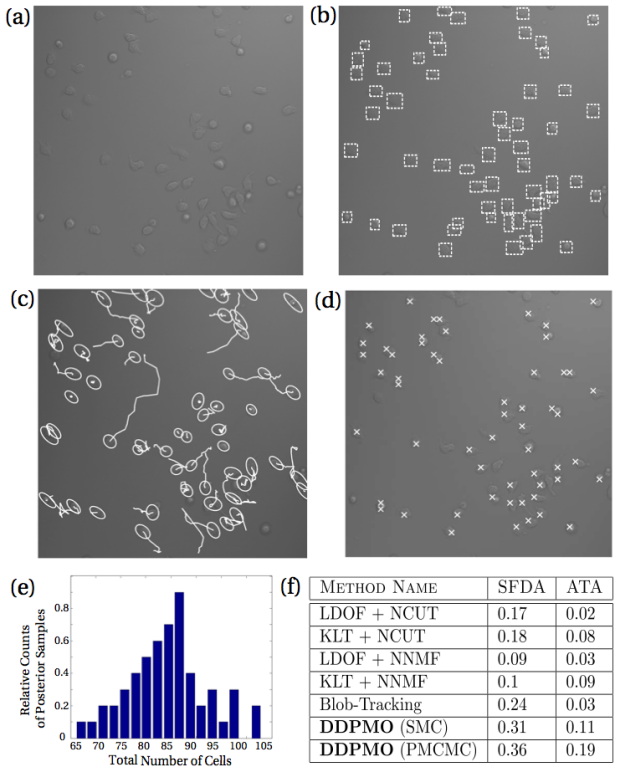


(f)

Method Name	SFDA	ATA
Breitenstein [9]	0.57	0.30
Yang [32]	0.55	0.45
Conte [13]	0.53	0.06
DDPMO (SMC)	0.51	0.30
Berclaz [7]	0.48	0.15
Alahi 1 [1]	0.43	0.04
Alahi 2 [1]	0.42	0.05
Bolme 1 [8]	0.41	NA
Ge [20]	0.38	0.04
Bolme 2 [8]	0.34	NA
Arsic [4]	0.18	0.02

Figure 4: DDPMO results on the PETS human tracking benchmark dataset and comparison with object-detector-based methods. The MAP object parameter samples are overlaid on four still video frames (a-d). The MAP object parameter samples are also shown for a sequence of frames (a 50 time-step sequence) along with spatial pixel observations (e) (where the assignment variables $c_{t,n}$ for each pixel are represented by marker type and color). The SFDA and ATA performance metric results for the DDPMO and ten human-specific, detection-based tracking algorithms are shown in (f), demonstrating that the DDPMO achieves comparable performance to these human-specific methods. Comparison results were provided by the authors of [16].

5(b) and PMCMC inference results (where the MAP sample is plotted) are shown in in 5(c). A histogram illustrating the inferred posterior over the total number of cells is shown in 5(e). It peaks around 87, near the true value of 92 cells.



(f)

Method Name	SFDA	ATA
LDOF + NCUT	0.17	0.02
KLT + NCUT	0.18	0.08
LDOF + NNMF	0.09	0.03
KLT + NNMF	0.1	0.09
Blob-Tracking	0.24	0.03
DDPMO (SMC)	0.31	0.11
DDPMO (PMCMC)	0.36	0.19

Figure 5: T cells are numerous, and hard to detect due to low contrast images (a). For a single frame, ground-truth bounding boxes are overlaid in (b), and inferred detection and tracking results are overlaid in (c). A histogram showing the posterior distribution over the total number of cells is shown in (e). The SFDA and ATA for the detection-free comparison methods are shown in (f). Inferred cell positions (unsupervised) were used to automatically train an SVM for supervised cell detection; SVM detected cell positions for a single frame are shown in (d).

Manually hand-labeling cell positions to train a detector is feasible but time consuming; we show how unsupervised detection results from the DDPMO can be used to automatically train a supervised cell detector (a linear SVM), which can then be applied (via a sliding window across each frame) as a secondary, speedy method of detection (Figure 5(d)). This type of strategy in conjunction with the DDPMO could allow for an ad-hoc way of constructing detectors for arbitrary objects on the fly, which could be taken and used in other vision applications, without needing an explicit predefined algorithm for object detection.

5 Conclusion

The DDPMO provides the ability to find, track, and learn representations of arbitrary objects in a video, in a single model framework, in order to accomplish detection-free tracking. We detail inference algorithms that can be used in both online and offline settings and provide results on a number of real video datasets. We consistently achieve better performance than other detection-free tracking strategies and even achieve competitive performance with object-specific detector-based methods on a human tracking benchmark video. Furthermore, we've demonstrated the ability of our model to perform accurate localization and tracking in videos with large numbers of objects, and in those that contain instances of full or partial occlusion, objects with shifting appearance or orientation, and objects for which it is difficult to construct an explicit detection strategy.

We've also shown how the DDPMO can provide an unsupervised, detection-free way to train a discriminative object detector for arbitary objects. This combination could provide a way to build object detectors for unknown objects on the fly and increase the accuracy or speed of localization and tracking within our model framework. We envision the DDPMO to be particularly useful in settings where the number and type of objects are unknown, or the objects' appearances are highly variable, and a high-quality general-purpose object localization and tracking method is desirable.

References

[1] A. Alahi, L. Jacques, Y. Boursier, and P. Vandergheynst, *Sparsity-driven people localization algorithm: Evaluation in crowded scenes environments*, Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on, IEEE, 2009, pp. 1–8.

[2] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein, *Particle markov chain monte carlo methods*, Journal of the Royal Statistical Society: Series B (Statistical Methodology) **72** (2010), no. 3, 269–342.

[3] Charles E Antoniak, *Mixtures of dirichlet processes with applications to bayesian nonparametric problems*, The annals of statistics (1974), 1152–1174.

[4] D. Arsic, A. Lyutskanov, G. Rigoll, and B. Kwolek, *Multi camera person tracking applying a graph-cuts based foreground segmentation in a homography framework*, Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on, IEEE, 2009, pp. 1–8.

[5] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie, *Robust object tracking with online multiple instance learning*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **33** (2011), no. 8, 1619–1632.

[6] John L Barron, David J Fleet, and SS Beauchemin, *Performance of optical flow techniques*, International journal of computer vision **12** (1994), no. 1, 43–77.

[7] J. Berclaz, F. Fleuret, and P. Fua, *Multiple object tracking using flow linear programming*, Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on, IEEE, 2009, pp. 1–8.

[8] D.S. Bolme, Y.M. Lui, BA Draper, and JR Beveridge, *Simple real-time human detection using a single correlation filter*, Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on, IEEE, 2009, pp. 1–8.

[9] Michael D Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool, *Markovian tracking-by-detection from a single, uncalibrated camera*, (2009).

[10] Thomas Brox and Jitendra Malik, *Large displacement optical flow: descriptor matching in variational motion estimation*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **33** (2011), no. 3, 500–513.

[11] F. Caron, M. Davy, and A. Doucet, *Generalized Polya urn for time-varying Dirichlet process mixtures*, 23rd Conference on Uncertainty in Artificial Intelligence (UAI'2007), Vancouver, Canada, July 2007, 2007.

[12] Anil M Cheriyadat and Richard J Radke, *Non-negative matrix factorization of partial track data for motion segmentation*, Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 865–872.

[13] D. Conte, P. Foggia, G. Percannella, and M. Vento, *Performance evaluation of a people tracking system on pets2009 database*, Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on, IEEE, 2010, pp. 119–126.

[14] D. Doermann and D. Mihalcik, *Tools and techniques for video performance evaluation*, Pattern Recognition, 2000. Proceedings. 15th International Conference on, vol. 4, IEEE, 2000, pp. 167–170.

[15] Randal Douc and Olivier Cappé, *Comparison of resampling schemes for particle filtering*, Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on, IEEE, 2005, pp. 64–69.

[16] A. Ellis and J. Ferryman, *Pets2010 and pets2009 evaluation of results using individual ground truthed single views*, Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on, IEEE, 2010, pp. 135–142.

[17] Rémi Emonet, Jagannadan Varadarajan, and J-M Odobez, *Extracting and locating temporal motifs in video scenes using a hierarchical non parametric bayesian model*, Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 3233–3240.

[18] Katerina Fragkiadaki and Jianbo Shi, *Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement*, Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE, 2011, pp. 2073–2080.

[19] Jan Gasthaus, *Spike sorting using time-varying Dirichlet process mixture models*, 2008.

[20] W. Ge and R.T. Collins, *Evaluation of sampling-based pedestrian detection for crowd counting*, Performance Evaluation of Tracking and Surveillance (PETS-Winter), 2009 Twelfth IEEE International Workshop on, IEEE, 2009, pp. 1–7.

[21] A. Gelman, *Bayesian data analysis*, CRC press, 2004.

[22] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang, *Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol*, IEEE Transactions on Pattern Analysis and Machine Intelligence (2008), 319–336.

[23] Steven N MacEachern, *Dependent dirichlet processes*, Unpublished manuscript, Department of Statistics, The Ohio State University (2000).

[24] Peter Meer, *Kernel-based object tracking*, IEEE Transactions on pattern analysis and machine intelligence **25** (2003), no. 5.

[25] Jianbo Shi and Jitendra Malik, *Normalized cuts and image segmentation*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22** (2000), no. 8, 888–905.

[26] Chris Stauffer and W. Eric L. Grimson, *Learning patterns of activity using real-time tracking*, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22** (2000), no. 8, 747–757.

[27] Carlo Tomasi and Takeo Kanade, *Detection and tracking of point features*, School of Computer Science, Carnegie Mellon Univ., 1991.

[28] Pavan Turaga, Rama Chellappa, Venkatramana S Subrahmanian, and Octavian Udrea, *Machine recognition of human activities: A survey*, Circuits and Systems for Video Technology, IEEE Transactions on **18** (2008), no. 11, 1473–1488.

[29] Harini Veeraraghavan, Paul Schrater, and Nikos Papanikolopoulos, *Robust target detection and tracking through integration of motion, color, and geometry*, Computer Vision and Image Understanding **103** (2006), no. 2, 121–138.

[30] Xiaogang Wang, Xiaoxu Ma, and Eric Grimson, *Unsupervised activity perception by hierarchical bayesian models*, Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE, 2007, pp. 1–8.

[31] Bo Wu and Ram Nevatia, *Cluster boosted tree classifier for multi-view, multi-pose object detection*, Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE, 2007, pp. 1–8.

[32] J. Yang, PA Vela, Z. Shi, and J. Teizer, *Probabilistic multiple people tracking through complex situations*, 11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, 2009.

[33] Alper Yilmaz, Omar Javed, and Mubarak Shah, *Object tracking: A survey*, Acm Computing Surveys (CSUR) **38** (2006), no. 4, 13.