MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Detecting and Tracking Unknown Number of Objects with Dirichlet process Mixture Models and Markov Random Fields

Topkaya, I.S.; Erdogan, H.; Porikli, F.

TR2013-078 July 2013

Abstract

We present an object tracking framework that employs Dirichlet Process Mixture Models (DPMMs) in a multiple hypothesis tracker. DPMMs enable joint detection and tracking of an unknown and variable number of objects in a fully automatic fashion without any initial labeling. At each frame, we extract foreground super pixels and cluster them into objects by propagating clusters across consecutive frames. Since no constraint on the number of clusters is required, we can track multiple cluster hypotheses at the same time. By incorporating super pixels and an efficient pruning scheme, we keep the total number of hypotheses low and tractable. We refine object boundaries with Markov random fields and connectivity analysis of the tracked clusters. Finally, we group tracked hypotheses to combine possible parts of an object as one.

International Symposium on Visual Computing (ISVC)

Detecting and Tracking Unknown Number of Objects with Dirichlet Process Mixture Models and Markov Random Fields

Ibrahim Saygin Topkaya[1], Hakan Erdogan[1], and Fatih Porikli[2]

[1] Sabanci University VPALAB, Istanbul, Turkey
{isaygint,haerdogan}@sabanciuniv.edu
[2] Mitsubishi Electric Research Laboratories, Cambridge MA, USA
fatihporikli@ieee.com

Abstract. We present an object tracking framework that employs Dirichlet Process Mixture Models (DPMMs) in a multiple hypothesis tracker. DPMMs enable joint detection and tracking of an unknown and variable number of objects in a fully automatic fashion without any initial labeling. At each frame, we extract foreground superpixels and cluster them into objects by propagating clusters across consecutive frames. Since no constraint on the number of clusters is required, we can track multiple cluster hypotheses at the same time. By incorporating superpixels and an efficient pruning scheme, we keep the total number of hypotheses low and tractable. We refine object boundaries with Markov random fields and connectivity analysis of the tracked clusters. Finally, we group tracked hypotheses to combine possible parts of an object as one.

1 Introduction

Conventional object tracking methods often use multiple hypothesis tracking, which can establish correspondence between many hypotheses in parallel [1], or joint probabilistic data association, which can utilize weighted contributions of each observation to update target states [2]. These hypotheses are usually sampled in a particle filtering framework [3] that results in a probabilistic update of the object states. Recently, Dirichlet Process Mixture Models (DPMMs) received increasing attention for tracking applications. For example, DPMMs are adopted for tracking of acoustic energy around a particular frequency in the speech wave for speech recognition [4] and employed a Sequential Monte Carlo (SMC) inference and Gibbs sampler in visual tracking [5].

In this work, we propose a nonparametric model based multiple object tracker for unknown number of targets. Instead of applying batch inference by a Gibbs sampler as in [5], we explore a full-association space inspired by [4] and aim to track all feasible associations. In addition, we select observations using a Gaussian Mixture Model (GMM) based change detection algorithm as opposed to arbitrarily use all image pixels. To reduce the number of tracked association hypotheses, we employ superpixels as atomic observations. After obtaining tracking hypotheses for unknown number of targets, we refine pixel-level boundaries using Markov random fields (MRF) that incorporates the clusters obtained during the previous tracking stages into the refinement process. Finally, we carry out a temporal grouping of clusters to combine different parts of a tracked object into one.

Within the scope of this paper we refer *observations* for any atomic observation to be associated to a target, *targets* for *clusters* (used interchangeably) of observations obtained by DPMM tracking, *objects* for tracked objects that are formed by one or more targets, and *hypotheses* for a tracking hypothesis that defines target states and observation to target associations.

In the next section we give a brief review of DPMMs. In Section 3 we present our object tracking framework using DPMMs followed by the refinement of target boundaries in Section 4 and the grouping of targets in Section 5.

2 Dirichlet Process Mixture Models

DPMMs provide a way to model data as a mixture model having unknown number of mixture components or clusters [6]. Let $X_n; n = 1..N$ be the observed data that is to be modeled as a mixture of distributions having the form $F(\theta)$. If θ_k denotes parameters of the kth mixture component, then $X_n \sim F(\theta_k)$ if $X_n \in k$. Let c_n be the latent indicator variable such that $c_n = k$ indicates $X_n \in k$. The discrete probability distribution $p(c_n = k)$ has Dirichlet distribution as conjugate prior and taking the number of mixture components to infinity results in the Dirichlet process.

As opposed to the GMM and hidden Markov model (HMM) that require the number of habitats to be specified prior to training, DPMM has the appealing property that the number of mixtures or clusters does not need to be known a priori. It assumes that there are infinite number of mixture components $k = 1..\infty$ yet only a finite number of these components have observations assigned to them. Modeling the data with DPMMs consists of finding the parameters of those finite and unknown number of mixture components, i.e. clusters. A detailed review of Markov chain sampling methods (like Gibbs sampling) to estimate the cluster parameters can be found in [7]. These methods iterate over all observations and for each observation calculate probabilities of belonging to an existing or a new cluster controlled by an aggregation parameter α. For higher values of α more clusters are generated during the clustering. The probability that an observation belongs to a mixture component is given as

$$p(c_n; \alpha) = \begin{cases} \frac{N_k}{N+\alpha-1}\ p(X_n|\theta_k) & \text{existing k,} \\ \frac{\alpha}{N+\alpha-1}\ \int_\theta p(X_n|\theta)\ d\theta & \text{new cluster,} \end{cases} \tag{1}$$

where N_k is the number of assignments to cluster k and N is the number of all observations. The graphical model for the DPMMs is depicted in Fig. 1.

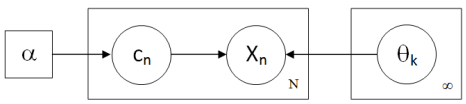
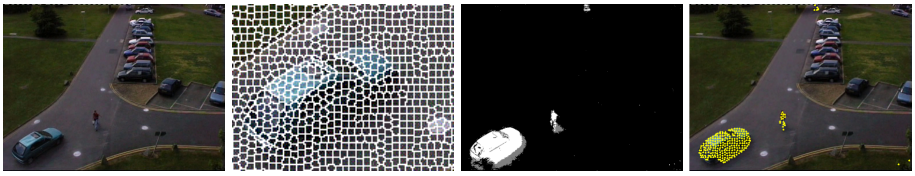


Fig. 1: Graphical model for the DPMMs: The observation (X_n) depends on one of the infinite number of cluster parameters (θ_k), assignment (c_n) of which is controlled by α



(a) (b) (c) (d)

Fig. 2: A sample frame (a), superpixel borders for the bottom-left part of the frame (b), the foreground probability map for each pixel (c) and centers of superpixels that contain foreground pixels (d).

3 Object Tracking with DPMMs

3.1 Extracting Observations for Tracking

To reduce the number of observations and decrease the DPMM association time, we incorporate superpixels that segment an image into small, compact and almost-regular regions while keeping color variation within regions low. With a small computational overload, a rough superpixel extraction [8] significantly decreases the number of observations by around *95%*, an example of which can be seen in Fig. 2.

We extract foreground regions on the frame using GMM based background representation [9] that models the previous color changes of each pixel using a mixture of Gaussians by applying an expectation maximization update. Here, we select superpixels that contain pixels whose foreground probability is high. For the sample frame shown in Fig. 2, the number of observations decrease from *∼160,000* to *∼4,500* foreground pixels and then finally to around *200* foreground superpixels.

3.2 Observation and Target Models

We model each observation using the spatial center of the superpixel (i.e. x and y pixel coordinates) and the mean value of the a and b pixel color components in the Lab color space. Similarly, we model each cluster/target using the mean and variance of the same components along with the spatial covariance.

For computational ease, we do not model Gaussian models with full covariance matrices for targets. Instead, we analyze the covariance between spatial components since it is strongly related to the appearance of the target on the image by approximating the target appearance with a rotated ellipse (for human tracking). Thus, each observation is defined with four parameters; $X : (\mu_x, \mu_y, \mu_a, \mu_b)$ and each target with six parameters; $\theta : (\mu_{xy}, \Sigma_{xy}, \mu_a, \sigma_a, \mu_b, \sigma_b)$. Under this model, the likelihood that an observation X_n is generated by a target k with parameters θ_k is

$$p(X_n|\theta_k) = \mathcal{N}(X_{xy}|\mu^k_{xy}, \Sigma^k_{xy})\ \ \mathcal{N}(X_a|\mu^k_a, \sigma^k_a)\ \ \mathcal{N}(X_b|\mu^k_b, \sigma^k_b) \tag{2}$$

where the parameters of the Gaussians in Eq. 2 are estimated from the observations that are assigned to the targets. Here, μ_{xy} models the spatial center and μ_a and μ_b the

average color values for the targets. The spatial covariance Σ_{xy} models the size and orientation of the target in the image. The larger the covariance the more spread in that direction the target becomes.

Eq. 1 and Eq. 2 together define the assignment probability of an observation to an existing or a new target. For a new cluster, the integral in Eq. 1 is calculated over the whole prior distribution. The prior for Gaussian distribution is Normal-Inverse Wishart distribution and integrating over it gives a t-distribution [10]. However, [10] shows that this can be approximated by a Gaussian with properly chosen parameters. We choose it as a Gaussian that is centered on the frame and having a variance that covers the whole frame. The color components have a similar coverage.

3.3 Target Assignment and Tracking with DPMM Clustering

In [5] two inference methods for tracking are defined; one depends on Markov Chain Monte Carlo (MCMC) to perform batch inference and the other uses SMC in a particle filtering framework. At each frame, iterative Gibbs sampling is performed and assignments are selected for each observation. Then, parameters of each cluster are again sampled from the current and past assignments. In [4], at each time and for each observation, a whole exploration of the assignment space is done in a Rao-Blackwellized [11] fashion and new hypotheses are generated for each assignment instead of Gibbs sampling.

We follow a similar approach. After initially estimating the positions of the targets using past motion dynamics (i.e. *Rao-Blackwellization*), we evaluate the observations one by one and calculate association probabilities of observations to an existing or a new target with Eq. 1. Each association represents a new hypothesis with a calculated weight.

For each frame f, the DPMM clustering inherits K_{f-1} number of clusters from the previous frame and performs clustering of the new observations to those existing (or new) clusters. Note that, some of the inherited clusters may be kept with new observation assignments, some of them may be dropped if no observations are assigned to them, and some new clusters may be generated. Altogether, they form K_f number of clusters as the tracking result for frame f. These clusters are projected to the next frame $f+1$ later.

We prune association hypotheses with very low weights after evaluating each observation, which prevents the number of the hypotheses to grow. In our experiments we have observed that much less than 10 hypotheses are kept between frames.

Our association scheme differs from [5] in the sense that the whole assignment space is explored and cluster parameters are updated deterministically instead of random sampling of the assignments and cluster parameters.

A difference between [4] and our method is the weight update rule of the hypotheses. We consider transition probabilities of the clusters while updating the weights of the hypotheses, which allows us after evaluating all observations in a frame, to remove the hypotheses that have unusual change in the states of the clusters. In [4], after each observation (X_n) is assigned to a target, a new hypothesis is derived weight of which (w_h) is updated as:

$$w_h = w_h \; p(c_n = k; \alpha). \tag{3}$$

We also perform the same update rule while generating new hypotheses during handling the observations. In addition, for one frame, after all observations are assigned to targets and target parameters are updated, we calculate the following transition probabilities for each cluster and update the weight of the hypothesis:

$$w_h = w_h \prod_{k \in h} p(\theta_k^f | \theta_k^{f-1}). \tag{4}$$

The transitions are calculated for clusters inherited from previous frame $(f-1)$ and kept in current frame (f). Addition of new clusters is controlled by the parameter α in Eq. 1, and at this stage there is no special handling for the deletion of the clusters, which we leave as a future work. The transition probability in Eq. 4 is taken as:

$$\begin{aligned} p(\theta_k^f | \theta_k^{f-1}) = \mathcal{N}(\mu_{xy}^f | \mu_{\hat{x}\hat{y}}^f, \Sigma_{xy}^{f-1}) \times \\ \mathcal{N}(\sigma_x^f | \sigma_x^{f-1}, 0.1\ \sigma_x^{f-1}) \times \\ \mathcal{N}(\sigma_y^f | \sigma_y^{f-1}, 0.1\ \sigma_y^{f-1})\ , \end{aligned} \tag{5}$$

where $\hat{x}$ and $\hat{y}$ denote the initial spatial estimates of the positions of the targets estimated from their previous motions with the Rao-Blackwellization. Considering also that the variance of a target is proportional to its size, the first probability in Eq. 5 represents the typical assumption that the position of the tracked target conforms with the past dynamics with an uncertainty proportional to its size. The latter two probabilities represent the assumption that the size of the tracked target changes at most around *10%* of its size between frames.

4 Refining Object Boundaries

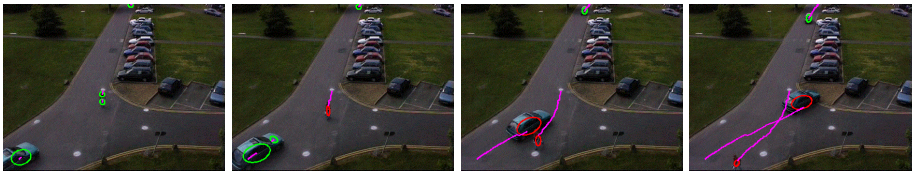
Tracking by the DPMM clustering scheme presented in Section 3.3 generates the labeled foreground superpixels, where the labels correspond to tracked clusters/targets. To compensate border artifacts caused by the quick but rough superpixel extraction we apply a refinement step.

MRFs [12] are commonly used graphical models in image labeling tasks to obtain smooth maps. The labeling is considered as an optimization problem where energies for labeling of the image are defined for individual pixels within local neighborhoods on uniform grid.

The energies for neighborhoods are used to enforce smoothness in the local regions. Having a graphical model where nodes $\mathfrak{n}$ correspond to pixels and vertices υ to neighborhoods, the aim is to find the lowest overall energy E of a labeling $\mathcal{L}$ for image $\mathcal{I}$, which is calculated as sum of unary and pair-wise energies as

$$E(\mathcal{L}) = \sum_{u \in \mathfrak{n}} E(\mathcal{L}_u) + \sum_{(u_1, u_2) \in \upsilon} E(\mathcal{L}_{u_1}, \mathcal{L}_{u_2}), \tag{6}$$

where $E(\mathcal{L}_u)$ is the cost of labeling individual pixels (unary term) and $E(\mathcal{L}_{u_1}, \mathcal{L}_{u_2})$ is the cost of labeling neighboring pixels (pair-wise term), which is used to enforce



(a) (b) (c) (d)

Fig. 3: Four sample frames from PETS 2001 dataset. Ellipses are the isocontours of the Gaussian distribution (for σ=1) that defines the target position (Section 3.2). Green ellipses correspond to single targets, red ellipses correspond to two or more merged/grouped targets (Section 5). Pink trajectories denote the past spatial centers of the relevant targets.

smoothness. In our work, we label each pixel in the frame as one of the targets obtained with the DPMM tracking and the background. During that process, we employ MRFs as with any other labeling task [12].

The DPMM clustering results in target clusters for a frame and for each pixel. Using position and color of the pixel, Eq. 2 can be used to calculate the likelihood of the pixel for each target. To impose this likelihood as the unary term for a single pixel X_n in Eq. 6, we take the negative log of Eq. 2:

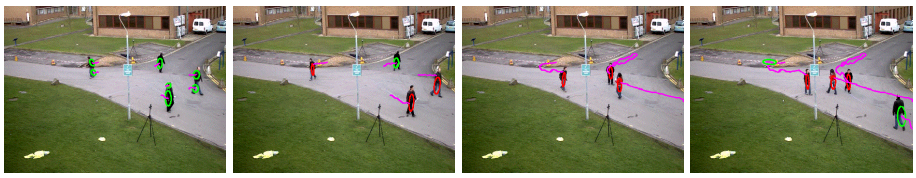
$$E(\mathcal{L}_{X_n}) = -\log(p(X_n|\theta_k)), \tag{7}$$

For the background label, we again take negative log of the background probability obtained with the background subtraction results produced in Section 3.1 to detect foreground superpixels. For the pair-wise term in Eq. 6, we set 8-pixel neighborhoods and give a fixed energy value if the neighboring pixels have different labels where same labels incur zero penalty.

5 Grouping Targets into Objects

We empirically obtained results with different α values (of Eq. 1) and observed that even with suitable values there may be cases that parts of an object may be assigned to different targets because of color differences. Thus, we run a final step to detect and group those different parts of an object into one object, and present the final tracking output by representing objects as those grouped targets.

To decide whether any two targets can be merged into one, we analyze the historical motion of the targets by measuring the similarity of the motions of the pairs of targets. To measure the similarity, we use cross-correlation of historical spatial (x and y components) positions of the targets.



(a) (b) (c) (d)

Fig. 4: Sample frames from PETS 2009 dataset. Ellipses are the isocontours of the Gaussian distribution (for σ=1) that defines the target position (Section 3.2). Green ellipses correspond to single targets, red correspond to two or more merged/grouped targets (Section 5). Pink trajectories denote the past spatial centers of the relevant targets.

Cross-correlation for x component of any two targets historical positions of which are known is calculated as

$$\rho_x = \frac{\sum_t (x_1(t) - \mu_1^x)(x_2(t) - \mu_2^x)}{\sqrt{\sum_t (x_1(t) - \mu_1^x)^2} \sqrt{\sum_t (x_2(t) - \mu_2^x)^2}}, \tag{8}$$

and similarly for ρ_y. At each frame, we calculate ρ_x and ρ_y using last t frames after tracking with DPMMs is achieved. In case their sum exceeds a threshold value for any two targets, we assume those targets move together, thus belong to the same object. Such targets are merged into the same object.

6 Experiments and Results

We implemented the proposed algorithm in C#. For superpixel extraction, we used SLIC superpixels [8]. We integrated the implementation in VLFeat library [13] for smoothing with MRFs, which comes with a FastPD MRF optimization [14, 15]. To extract the foreground pixels, we applied the GMM implementation of OpenCV [16]. Object boundaries were obtained with α-shapes [17] implementation of CGAL library [18]. We run the experiments on a sequence of 200 frames in PETS 2001 [19] and 100 frames in PETS 2009 [20] datasets where α in Eq. 1 is fixed ($\alpha = 1$) for the proposed method.

6.1 Tracking and Target Grouping Results

Figures 3 and 4 present the tracking results by denoting the tracked targets as 2D Gaussian ellipses drawn on the image frame, as well as their past tracks superimposed onto the image.

The results demonstrate that the proposed tracking algorithm works accurately in complex situations where some tracked objects are partially occluded by others like in the last two sample frames from the PETS 2009 sequence. In Fig. 4c, it can be seen that

the pedestrian that entered the scene from right passes in front of the pedestrian that was walking in the middle of the scene from the beginning. The proposed tracker continued to track those two pedestrians in the following frames (Fig. 4d) without having any drift problems.

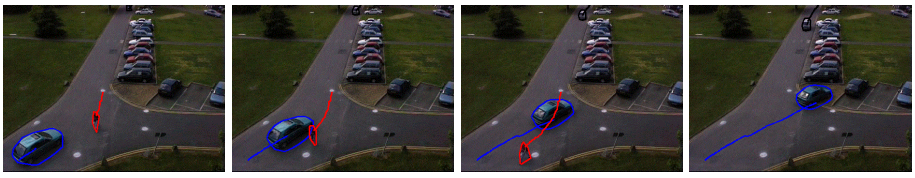
The figures also show the success of the proposed target grouping scheme. For example, initially at Fig. 3a, parts of the pedestrian in the middle of the scene are detected and being tracked as separate targets, as well as the car in Fig. 3b. After 10 frames, using the proposed target grouping scheme, parts of the targets that belong to the same object are grouped and assigned as one (Fig. 3c and Fig. 3d). In all figures, the red ellipses denote the targets that are obtained by grouping two or more targets.

Similarly, initially at Fig. 4a, parts of three pedestrians are detected and being tracked as separate targets. After 10 frames, parts of the pedestrians are grouped and represented as a single object (Fig. 4b). Again, the grouped targets are denoted by red ellipses in the figure.

6.2 Refining Object Boundaries

Figures 5 and 6 display results in which accurate object boundaries are drawn rather than ellipses. After target clusters and superpixel assignments are obtained, a finer labeling of pixels is obtained with MRFs.

The refinement step provides pixel-wise assignments for targets. Using those assignments, boundaries are calculated as sets of pixels that represent the border contains all pixels assigned to the target cluster. Note that, borders are not necessarily convex. While the boundaries are being determined, pixel assignments that are far from the target cluster are filtered out where the cut-off distance is determined by the standard deviation of the spatial components of the target cluster (i.e. σ_x and σ_y). The results show that even in complex situations that objects come together (e.g. Fig. 6c); the boundaries of them can be detected accurately by taking the grouping of the targets into consideration.



(a) (b) (c) (d)

Fig. 5: Sample object boundaries (Section 4) from PETS 2001 dataset. The target borders are denoted with distinct colors across frames, as well as the historical centers as the superimposed trajectory having the same color with the target borders.

	Proposed	Neiswanger [5]
ATA	**78.07**	29.93
SPF	**9.91**	18.08

Table 1: Tracking accuracy (100×ATA) and running time (SPF) for PETS 2001 sequence comparing the proposed method and [5] (best results are in bold).

6.3 Quantitative Analysis

We report the *average tracking accuracy* (ATA) scores given as [21]:

$$STDA = \sum_{i=1}^{N_o} \frac{\sum_{f=1}^{N_f} \frac{G_i^f \cap D_i^f}{G_i^f \cup D_i^f}}{N_{(G_i^f \cup D_i^f)\neq 0}} \quad , \quad ATA = \frac{STDA}{(\frac{N_G + N_D}{2})} \tag{9}$$

which is the normalization of *sequence track detection accuracy* (STDA) by the number of detected (N_D) and ground truth objects (N_G). STDA is calculated using overlaps and unions pixels that belong to the ground truth (G_i^f) and detected (D_i^f) objects. This *overlap ratio* is aggregated over the sequence of N_f frames and normalized by the number of frames ($N_{(G_i^f \cup D_i^f)\neq 0}$) where a ground truth or detected object exists, and calculated for all N_o objects.

In Table 1, we report the ATA values of the proposed method for the PETS 2001 sequence that contains three distinct objects along with the accuracy of the method proposed in [5]. For both methods, we filter clutter by removing targets that appear less than five frames. For our method, we take the target merges into consideration. For [5], we repeat the experiments with different α and *covariance confidence*, which is a specific parameter used for output representation in [5], and report the best result. We also report the average running times to process one frame, i.e. *seconds per frame–SPF* in the same table. The results demonstrate that the proposed method significantly outperforms [5]. The primary reason is that our method calculates target parameters deterministically as opposed to probabilistic sampling approach in [5]. In addition, we

(a) (b) (c) (d)

Fig. 6: Sample object boundaries (Section 4) from PETS 2009 dataset. The target borders are denoted with distinct colors across frames, as well as the historical centers as the superimposed trajectory having the same color with the target borders. Some targets in (a) *merged* (Section 5) into others in the following frames.

	SP size=9	SP size=25	SP size=100
ATA	77.98	**78.07**	62.57
SPF	28.27	9.91	**5.88**

(a)

	Clean	Noisy
ATA	**78.07**	77.96
SPF	**9.91**	13.21

(b)

Table 2: Tracking accuracy ($100\times$ATA) and running time (SPF) of the proposed method for PETS 2001 sequence; (a) comparing the different superpixel sizes, (b) comparing noisy observations.

extract the foreground pixels by a more robust background generation method instead of the simple frame differences as in [5].

In Table 2-a, we give the tracking accuracy scores of our method for different average superpixel sizes. The results show that the superpixel size plays an important role for both tracking accuracy and running time.

As seen in the first two columns, increasing superpixel size significantly decreases the running time by reducing the number of observations. However, using large superpixels (as in the third column) may negatively impact the tracking accuracy too. The reason is that as superpixels get larger, there is risk of grouping some pixels of the background and nearby targets into same superpixels, thus losing the distinction between targets and deforming their boundaries at the observation level that causes tracking drift problems. For tracking accuracy, it is not always preferable to use smaller superpixels either as the optimal size for the ATA is around 25.

In Table 2-b, we report tracking accuracies when a detection noise is added by adding a random number of false observations. The number of false observations are controlled such that during foreground extraction, each background superpixel is chosen falsely as with some particular probability, which was set as 0.001 for the presented result. The effect of noise can be seen on the running time. Noisy observations introduces false targets, which involves more calculations in the process of the observation-to-target assignment during the generation of tracking hypotheses. Table 2-b shows that the tracking accuracy changes minimally for noisy observations, which indicates that the proposed method is very robust to observation errors.

7 Discussion and Future Work

We use DPMMs in visual object tracking in a deterministic multiple-hypotheses tracking framework. DPMMs allow us to detect and track unknown number of object in a fully automatic fashion, without any initial labeling required. Since our method is based on superpixels and incorporates an efficient pruning step, the number of hypotheses does not grow in memory and is tractable. It also achieves refinement of object boundaries with MRFs while employing a target grouping step to compensate clustering errors.

In the future we plan to extend MRF refinement by enforcing different higher order constraints such as shape and histograms to incorporate object specific priors into segmentation.

References

1. Reid, D.B.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control **24** (1979) 843–854
2. Fortmann, T., Bar-Shalom, Y., Scheffe, M.: Multi-target tracking using joint probabilistic data association. (1980) 807–812
3. Arulampalam, M.S., Maskell, S., Gordon, N.: A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Transactions on Signal Processing **50** (2002) 174–188
4. Özkan, E., Özbek, I.Y., Demirekler, M.: Dynamic speech spectrum representation and tracking variable number of vocal tract resonance frequencies with time-varying dirichlet process mixture models. IEEE Transactions on Audio, Speech & Language Processing **17** (2009) 1518–1532
5. Neiswanger, W., Wood, F.: Unsupervised Detection and Tracking of Arbitrary Objects with Dependent Dirichlet Process Mixtures. ArXiv e-prints (2012)
6. Antoniak, C.E.: Mixtures of Dirichlet Processes with Applications to Bayesian Nonparametric Problems. The Annals of Statistics **2** (1974) 1152–1174
7. Neal, R.M.: Markov chain sampling methods for dirichlet process mixture models. Journal Of Computational And Graphical Statistics **9** (2000) 249–265
8. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: SLIC Superpixels Compared to State-of-the-art Superpixel Methods. IEEE Transactions on Pattern Analysis and Machine Intelligence (2012) 1
9. Stauffer, C., Grimson, E.: Adaptive background mixture models for real-time tracking. In: Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on. (2002)
10. Sudderth, E.: Graphical Models for Visual Object Recognition and Tracking. PhD thesis, Massachusetts Institute of Technology (2006)
11. Casella, G., Robert, C.P.: Rao-blackwellisation of sampling schemes. Biometrika **83** (1996) 81–94
12. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 2001
13. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/ (2008)
14. Komodakis, N., Tziritas, G.: Approximate labeling via graph cuts based on linear programming. IEEE Trans. Pattern Anal. Mach. Intell. **29** (2007) 1436–1453
15. Komodakis, N., Tziritas, G., Paragios, N.: Performance vs computational efficiency for optimizing single and dynamic mrfs: Setting the state of the art with primal-dual strategies. Comput. Vis. Image Underst. **112** (2008) 14–29
16. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
17. Bernardini, F., Bajaj, C.L.: Sampling and reconstructing manifolds using alpha-shapes. In: In Proc. 9th Canad. Conf. Comput. Geom. (1997) pages
18. CGAL: Computational Geometry Algorithms Library. (http://www.cgal.org)
19. PETS2001: Second IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. (ftp://ftp.pets.rdg.ac.uk/pub/PETS2001/)
20. PETS2009: Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance. (ftp://ftp.pets.rdg.ac.uk/pub/PETS2009/)
21. Kasturi, R., Goldgof, D., Soundararajan, P., Manohar, V., Garofolo, J., Bowers, R., Boonstra, M., Korzhova, V., Zhang, J.: Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. IEEE Trans. Pattern Anal. Mach. Intell. **31** (2009) 319–336