SIViP (2016) 10:795–802
DOI 10.1007/s11760-015-0817-x

ORIGINAL PAPER

Tracklet clustering for robust multiple object tracking using distance dependent Chinese restaurant processes

Ibrahim Saygin Topkaya[1] · **Hakan Erdogan**[1] · **Fatih Porikli**[2,3]

Received: 4 June 2015 / Revised: 19 August 2015 / Accepted: 10 September 2015 / Published online: 19 September 2015

Abstract To contrive an accurate and efficient strategy for object detection–object track assignment problem, we present a tracklet clustering approach using distance dependent Chinese restaurant processes (ddCRPs), which employ a two-level robust object tracker. The first level is an ordinary tracklet generator that obtains short yet reliable tracklets. In the second level, we cluster the tracklets over time based on color, spatial and temporal attributes, where the nonparametric process of clustering with ddCRPs allows us to maintain an unknown number of objects. Unlike the previously proposed Chinese restaurant processes and Dirichlet process mixture models, our ddCRPs method does not require prescribed complex cluster models to be initialized and updated, and thus, we can cluster complex tracklets by only computing similarities between them. Our comparative evaluations on tracking different object types demonstrate the generality of our approach.

Keywords Distance dependent Chinese restaurant process · Tracklet clustering · Object tracking

1 Introduction

Tracking-by-detection methods generate a set of object candidates at each frame and assign them to existing tracks over time. Independent from the object detector or tracker being used, the decision criteria to assign a detection result to a track essentially determines the success of the tracking process. An over-conservative criterion often dismisses the correct assignments yielding incomplete and partitioned tracks, whereas an over-relaxed criterion causes false assignments that quickly lead into tangling and drift from the actual trajectories. Extracting highly confident but relatively short tracks, called *tracklets*, and then hierarchically merging them later into longer and more complete tracks is a well established [1] solution for this dilemma. For instance, a recent study [2] on tracklet linkage attempts to extract tracklets from face detections and group them using pairwise tracklet similarities and hidden Markov models. However, this method requires the number of groups, i.e., tracks, to be known a priori.

Here, we present a nonparametric tracklet clustering approach for tracking an unknown number of objects. We take object detection windows, construct short and reliable tracklets, and extract color, spatial and temporal features of tracklets. Using distance dependent Chinese restaurant processes (ddCRPs), we cluster these tracklets. By our definition, each cluster of tracklets corresponds to a unique track and thus a distinct trajectory of a single object over time. A very recent work [3] builds color-based appearance models using temporally coherent Chinese restaurant processes (CRPs) and Dirichlet process mixture models (DPMMs) to cluster tracklets for person discovery. The difference of our work is that we use temporal information, e.g., position change in time, in addition to appearance information, which enables us tracking objects without depending on complex global cluster models. In addition, we obtain clusters using only pairwise tracklet assignments.

In the following Sect. 2, we describe the tracklet extraction scheme that we employ. In Sect. 3, we introduce ddCRPs and explain the fundamental distinction of ddCRPs from CRPs (and thus DPMMs). In Sect. 4, we present our tracklet clustering scheme and discuss how we leverage the flexibility of

✉ Ibrahim Saygin Topkaya
isaygint@sabanciuniv.edu

[1] Sabanci University, VPALAB, Istanbul, Turkey

[2] Research School of Engineering, ANU, Canberra, Australia

[3] NICTA, Canberra, Australia

ddCRPs in order to cluster tracklets efficiently. Our experimental results on various datasets and object types in Sect. 5 show that ddCRPs have significant potential to be applied to a broad class of tracking problems.

2 Tracklet generation

Any object detector can be incorporated into our method. We also use a Gaussian mixture model-based background model [4,5], which is suitable for stationary camera setups, to filter out potentially false detections caused by the background. We handle missed detections (false negatives), e.g., due to occlusions or objects blending into background, during the clustering stage.

We denote an object detection $\mathrm{d}: (\mathrm{d}_x, \mathrm{d}_y, \mathrm{d}_u, \mathrm{d}_v, \mathrm{d}_{ha}, \mathrm{d}_{hb})$ as a rectangular window centered on $\mathrm{d}_x, \mathrm{d}_y$ with width d_u and height d_v and having a color distribution characterized by two histograms d_{ha} and d_{hb}, corresponding to the *Lab* channels of the foreground pixels in the window. Accordingly, we represent the parameters of a tracklet inherited from the previous frame $(f-1)$ by ϕ^{f-1}: $(\phi_x^{f-1}, \phi_y^{f-1}, \phi_u^{f-1}, \phi_v^{f-1}, \phi_{ha}^{f-1}, \phi_{hb}^{f-1})$. At a particular frame in time, a tracklet is defined by its final position, size and accumulated color histogram values at that frame (f), i.e., $\phi_x^f, \phi_y^f, \phi_u^f, \phi_v^f$ are the position and size of the last detection assigned to the tracklet, and ϕ_{ha}^f, ϕ_{hb}^f are the accumulated color histograms of all detections assigned to it through time.

We assume that at one frame, a single object can exist at *only one location* and is represented with at most one detection. Consecutively at each frame, object detections are either assigned to existing tracklets inherited from the previous frame based on similarity of color, location and size features, or a new tracklet is initiated starting from the current frame. In case a detection is assigned to a tracklet, the position and the size of the tracklet are updated.

Accordingly, we define the affinity $A(\phi, \mathrm{d})$ of detection d to the tracklet ϕ at frame f similar to [6] and [2] as:

$$\begin{aligned} A(\phi, \mathrm{d}) = {} & \mathcal{N}(\mathrm{d}_x | \phi_x^{f-1}, \sigma_x) \times \mathcal{N}(\mathrm{d}_y | \phi_y^{f-1}, \sigma_y) \\ & \times \mathcal{N}(\mathrm{d}_u | \phi_u^{f-1}, \sigma_u) \times \mathcal{N}(\mathrm{d}_v | \phi_v^{f-1}, \sigma_v) \\ & \times \mathcal{N}(S(\mathrm{d}_{ha}, \phi_{ha}^{f-1}) | 0, \sigma_a) \\ & \times \mathcal{N}(S(\mathrm{d}_{hb}, \phi_{hb}^{f-1}) | 0, \sigma_b), \end{aligned} \tag{1}$$

where $\mathcal{N}$ is the probability density under Gaussian distribution. The similarity operator S between two histograms is defined using the following sum of ratios over all B histogram bins [7]:

$$S(h_1, h_2) = 1 - \frac{1}{B} \sum_{b=1}^{B} \frac{\min(h_1(b), h_2(b))}{\max(h_1(b), h_2(b))}. \tag{2}$$

If no detections can be assigned to an existing tracklet, it is terminated and removed from the current set Φ of tracklets that is used to assign new detections D on the following frames. In the end, a tracklet is defined by the sequence of position and size values (i.e., sequences of $\phi_x, \phi_y, \phi_u, \phi_v$) of the constituting detections and accumulated color histogram values (i.e., accumulated ϕ_{ha} and ϕ_{hb}) over time.

To prevent ambiguity and drifts, we make sure that a detection is assigned to a tracklet if the affinity score is significantly greater than the affinities to all other tracklets. Even after this, only the detection with the best affinity score is assigned to a tracklet.

We aim to filter out false detections during and after tracklet extraction. Before assigning detections to tracklets, we eliminate detections that do not have any foreground pixels, as illustrated in Fig. 1a. We also learn the regions of the scene that produce false detections by an online process where we eliminate and keep the record of detections that lasted only for a single frame without being assigned to a tracklet. After a region produces such single-frame detections for κ_A number of times, we begin to eliminate detections from that region as well, as shown in Fig. 1b. These two simple elimination schemes help us to filter out false detections during tracklet generation and prevent possible local drifting.

After tracklets are generated, we calculate spatial neighborhood statistics for detections that constitute the tracklets at every frame. For each detection d, we gather all detections from all frames that intersect with d window to form the neighborhood N_d. For detections in N_d, we calculate

Fig. 1 Sample detections and tracklets that have been filtered out

two statistics: $\tilde{N}_{\mathrm{d}}$, the median height of the detections, and $|N_{\mathrm{d}}|$, the number of detections. Using these statistics, we filter out tracklets: (i) all detections of which are at least κ_s times greater in height than the median of their neighborhoods (e.g., green rectangle in Fig. 1c) and (ii) average number of neighbors of the detections of which are smaller than a ratio κ_d of the total number of frames in the sequence (e.g., Fig. 1d). Using a local size statistic rather than a global size heuristic as in (i) allows us to apply a perspective invariant size filtering to the detections, and using the number of neighbors as in (ii) allows us to filter out detections which are likely to be outliers as they appear in the areas of the scene with a low overall object detection likelihood.

We finally remove the tracklets, all detections of which intersect with a larger detection by at least half of their own area, since they usually are false detections for parts of an object (e.g., blue rectangle in Fig. 1c).

The overall pseudocode for tracklet generation is presented in Algorithm 1.

Input: For frame $f = 1, \ldots, F$
D_f detections, BG_f background for f
Result: $\Phi = \{\phi_1, \phi_2 ...\}$ set of tracklets
$\Phi \leftarrow \{\}$, $FD \leftarrow \{\}$
for $f \in \{1...F\}$ **do**
  for $\mathrm{d} \in \mathrm{D}_f = \{\mathrm{d}_1, \mathrm{d}_2 ...\}$ **do**
    // Background or false detections
    if $BG_f(\mathrm{d}) = 1$ **or** $FD(\mathrm{d}) \geq \kappa_A$ **then**
      $\mathrm{D}_f \leftarrow \mathrm{D}_f - \{\mathrm{d}\}$
    else
      $\Phi \Leftarrow \mathrm{d}$ // Update Φ using Eq. (1)
    end
  end
  // Update false detection areas
  for $\phi \in \Phi$ **do**
    if $\phi_f = \emptyset$ **and** $Length(\phi) = 1$ **then**
      $\Phi \leftarrow \Phi - \{\phi\}$
      $FD(\phi^{f-1}) = FD(\phi^{f-1}) + 1$
    end
  end
end
// Post process and filter tracklets
for $\phi \in \Phi$ **and** $\mathrm{d} \in \phi$ **do**
  $N_d \leftarrow \{\}$, $C \leftarrow \{\}$
  for $\phi_n \in \Phi$ **and** $\mathrm{d}_n \in \phi_n$ **do**
    if $Area(\mathrm{d} \cap \mathrm{d}_n) > 0$ **then**
      $N_d \leftarrow N_d \cup \{\mathrm{d}_n\}$
      if $SameFrame(\mathrm{d}, \mathrm{d}_n)$ **and**
        $Area(\mathrm{d} \cap \mathrm{d}_n) \geq 0.5 \cdot Area(d)$ **then**
        $C \leftarrow C \cup \mathrm{d}_n$
      end
    end
  end
  if $\mathrm{d}_v / \tilde{N}_d > \kappa_s$ **or** $avg(|N_d|)/F < \kappa_d$ **or** $\mathrm{d} \in C$
    $\forall \mathrm{d} \in \phi$ **then**
    $\Phi \leftarrow \Phi - \{\phi\}$
  end
end

Algorithm 1: Algorithm for tracklet extraction

3 Distance dependent Chinese restaurant processes

DPMMs and CRPs, as well as ddCRPs, are statistical tools for nonparametric clustering of data. In this section, we summarize their basic aspects and present our motivation behind employing ddCRPs.

3.1 DPMMs and CRPs

DPMMs provide a way to model a set of observation data as a mixture of unknown number of distributions having the same form $G(\theta)$, called as the base distribution [8]. DPMM clustering task assumes an infinite number of mixture components exist, but with only a finite subset of these components having data assigned to them. In other words, DPMMs regard the task of clustering as finding the parameters of those finite and unknown number of mixture components.

Suppose the data to be clustered are $\{X_n | n = 1 \ldots N\}$. For $X_n \in k$, $X_n \sim G(\theta_k)$, i.e., observations are distributed according to the parameters of the mixture components that they are assigned to, where θ_k denotes parameters of the kth mixture component. A common application of DPMMs is to take $G(\theta)$ as a Gaussian distribution, where θ in this case defines the distribution parameters μ and σ.

Methods such as Gibbs sampling are widely used [9] for DPMMs. Gibbs sampling handles all observations iteratively, and for each observation, it computes the probability of an assignment to an existing or new cluster with:

$$p(c_n = k | X_n, c_{-n}, \alpha, \theta) \propto \begin{cases} N_k \times p(X_n|\theta_k) & k \leq K \\ \alpha \times \int_\theta p(X_n|\theta)\, \mathrm{d}\theta & k = K+1 \end{cases} \tag{3}$$

where $c_n = k$ denotes $X_n \in k$, c_{-n} denotes all assignments except nth, N_k is the number of assignments to cluster k, K is the number of existing clusters before the new assignment and α is a control parameter whose higher values result in more clusters.

For both conditions, the assignment prior of Eq. (3), i.e., left of $p(X_n|\theta)$ cluster likelihood, differs only on N_k and α. That is, the more observations are assigned to a cluster, the more probable a new observation will be assigned to it. These conditions and the *infinite number of components* assumption mentioned above bear the *CRPs* analogy [10], where a Chinese restaurant with an infinite number of tables without any capacity limit is considered. Each new customer chooses a table with existing customers, probability of which is proportional to the number of previous customers sitting at that table (proportional to N_k), or chooses to sit down at a new table with a fixed probability (proportional to α). Increasing α results in more occupied tables with fewer customers or few tables with more customers vice versa. Since each

customer can sit at one table, the customers are *partitioned* across tables (clusters).

DPMM-based clustering has recently received attention for object tracking in computer vision literature. In [11], the tracked objects themselves are modeled as mixture components, and the parameters of these components at a particular time (i.e., frame) are selected as visual features including color, size and position of the tracked objects.

3.2 Distance dependent CRPs

Continuing with the restaurant analogy, ddCRP [12] seeks assignments between customers (i.e., observations) only. As a result, a new arriving customer chooses to sit down with an existing customer (consecutively at the same table) or by itself. Thus, customers that choose to sit down together, either directly or indirectly through another customer, constitute a table. In other words, observations that are assigned to each other directly or indirectly through other observations, constitute a cluster. Under this analogy, the Gibbs sampling probability used to sample observation assignments becomes:

$$p(c_i = j|X, c_{-i}, \alpha, F) \propto \begin{cases} F(X_i, X_j) \times p(X|z(c)) & i \neq j \\ \alpha \times p(X|z(c)) & i = j \end{cases} \quad (4)$$

where c_{-i} denotes all assignments except ith, $c_i = j$ denotes that X_i is *linked* to (thus *assigned* to the same cluster with) X_j and the assignment prior is proportional to $F(X_i, X_j)$, which is the similarity measure between these two. Notice that no explicit cluster parameters are defined like θ_k in Eq. (3).

The $z(c)$ term, where $c = c_i \cup c_{-i}$, denotes the structure of the underlying clusters after assignment of c_i, and it is used to calculate the cluster likelihood for all observations X. In [12], different configurations of observation assignments at each Gibbs sampling step for a single observation are considered. We leave the discussion of this likelihood to the following section, where we define our likelihood within our application.

The similarity function $F(X_i, X_j)$ is defined between observations. Often a decay function of an exponential form [12], which is applied on the distance, is employed. We discuss the selection of our similarity function and how we build it around the features that we define in the following section as well.

4 Tracklet clustering with ddCRPs

The main contribution of our work is the clustering scheme that groups the tracklets extracted in Sect. 2 into tracks using ddCRPs. Within this definition and the ddCRP clustering framework, tracklets are observations and tracks are clusters. We denote a single tracklet/observation with ϕ_i and the set of all tracklets with Φ as in Sect. 2, rather than X_i and X as in Sect. 3. The reason we prefer ddCRPs over CRPs and DPMMs is the flexibility of integrating a custom similarity function $F(\phi_1, \phi_2)$ between tracklets ϕ_1 and ϕ_2 in Eq. (4) directly into the clustering process.

We use a similarity function that takes the changes of position, size and color features of tracklets over time into account. Within a DPMM framework, this would not be possible since it would not be easy to integrate a mixture model that covers such a custom likelihood. In other words, instead of defining a complex cluster model and assigning tracklets to clusters as well as updating cluster parameters at each assignment, we only define and calculate tracklet similarities once and obtain the clusters automatically by using the tracklet assignments.

4.1 Extracting features

The tracklets to be clustered are defined by vectors that hold historical information of position and size, and aggregated color information; i.e., ϕ:($\mathbf{x}$, $\mathbf{y}$, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{f}$, $\mathbf{h}_a$, $\mathbf{h}_b$), where $\mathbf{f}$ is the vector of frame numbers when the tracklet is visible. $\mathbf{x}$, $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{v}$ are vectors of same length with $\mathbf{f}$ and hold position and size information of the tracklet at each frame. $\mathbf{h}_a$ and $\mathbf{h}_b$ are normalized histograms with fixed bin numbers accumulated over all frames.

4.2 Tracklet similarity

We use a pairwise similarity function based on the probability of two tracklets belonging to the same tracked object. With the assumption that at one frame one object is represented with at most one detection, we set similarity as zero for temporally overlapping tracklets.

For two non-overlapping tracklets ϕ_1 and ϕ_2, let tracklet ϕ_1 be the former one and visible between frames f_1^s and f_1^e, i.e., $\mathbf{f_1} = \{f_1^s, f_1^s + 1 \ldots f_1^e - 1, f_1^e\}$ and ϕ_2 between frames f_2^s and f_2^e and $f_1^e < f_2^s$ and $\delta_{12} = f_2^s - f_1^e$ where $\delta_{12} > 0$. Here, $\mathbf{x_1}$ is a vector of the same length with $\mathbf{f_1}$, having values $\mathbf{x_1} = \{x_{f_1^s}, x_{f_1^s+1} \ldots x_{f_1^e-1}, x_{f_1^e}\}$ denoting x-coordinates. Similarly, $\mathbf{y_1}$, $\mathbf{u_1}$ and $\mathbf{v_1}$ denote y-coordinates, width and height of the tracklet ϕ_1 over time, respectively. Same set of vectors are also extracted for tracklet ϕ_2. $\mathbf{h}_{a,1}$, $\mathbf{h}_{b,1}$, $\mathbf{h}_{a,2}$ and $\mathbf{h}_{b,2}$ denote the color histograms of ϕ_1 and ϕ_2, accumulated between frames f_1^s and f_1^e (for ϕ_1) and f_2^s and f_2^e (for ϕ_2).

We obtain two similarities: F_{12}, similar to [2], which seeks the probability that ϕ_2 is similar to ϕ_1 when extrapolated to the same time that ϕ_2 is visible, and F_{21}, vice versa.

For F_{12}, we linearly extrapolate four values: $\hat{x}_{f_2^s}$ which is the x value at f_2^s extrapolated from the vector $\mathbf{x_1}$; $\hat{y}_{f_2^s}$ from

$\mathbf{y_1}$; $\hat{u}_{f_2^s}$ from $\mathbf{u_1}$; and $\hat{v}_{f_2^s}$ from $\mathbf{v_1}$. Similarly, for F_{21}, we extrapolate the corresponding values $\hat{x}_{f_1^e}$ from $\mathbf{x_2}$, $\hat{y}_{f_1^e}$ from $\mathbf{y_2}$, $\hat{u}_{f_1^e}$ from $\mathbf{u_2}$ and $\hat{v}_{f_1^e}$ from $\mathbf{v_2}$.

Then, we define the likelihood F_{12}, which yields higher values where the extrapolated and observed sizes and positions for two tracklets are close (with the variances being proportional to the size of the object), histograms are similar, and the tracklets are closer temporally, as:

$$\begin{aligned} F_{12}(\phi_1,\phi_2) = \mathcal{N}(x_{f_2^s}|\hat{x}_{f_2^s}, u_{f_1^e}) \times \mathcal{N}(y_{f_2^s}|\hat{y}_{f_2^s}, v_{f_1^e}) \\ \times \mathcal{N}(u_{f_2^s}|\hat{u}_{f_2^s}, 0.1 \cdot u_{f_1^e}) \\ \times \mathcal{N}(v_{f_2^s}|\hat{v}_{f_2^s}, 0.1 \cdot v_{f_1^e}) \\ \times \mathcal{N}(S(h_{a,1}, h_{a,2})|0, \sigma_a) \\ \times \mathcal{N}(S(h_{b,1}, h_{b,2})|0, \sigma_b) \\ \times \mathcal{N}(\delta_{12}|0, f_1^e - f_1^s), \end{aligned} \tag{5}$$

where S is defined in Eq. (2). We calculate $F_{21}(\phi_1, \phi_2)$ similar to Eq. (5) for the other set of extrapolated and observed values and define the final similarity value as:

$$F(\phi_1,\phi_2) = \begin{cases} \max(F_{12}, F_{21}) & F_{12} > \varepsilon \text{ and } F_{21} > \varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Equation (5) does not impose tracklets to be sequential, and thus, occlusion handling is implicitly integrated into the model through the tracklet similarity function since we can assign non-sequential tracklets to each other and potentially cover the missed detections in-between.

4.3 Cluster likelihood

The assignment prior $F(\phi_i, \phi_j)$ of Eq. (4) can be obtained as in Eq. (6), but we also need to define our cluster likelihood term $p(\Phi|z(c))$. The $z(c)$ term denotes the cluster structure after assignment of c_i. We impose a hard limit on the cluster likelihood to prevent overlapping tracklets to constitute a cluster. It is necessary to check the overlap after every assignment, since even though the tracklets that are assigned to each other with c_i may not overlap, their newly formed cluster may contain members that overlap each other. Let $P(c)$ be the set of all directly and indirectly connected tracklet pairs implied by assignments c, then the likelihood is proportional to:

$$p(\Phi|z(c)) \propto \begin{cases} 1 & \text{if } \forall(\phi_i,\phi_j) \in P(c); \mathbf{f_i} \cap \mathbf{f_j} = \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

4.4 Gibbs sampling and cluster/track outputs

For ease of implementation and in order to speed up likelihood calculations, before applying Gibbs sampling, we construct a similarity matrix M (similar to [2]) where $M_{ij} = F(\phi_i, \phi_j)$ from Eq. (6), which is symmetric and values for the overlapping tracklets are zero. These tracklet similarities are calculated once, and same values (of $M_{ij} = F(\phi_i, \phi_j)$) are used during clustering, since similarities of tracklet pairs do not change.

We iteratively perform Gibbs sampling of tracklet assignments (to another tracklet) using Eqs. (4) and (7) and the similarity matrix M. At any time of Gibbs sampling, a cluster is defined by tracklets that have been assigned to each other directly or indirectly. After each sampling of a new assignment, previous clusters can be split and/or merged depending on the change of assignment [12]. At the end of Gibbs sampling iterations, we obtain tracklet assignments and clusters/tracks as a byproduct of those.

Since tracklets clustered into a track do not overlap by our definition (because similarities of overlapping tracklets are zero), we output each track by simply ordering the tracklets assigned to it with respect to their timestamp and interpolate for the missing frames accordingly.

The overall pseudocode for tracklet clustering is presented in Algorithm 2.

Input: $\Phi = \{\phi_1, \phi_2 ...\}$ set of tracklets
Result: $\mathbf{C} = \{c_1, c_2 ...\}$ clusters of tracklets, i.e., tracks
// $C(\phi_i)$ denotes cluster that ϕ_i belongs to
$C(\phi_i) = i \ \ \forall i$, $M \leftarrow [\,]$
for $\phi_1 \in \Phi$ **do**
  for $\phi_2 \in \Phi$ **do**
    if $\phi_1 \cap \phi_2 \neq \emptyset$ **then**
      $M[\phi_1, \phi_2] = M[\phi_2, \phi_1] \leftarrow 0$
    else
      $M[\phi_1, \phi_2] = M[\phi_2, \phi_1] \leftarrow F(\phi_1, \phi_2)$
      // Eq. (5)
    end
  end
end
for $\phi_i \in \Phi$ **do**
  Sample $c_i \propto p(c_i|\Phi, c_{-i}, \alpha, M)$ // Eq. (4)
  if $c_i = i$ **then**
    $C(\phi_i) = i$
  else
    $C(\phi_i) = C(\phi_{c_i})$
  end
  for $\phi_k \in \Phi$ **do**
    $C(\phi_k) = C(\phi_{c_k})$ // Update assignments
  end
end

Algorithm 2: Algorithm for tracklet clustering

5 Experiments

We present experimental results on many video sequences from PETS 2009 [13] (person tracking), TownCentre [14]

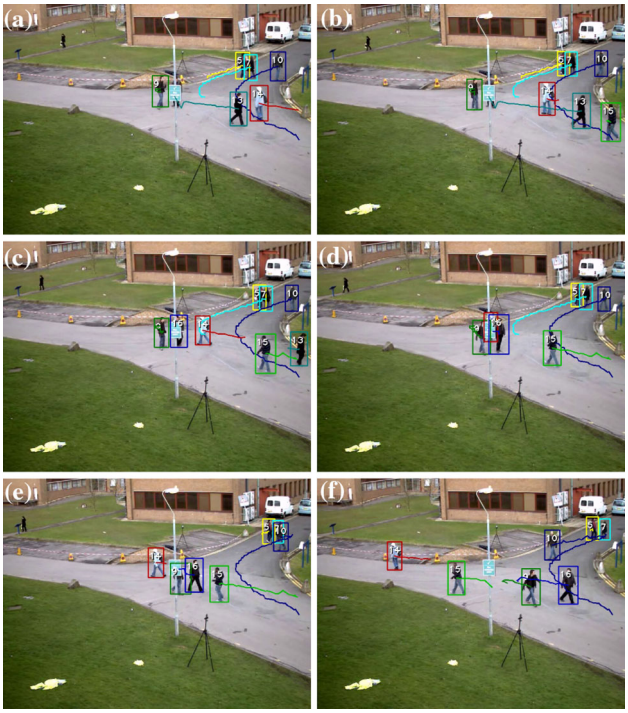

Fig. 2 Samples from PETS 2009 S2-L1,12-34,View-1

(person tracking), SPEVI [15] (face tracking), TUD Stadmitte [16] (person tracking) and ETH [17] (person tracking with moving cameras) datasets.

For tracklet extraction, we use the same object detection outputs with the previous work that we compare our results with in order to perform an unbiased comparison. For SPEVI, we run Haar-like feature-based face detector [18]. We use C# using EmguCV library [19,20]. The code for the overall implementation is available online.[1]

We use the same parameter values for all datasets during tracklet extraction, and we report our clustering results using α values that give the best results for each dataset.

In Figs. 2 and 3, we show sample visual results. We indicate the final clustering/tracking results with distinctly colored and numerically labeled rectangles. At each frame, we also present the trail of the tracklet that constitutes the relevant track for that frame—or no trail if that location was not captured by a detection or a tracklet, and interpolated instead. An example of a single object tracked by many clustered tracklets is the person labeled as 14 and entered the scene from right in Fig. 2a. In Fig. 2c, e, f, trails of the three distinct tracklets, which are eventually clustered into the same track (since they belong to the same labeled person), can be seen clearly. Likewise, persons labeled as 13 (compare Fig. 2b, c) and 15 (compare Fig. 2d, f) had been tracked with at least two distinct tracklets before being clustered into tracks. Examples of total occlusion can be seen in Fig. 2d for the person labeled

[1] http://students.sabanciuniv.edu/~isaygint/sivp15.

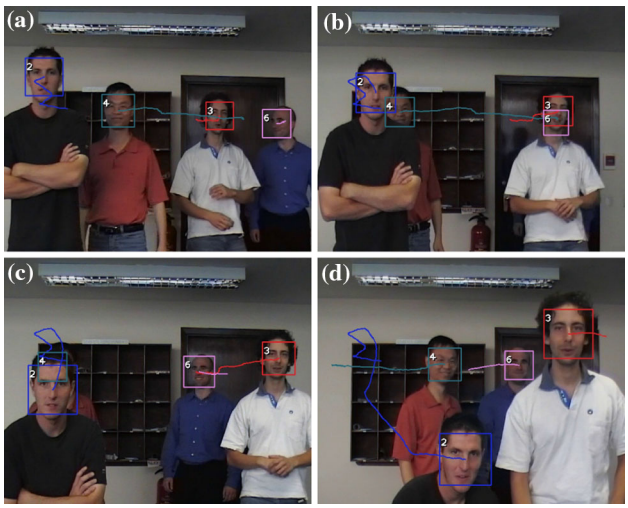

Fig. 3 Sample frames from SPEVI frontal face

as 14, Fig. 2e for the person labeled as 9, Fig. 3b for the face labeled as 6 and Fig. 3c for the face labeled as 4. For all four examples, no tracklets for the tracked objects exist at those frames; however, the tracking processes are not interrupted and continue with the same label.

In Table 1, we give our numerical results and comparisons with the results of [21] for PETS, of [14] for TownCentre, of [3] for SPEVI and of [22] for TUD Stadmitte and ETH datasets using the same ground truths. We report the number of mostly tracked (MT), partially tracked (PT), and mostly lost tracks (ML) [23], maximum online tracking accuracy (MOTA) [24] and precision / recall values [25] using the implementation of [21]. In addition, compared results (rows with cited work) and results obtained with our proposed tracklet clustering scheme (rows with **Proposed**), we also present results for each dataset using only extracted tracklets in Sect. 2 (rows with **Tracklets**) without applying the proposed clustering scheme in order to present the improvement introduced by the proposed tracklet clustering scheme. In addition, we show results using clustering results obtained with the tracklet similarity function without the spatial components (i.e., without x, y, u and v in Eq. (5) in rows with **No spatial**) in order to emphasize the importance of the similarity function and the advantage of our proposed spatial similarity.

We ran our overall tracking algorithm on SPEVI frontal sequence also using face detections of [2] and compared with the ground truth of the same work and able to track all nine tracks in their ground truth with only one identity switch–as opposed to their five *mostly tracked* tracks with ten switches.

We obtain MOTA results with different α values as shown in Fig. 4. For higher α values, the results converge to the values in Table 1 where only tracklets are used without clustering (rows with **Tracklets**), which makes sense since by definition in Eq. (4), higher α values yield in more clusters (i.e.,

Table 1 Tracking results on different datasets

	Prec.	Recl.	MT	PT	ML	MOTA
PETS 2009 S2-L1 12-34 View 1						
[21]	90.8	93.5	18	1	0	83.5
Tracklets	94.2	86.9	16	3	0	78.8
No spatial	92.6	87.5	16	3	0	78.0
Proposed	92.6	92.4	18	1	0	84.5
TownCentre						
[14]	82.0	79.0	–	–	–	61.3
Tracklets	88.0	71.2	92	114	24	59.2
No spatial	85.2	71.7	94	112	24	56.9
Proposed	84.2	78.9	127	87	16	63.5
TUD Stadtmitte						
[22]	96.7	87.0	7	3	0	–
Tracklets	96.7	78.0	6	4	0	72.2
No spatial	90.6	84.4	7	3	0	73.6
Proposed	93.3	88.3	9	1	0	80.8
ETH (Bahnoff and Sunnyday)						
[22]	90.4	79.0	85	31	9	–
Tracklets	91.3	65.7	47	65	12	56.1
No spatial	90.0	64.8	47	66	11	54.1
Proposed	86.9	73.6	63	51	10	61.2
SPEVI frontal face						
[3]	98.0	78.2	0	4	0	75.8
Tracklets	97.6	84.5	4	0	0	81.1
No spatial	93.5	91.3	4	0	0	84.5
Proposed	96.9	91.9	4	0	0	88.7

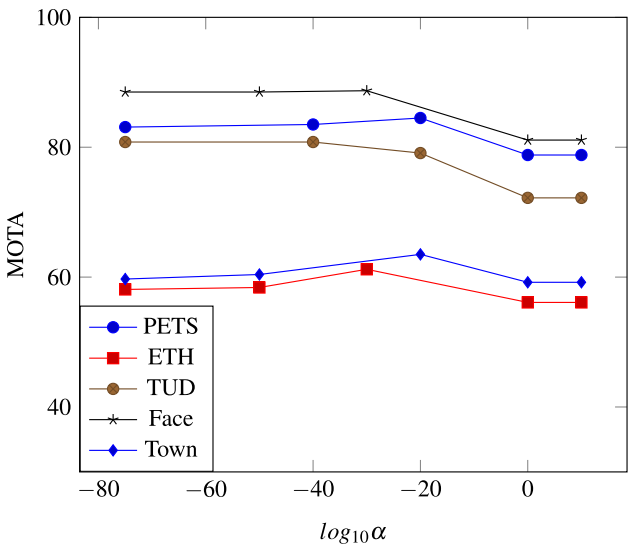

Fig. 4 Change of MOTA with different α values

observations assigned to themselves) eventually yielding no practical clustering where every observation is a cluster by itself alone. For lower α values, where proposed clustering scheme is practically in effect, the results do not change much between different α values which shows that the algorithm is robust to the α parameter.

With an Intel Xeon 2.40 GHz CPU, excluding image I/O and object detections, our algorithm including tracklet extraction and clustering runs at 50 FPS for TUD and ETH sequences being $\sim 5\times$ faster than [22], and 100 FPS for SPEVI sequence being $\sim 10\times$ faster than [3].

6 Conclusions and future work

We presented a tracklet clustering-based object tracker which is robust to occlusions, misses and short tracking errors. We demonstrated qualitative visual results and compared with the state- of-the-art methods. Our main contribution is the tracklet clustering scheme, which does not depend on how the tracklets are extracted or what kind of object detectors are used to extract the tracklets. We define color, spatial and temporal features of tracklets in our work, but the set of features can easily be extended by integrating new features into the tracklet similarity (Eq. 5).

Our results are superior or competitive with state-of-the-art methods except ETH datasets, which may indicate that the proposed algorithm is rather more suitable for stationary cameras. The main advantage of our method is the simplicity of the clustering algorithm, which does not require training complex models or optimizations. This results in the speed of the proposed method being much higher than the compared work. Precision values being higher than the recall in almost all of our results indicate misses during frames, investigation of which is left as future work.

Thanks to their flexible nature, ddCRPs are a promising tool for nonparametric clustering problems where the clusters are complex and cannot be easily modeled by general probabilistic models. Since the tracklet similarities are being calculated once and same similarity values are being used in Gibbs sampling iterations, the speed of the clustering process, even without any special optimization or parallelization, is quite high.

Our algorithm do not use any scene information or semantic tracking, which is still an area for improvement [26]. In the future, we aim for a system that can learn its optimal parameters (like α) from a few training scenes, which will increase the usability of the proposed method without taking run-time parameters into consideration.

References

1. Huang, C., Wu, B., Nevatia, R.: Robust object tracking by hierarchical association of detection responses. In: Computer Vision ECCV 2008, volume 5303 of Lecture Notes in Computer Science, pp. 788–801. Springer, Berlin (2008)

2. Wu, B., Lyu, S., Hu, B.-G., Ji, Q.: Simultaneous clustering and tracklet linking for multi-face tracking in videos. In: 2013 IEEE International Conference on Computer Vision (ICCV), pp. 2856–2863 (2013)
3. Mitra, A., Biswas, S., Bhattacharyya, C.: Temporally coherent CRP: a Bayesian non-parametric approach for clustering tracklets with applications to person discovery in videos. In: SIAM Conference on Data Mining (SDM) (2015)
4. Stauffer, C., Grimson, E.: Adaptive background mixture models for real-time tracking. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2002)
5. Zivkovic, Z.: Improved adaptive gaussian mixture model for background subtraction. In: 2004 IEEE Conference on Pattern Recognition (ICPR) (2004)
6. Qin, Z., Shelton, C.R.: Improving multi-target tracking via social grouping. In: IEEE Conference on Computer Vision and Pattern Recognition (2012)
7. Van der Weken, D., Nachtegael, M., Kerre, E.E..: Some new similarity measures for histograms. In: ICVGIP 2004, Fourth Indian Conference on Computer Vision, Graphics & Image Processing, Kolkata, India, 16–18 Dec 2004, pp. 441–446 (2004)
8. Antoniak, C.E.: Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. Ann. Stat. **2**(6), 1152–1174 (1974). (11)
9. Neal, R.M.: Markov chain sampling methods for Dirichlet process mixture models. J. Comput. Graph. Stat. **9**(2), 249–265 (2000)
10. Aldous, D.J.: Exchangeability and related topics. In: Volume 1117 of Lecture Notes in Mathematics, pp. 1–198. Springer, Berlin (1985)
11. Neiswanger, W., Wood, F., Xing, E.P.: The dependent dirichlet process mixture of objects for detection-free tracking and object modeling. In: Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, 22–25 Apr 2014, pp. 660–668 (2014)
12. Blei, D.M., Frazier, P.I.: Distance dependent Chinese restaurant processes. J. Mach. Learn. Res. **12**, 2461–2488 (2011)
13. PETS2009: Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. ftp://ftp.pets.rdg.ac.uk/pub/PETS2009/
14. Benfold, B., Reid, I.: Stable multi-target tracking in real-time surveillance video. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3457–3464 (2011)
15. Maggio, E., Piccardo, E., Regazzoni, C., Cavallaro, A.: Particle PHD filtering for multi-target visual tracking. In: IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007, vol. 1, pp. I-1101–I-1104 (2007)
16. Andriluka, M., Roth, S., Schiele, B.: Monocular 3d pose estimation and tracking by detection. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 623–630 (2010)
17. Ess, A., Leibe, B., Schindler, K., van Gool, L.: A mobile vision system for robust multi-person tracking. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08). IEEE Press (2008)
18. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2001, vol. 1, pp. I-511–I-518 (2001)
19. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
20. The EmguCV Library. http://www.emgu.com/
21. Milan, A., Schindler, K., Roth, S.: Detection- and trajectory-level exclusion in multiple object tracking. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3682–3689 (2013)
22. Yang, B., Nevatia, R.: An online learned crf model for multi-target tracking. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2034–2041 (2012)
23. Li, Y., Huang, C., Nevatia, R.: Learning to associate: hybridboosted multi-target tracker for crowded scene. In: IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009, pp. 2953–2960. IEEE (2009)
24. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: the clear mot metrics. J. Image Video Process. **2008**, 1:1–1:10 (2008)
25. Smith, K., Gatica-Perez, D., Odobez, J., Ba, S.: Evaluating multi-object tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition—Workshops, 2005. CVPR Workshops, pp. 36–36 (2005)
26. Madrigal, F., Hayet, J.-B., Lerasle, F.: Improving multiple pedestrians tracking with semantic information. Signal Image Video Process. **8**(1), 113–123 (2014)