A Sampling Algorithm for Tracking Multiple Objects

Hai Tao, Harpreet S. Sawhney, and Rakesh Kumar

Sarnoff Corporation
201 Washington Rd., Princeton NJ 08543
{htao,hsawhney,rkumar}@sarnoff.com

Abstract. The recently proposed CONDENSATION algorithm and its variants enable the estimation of arbitrary multi-modal posterior distributions that potentially represent multiple tracked objects. However, the specific state representation adopted in the earlier work does not explicitly supports counting, addition, deletion and occlusion of objects. Furthermore, the representation may increasingly bias the posterior density estimates towards objects with dominant likelihood as the estimation progresses over many frames. In this paper, a novel formulation and an associated CONDENSATION-like sampling algorithm that explicitly support counting, addition and deletion of objects are proposed. We represent all objects in an image as an object configuration. The *a posteriori* distribution of all possible configurations are explored and maintained using sampling techniques. The dynamics of configurations allow addition and deletion of objects and handle occlusion. An efficient hierarchical algorithm is also proposed to approximate the sampling process in high dimensional space. Promising comparative results on both synthetic and real data are demonstrated.

1 Introduction

Tracking multiple objects in videos is a key problem in many applications such as video surveillance, human computer interaction, and video conferencing. It is also a challenging research topic in computer vision. Some difficult issues involved are cluttered background, unknown number of objects, and complicated interaction between objects. Many tracking algorithms can be interpreted in a probabilistic framework called hidden Markov model (HMM) [1], explicitly or implicitly.

As shown in Fig.1, the states of an object $x_t \in X$ at different time instances $t = 1,2,\ldots n$ form a Markov chain. State x_t contains object deformation parameters such as positions and scale factors. At each time instance t, conditioned on x_t, observation z_t is independent of other previous object states or observations. This model is summarized as

$$P(x_1, x_2, \ldots x_n; z_1, z_2, \ldots z_n) = P(x_1)P(z_1 \mid x_1)\prod_{i=2}^{n}[P(x_t \mid x_{t-1})P(z_t \mid x_t)] \quad (1)$$

The tracking problem can be posed as the computation of the *a posteriori* distribution $P(x_t \mid \mathbf{Z}_t)$ for given observations $\mathbf{Z}_t = \{z_1, z_2, \ldots, z_t\}$. When a single object is tracked, the maximum *a posteriori* (MAP) solution is desired. If both the object dynamics

$P(x_t \mid x_{t-1})$ and the observation likelihood $P(z_t \mid x_t)$ are Gaussian distributions, $P(x_t \mid \mathbf{Z}_t)$ is also a Gaussian distribution. The MAP solution is $E(x_t \mid \mathbf{Z}_t)$.

In order to compute $P(x_t \mid \mathbf{Z}_t)$, a forward algorithm [1] is applied. It computes $P(x_t \mid \mathbf{Z}_t)$ based on $P(x_{t-1} \mid \mathbf{Z}_{t-1})$ inductively and is formulated as

$$\begin{aligned} P(x_t \mid \mathbf{Z}_t) &\propto P(z_t \mid x_t) P(x_t \mid \mathbf{Z}_{t-1}) \\ &= P(z_t \mid x_t) \int P(x_t \mid x_{t-1}) P(x_{t-1} \mid \mathbf{Z}_{t-1}) dx_{t-1} \end{aligned} \tag{2}$$

Using this formula, the well-known Kalman filter that computes $E(x_t \mid \mathbf{Z}_t)$ for a Gaussian process can be derived [2]. When multiple objects present, if the number of objects is fixed and the posterior of each object is Gaussian, similar solution in analytic form is obtained. If the number of objects may change over time, data association method such as multiple-hypothesis tracking (MHT) [3] has to be used. The complexity of MHT algorithm is exponential with respect to the time and pruning techniques are necessary for real applications [4].

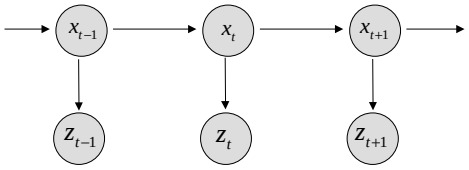


Fig. 1. The hidden Markov model.

When the analytic form of either $P(x_t \mid x_{t-1})$ or $P(z_t \mid x_t)$ is not available, sampling techniques such as the CONDENSATION algorithm [5] are preferred. The idea is to represent $P(x_t \mid \mathbf{Z}_t)$ with samples and to propagate the posterior distribution over time by computing the likelihood function $P(z_t \mid x_t)$ and simulating the dynamics $P(x_t \mid x_{t-1})$. In [6], a variance reduction method called importance sampling algorithm is used to reduce the number of samples and to handle data associate problems. A more recent paper [8] deal with fixed number of object using a sampling scheme.

The original CONDENSATION algorithm and its variants use a single object state as the basic state representation. Presence of multiple objects is *implicitly* contained in the multiple peaks of the posterior distribution. When the CONDENSATION algorithm is applied to such a representation, it is very likely that a peak corresponding to the dominant likelihood value will increasingly dominate over all other peaks when the estimation progresses over time. In other words, a *dominant peak* is established if some objects obtain larger likelihood values more frequently. If the posterior is propagated with fixed number of samples, eventually, all samples will be around the dominant peak. Dominant peak may occur in many model based tracking algorithms. For example, a head-shoulder contour deformable model may fit one person better than another in most frames of a video sequence. This phenomenon is further illustrated here with a synthetic example.

Fig. 6 shows two frames of the synthetic sequence. More details of the sequence can be found in Section 6. In Fig. 9, the tracking results using the original CONDENSATION algorithm are illustrated. Since the likelihood function is biased to certain objects, the differences between these objects and the other objects in the posterior distribution increase exponentially with respect to the number of frames observed. In frame 15 (Fig. 9b), three peaks can be identified. In frame 25 (Fig. 9c), one object looses most of its samples because of its constantly relatively smaller likelihood. . In frame 65 (Fig. 9d), another object vanishes due to its smaller likelihood. This phenomenon can also be observed in Fig. 9e and Fig. 9f.

Besides the dominant peak problem, the above example also illustrates that the events such as addition, deletion, and occlusion can not be naturally handled. In Fig. 9d, a new object appears but no samples are allocated to it. In Fig. 9h, an object disappears, but the samples are not redistributed to the other object.

Importance sampling [6] is a data-driven mechanism that may alleviate some of the above problems. However, in order to maintain and update the count and state of multiple objects explicitly, a new representation is required.

It should be noticed that the limitation described here is not of the CONDENSATION process but of the state representation that is used by the tracker. In this paper we present a new representation and apply a CONDENSATION-like sampling algorithm for the estimation of the joint distribution of multiple objects under the presence of clutter, varying object counts and appearance/disappearance of objects.

2 Tracking Multiple Objects

Our goal is to (i) track multiple instances of an object template, (ii) maintain an expected value of the number of objects at any time instant, and (iii) be resilient to clutter, occlusion/deocclusion and appearance/disappearance of objects. In order to be able to represent multiple objects, we enhance the basic representation by representing all objects in the image as an *object configuration* (the term configuration is used in the rest of this paper for conciseness). A configuration is represented by a set of object deformation parameters $s_t = \{x_{t,1}, x_{t,2}, \ldots, x_{t,m}\} \in X^m$, where m is the number of objects. If K is the maximum possible number of objects in an image, the configuration space is $\bigcup_{m=0}^{K} X^m$. Given the enhanced representation, the goal is to compute the *a posteriori* probability of the configuration parameters $P(s_t \mid \mathbf{Z}_t)$ instead of the *a posteriori* probability of object parameters $P(x_t \mid \mathbf{Z}_t)$. The posterior for a configuration is given by

$$\begin{aligned} P(s_t \mid \mathbf{Z}_t) &\propto P(z_t \mid s_t) P(s_t \mid \mathbf{Z}_{t-1}) \\ &= P(z_t \mid s_t) \int P(s_t \mid s_{t-1}) P(s_{t-1} \mid \mathbf{Z}_{t-1}) ds_{t-1} \end{aligned} \tag{3}$$

To estimate this distribution, the configuration dynamics $P(s_t \mid s_{t-1})$ and the configuration likelihood $P(z_t \mid s_t)$ need to be modeled. Then a CONDENSATION-like sampling algorithm can be applied. Distribution $P(s_t \mid s_{t-1})$ describes the temporal behavior of a configuration in terms of how each of the individual objects changes, how a new object is introduced, how an existing object is deleted, and how to

handle occlusion. The likelihood $P(z_t \mid s_t)$ measures how well the configuration fits the current observation.

2.1 Dynamics of a Configuration - $P(s_t \mid s_{t-1})$

$P(s_t \mid s_{t-1})$ is decomposed into object-level and configuration-level dynamics. Suppose s_{t-1} contains m objects, or $s_{t-1} = \{x_{t-1,1}, x_{t-1,2}, \ldots, x_{t-1,m}\}$. Object-level dynamics $P(\bar{x}_{t,i} \mid x_{t-1,i})$ is first applied to predict the behavior of each object. The resulted configuration is $\bar{s}_t = \{\bar{x}_{t-1,1}, \bar{x}_{t-1,2}, \ldots, \bar{x}_{t-1,m}\}$. Then, the configuration-level dynamics $P(s_t \mid \bar{s}_t)$ will perform the object deletion and addition.

2.1.1 Object-level Dynamics $P(\bar{x}_{t,i} \mid x_{t-1,i})$

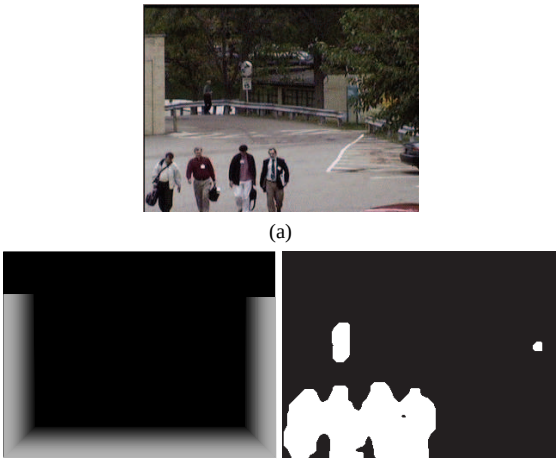
A commonly used model is:

$$\bar{x}_{t,i} = \mathbf{A} x_{t-1,i} + w \tag{4}$$

where $w : N(0, \Sigma)$ is a Gaussian noise and $\mathbf{A}$ is the state transition matrix. According to this model, $P(\bar{x}_{t,i} \mid x_{t-1,i})$ has Gaussian distribution $N(\mathbf{A} x_{t-1,i}, \Sigma)$.

2.1.2 Configuration-level Dynamics - $P(s_t \mid \bar{s}_t)$

The configuration-level dynamics should allow deletion and addition of objects in $\bar{s}_t$. Domain-dependent information should be brought in to model these events. For instance, knowledge about deletion and addition can be described as spatial birth and death processes [9].



(a) (b) (c)

Fig. 2. Configuration-level dynamics: (a) a video frame with static background (b) deletion probability $\beta(x, y)$ (c) motion blobs.

Deletion probability $\beta(x, y)$ is defined as a function of the image coordinates (x, y). For example, $\beta(x, y)$ may have higher values around the scene boundaries because objects usually disappear at those locations. For an object at (x, y), its chance of survival in the current frame is $1-\beta(x, y)$. When occlusion happen in an area with low deletion probability, the occluded object is unlikely to be deleted.

By the same token, addition probability is defined as $\alpha(x, y)$. Since new objects always cause image changes, motion blobs are used to construct $\alpha(x, y)$. For video with static background, motion blobs are detected by image differencing method. $\alpha(x, y)$ is non-zero only in the regions of the motion blobs. For the case of a pan/tilt or moving camera, the blob detection may be accomplished using background alignment techniques and change detection algorithms [10].

In Fig. 2a, a frame from a test video clip is shown. Fig. 2b shows the deletion probability $\beta(x, y)$. The highest value in the image is around the border. The motion corresponding blobs are shown in Fig. 2c. The addition probability $\alpha(x, y)$ is 0.01 in these blobs.

2.2 Likelihood of a Configuration $\pi_t = P(z_t \mid s_t)$

$P(z_t \mid s_t)$ is a very complicated distribution. One possible type of approximation is observation decomposition [7]. The image is spatially decomposed into small regions and the likelihood is formulated as the product of local likelihood. Since the configuration is not decomposed, it will lead to algorithms manipulating in a high-dimensional configuration space. In this paper, we propose an approximation using configuration decomposition. The likelihood is replaced by an energy function and decomposed into object-level and configuration-level terms. The energy function is designed to gives the more desired configurations higher values. Intuitively, three factors should be considered. The first factor is, in average, how well individual objects in a configuration fit the observation. This is noted as the object-level likelihood. For example, a contour matcher may be applied to calculate the likelihood of each object in a configuration, and their geometric average is computed as the object-level likelihood for that configuration. The average is taken to make it independent of the number of objects. The second factor is how much of the observation has been explained by the configuration. This is noted as the configuration coverage. The third factor is the compactness. It is always desirable to explain the observation using minimum number of objects. All these three factors are indispensable. In Fig. 3, the likelihood of some configurations is illustrated.

2.2.1 Object-level Likelihood

For a given object, the likelihood $L(z_t, x_{t,i})$ measures how well the image data supports the presence of this object. The likelihood can be defined as any reasonable match measures, e.g. the normalized correlation between the object and the image, or the Chamfer matching score for a contour representation of an object. For a

configuration with m objects, the object-level likelihood is computed as the geometric average of $L(z_t, x_{t,i})$. More precisely,

$$\lambda = \left(\prod_{i=1}^{m} L(z_t, x_{t,i}) \right)^{\frac{1}{m}}. \tag{5}$$

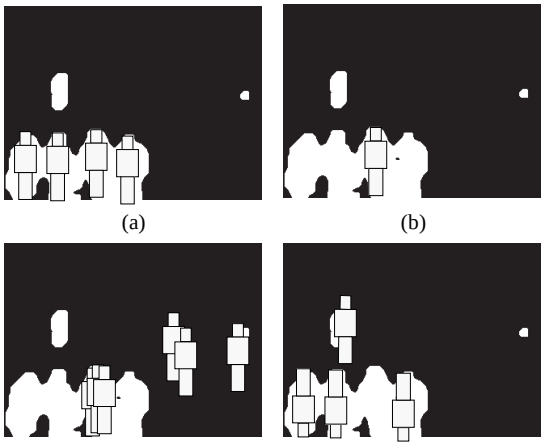


Fig. 3. Likelihood of a configuration (a) highest (b) low: interested region is not covered (c) low: too many object are used to explain the data (d) low: likelihood of individual objects are low.

2.2.2 Configuration Coverage

In general, it is difficult to compute configuration coverage. However, for moving object tracking, motion blobs are good cues. If we assume all the motion blobs in a frame are caused by the objects to be tracked, the configuration coverage can be computed as the percentage of the motion blob areas being covered by objects. It is formulated as

$$\gamma = \frac{|A \cap (\bigcup_{i=1}^{m} B_i) + b|}{|A| + b} \tag{6}$$

where A is the union of motion blobs. B_i is the area covered by object *i* in a configuration. b is a small positive constant used to avoid zero division. If $|A| = 0$, $\gamma = 1$.

2.2.3 Configuration Compactness

The compactness is defined as the ratio between data that has been explained and the amount of cost. In terms of motion blobs, it can be computed as

$$\xi = \frac{|A \cap (\bigcup_{i=1}^{m} B_i) + c|}{(|\bigcup_{i=1}^{m} B_i| + a)} \tag{7}$$

where a is a small positive constant like b. If too many objects are used to explain a small area, ξ will be small. c is a positive number so that when $|A| = 0$, the configurations with smaller number of objects have higher score.
Finally, the overall likelihood of configuration s_t is approximated by

$$\pi_t = \lambda \cdot (\gamma\xi)^{\beta} \tag{8}$$

where β, a positive constant that controls the relative importance of the last two terms. It should be mentioned that, depending on the application, different cues may be used to compute the configuration coverage and compactness. For instance, color blobs with skin colors can be applied for face tracking.

3 A Sampling Algorithm

Given the above formulation of configuration dynamics and likelihood, we now present a CONDENSATION-like algorithm to estimate the *a posteriori* configuration densities. Subsequently, we show how the standard CONDENSATION algorithm can be approximated using a fast hierarchical algorithm.

Suppose $\pi_t^j = P(z_t \mid s_t^j)$, $j = 1,2,\ldots R_s$ is the likelihood of the j th configuration s_t^j, where R_s is the total number of configuration samples. R_s is a constant in the algorithm.

For j from 1 to R_s, perform the following three steps.

Step 1. At time instance $t > 1$, randomly select the *j*th configuration sample s'^j_{t-1} from all R_s samples s^i_{t-1}, $i = 1,2,\ldots R_s$ in the previous frame according to their corresponding likelihood $\pi^i_{t-1}, i = 1,2,\ldots R_s$.

Step 2. Apply the dynamics to predict the current configuration s_t^j from s'^j_{t-1} using $P(s_t^j \mid s'^j_{t-1})$

Step 3: Compute the new likelihood $\pi_t^j = P(z_t \mid s_t^j)$

To initialize this process, s_1^j is sampled randomly in the configuration space $\bigcup_{m=0}^{K} X^m$. For example, if the maximum possible number of objects in a configuration is $K = 9$ and 1000 configuration samples are initiated ($R_s = 1000$), then for the 10 categories of configurations that contain 0 to 9 objects, 100 samples are assigned to each category. For a configuration sample with m objects, the parameters of each object are randomly chosen in the parameter space. The configuration likelihood is then computed. If the likelihood of a configuration is high, according to Step 1, in the next iteration, this configuration is likely to be selected. The expected

number of objects in a frame can also be computed as $\sum_{j=1}^{R_s} |s_t^j| \pi_t^j$, where $|s_t^j|$ is the number of objects in s_t^j.

The above algorithm samples the *a posteriori* distribution of the configurations in a high dimensional space $\bigcup_{m=0}^{K} X^m$. If there are m objects in the scene, the posterior has to be sampled in the space X^m. To maintain the same sample density, the number of samples needs to be exponential with respect to m, which makes the algorithm impractical. Importance sampling techniques [6] alleviate the problem to some extent by reducing the volume of the parameter space X, however, the dimensionality of the sampling space is not reduces. A possible solution to this problem is to sample from configurations with high likelihood. More specifically, in the first step, s'^j_{t-1} is only drawn from s^i_{t-1} with relatively large π^i_{t-1}. This strategy makes the sampling process focus on the posterior distribution around the MAP solution, which is desirable because the goal of the tracking process is to actually obtain the MAP configuration. A problem of this method is that the tracker is easily trapped by local maximum solutions.

4 An Efficient Hierarchical Sampling Algorithm

In this section, we describe an efficient hierarchical algorithm that decouples the sampling process into two stages: local configuration sampling stage and global configuration sampling stage. The local sampling stage track the motion of individual objects, while the configuration sampling process handles object addition, deletion. Strictly speaking, it does not propagate the configuration posterior distribution. It reinforces the likelihood portion to some extent so that the tracker is less likely to be trapped by local optimal solutions. To explain the algorithm more clearly, examples will be provided for each step of the algorithm.

The first step is selecting new configuration samples based on the previous samples and their corresponding likelihood (see Section 3). For example, in Fig. 4a, four configurations are selected. They contain two, three, four, and four objects respectively. There are total of thirteen objects in these four configurations. Different shapes are used in the figure to distinguish objects in different configurations.

The second step is local sampling of the object-level *a posteriori* distribution conditioned on given configurations. More specifically, the image is first partitioned into non-overlapping regions and configurations are broken into *sub-configurations* according to the partition. For example, in Fig. 4b, the configuration marked by "△" is decomposed into three sub-configurations in region 2, 3, and 4. The sub-configuration in region 4 contains two objects. In region 4, there are three other sub-configurations containing 1, 1, and 2 objects respectively. After the partitioning, in each region, object-level dynamics is applied to every object and likelihood is computed for each sub-configuration (Fig. 4c). Note that the configuration-level dynamics such as object deletion and object addition is not performed in this step.

Next, in each image region, all sub-configurations with the same number of objects are grouped together. According to their likelihood, they are sampled to produce the same

number of new sub-configurations. These samples are then assigned back to the global configurations randomly (because there is no identity left after sampling). For example, in region 4, based on the two resulted two-object sub-configurations in Fig. 4c and their corresponding likelihood, sampling process is applied to obtain two "new" sub-configurations (Fig. 4d). Actually, these two sub-configurations are identical because the sub-configuration with higher likelihood has been selected twice. The resulted configurations are assigned arbitrarily back to the global configuration.

In the third step, configuration-level dynamics computation (see Section 2.1.2) is applied and likelihood is computed for each configuration (see Section 2.2). Fig. 4e shows the result after configuration-level dynamics being applied. A new "⌂" object is added and a "o" object is deleted.

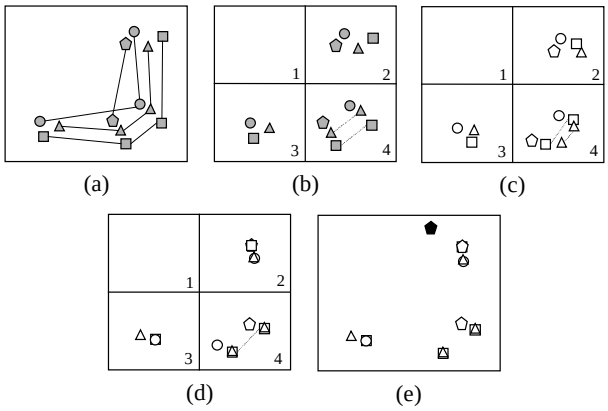


Fig. 4. The hierarchical sampling algorithm for tracking multiple objects. (a) select configurations (b) partition configurations into sub-configurations (c) local object-level sampling (d) recover configurations from new sub-configurations (e) global configuration-level dynamics and likelihood computation.

The hierarchical tracking algorithm is summarized as follows:

Step 1. Select configurations: At time $t > 1$, select R_s configuration samples. The *j*th configuration sample s'^{j}_{t-1} is select randomly from all R_s samples s^{i}_{t-1}, $i = 1,2,\ldots R_s$ in the previous frame according to their likelihood π^{i}_{t-1}, $i = 1,2,\ldots R_s$.

Step 2. Local object-level sampling: Partition the 2D image into regions and break configurations into sub-configurations. In each region, apply object-level dynamics. For sub-configurations containing the same number of objects, do sampling according to their local configuration-level likelihood. Assign them randomly back to the global configurations.

Step 3: Global configuration-level sampling: The configuration-level samples are recovered. The likelihood $\pi^{j}_{t} = P(z_t \mid s^{j}_{t})$ is computed. Go to the next frame.

5 Implementation

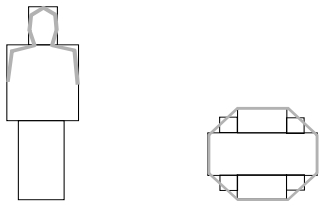
The proposed hierarchical algorithm has been implemented on a Pentium II 400 MHz PC. It runs at 1 frame/s when 300 configuration samples are used on 320x240 video frames.

5.1 Video Preprocessing

Detecting motion blobs is an important step for computing configuration-level likelihood. Several methods such as background subtraction, two-image or three-image differencing algorithms are available. Three-image differencing method is used in our implementation [10].

5.2 Object Representation

As shown in Fig. 5, a contour-plus-region representation is designed. To track multiple people, the head-shoulder contour in Fig. 5a is compared with the edge images in order to obtain object likelihood $L(z_t, x_{t,i})$. The contour template is divided into several line segments. $L(z_t, x_{t,i})$ is computed as the weighted average of the matching score for individual template contour segments. The regions of the template are represented by rectangles and are used to compute γ and ξ using Equation (6) and (7). The parameter β, which controls the relative importance of the object-level likelihood and the configuration-level likelihood, equals 1.5.



(a) (b)

Fig. 5. (a) A simple contour-region representation of people, (b) a coarse 2D contour-region representation of spherical objects.

5.3 The Hierarchical Algorithm

A fixed number of configuration samples are used in the algorithm. These samples are evenly distributed to configurations with different number of objects at the initialization stage. For the first frame, several iterations of the algorithm are executed to obtain the initial prior (with different dynamics). The size of each local image region in our implementation is 10×10 pixels.

6 Experimental Results

6.1 The Synthetic Sequence

Both a synthetic image sequence and natural video data are tested. The synthetic sequence contains four moving objects of similar shapes (Fig. 6). They approximate a circle with six, seven, eight, and thirty laterals. These objects undergo only translations in this test sequence. A translation invariant object-level likelihood function is computed based on a generic contour model and a contour matching algorithm. The likelihood values of these four objects remain consistent over time and have small differences due to their different shapes. This setup resembles many model based trackers in the way that a generic model (built either by learning or designing) is used to track an entire class of objects. These objects enter and leave the scene at the image boundaries. There is one instance of object occlusion in this sequence.

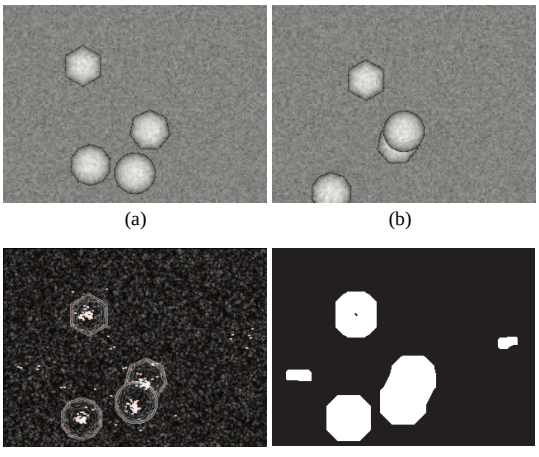


Fig. 6. (a)(b) Two frames in the synthetic sequence (c) the edge map and the tracking result (white dots are object samples) (d) motion blobs.

The background image is formed by Gaussian noise. To simulate some random irrelevant moving objects, white noise is added to the background at two locations that gives rise to some spurious motions blobs. Finally, noise is added to the appearance of the moving objects. Quantitative analysis is conducted based on the tracking results and the actual number and positions of objects in each frame.

For the synthetic sequence, we compared the results of the CONDENSATION algorithm and the hierarchical algorithm. In the CONDENSATION algorithm, 600 object samples are used. In the latter one, 300 configuration samples are initialized. These 300 samples are evenly distributed in terms of number of objects in a configuration and object parameter values.

In Fig. 9, the tracking results of the original CONDENSATION algorithm with a single object state representation are shown. Importance sampling is used in the first frame to obtain a better prior. Object samples are represented by white dots. In Fig. 9i, marginal sample distributions on vertical image axis for every five frames are

shown. As explained in Section 1, dominant peaks and inappropriate handling of object addition and deletion are observed.

In Fig. 10, the corresponding results of the hierarchical tracking algorithm with the multiple object representation are demonstrated. Four distinct trajectories are observed in Fig. 10i. Events such as addition, deletion, and occlusion can be easily distinguished.

As mentioned in Section 3, by applying the new representation, expected number of objects in each frame is computed from configuration samples. In Fig. 7, the expected number of objects in each frame using the hierarchical algorithm is shown. In the same figure, the actual number of objects is also drawn. (The first 20 iterations are used the algorithm initialization and are not significant in the comparison). The number of objects in most of frames is correctly estimated, even during the occlusion period.

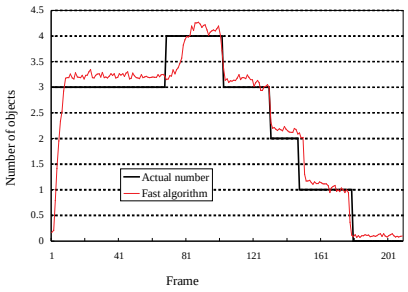


Fig. 7. Object counts in the synthetic sequence.

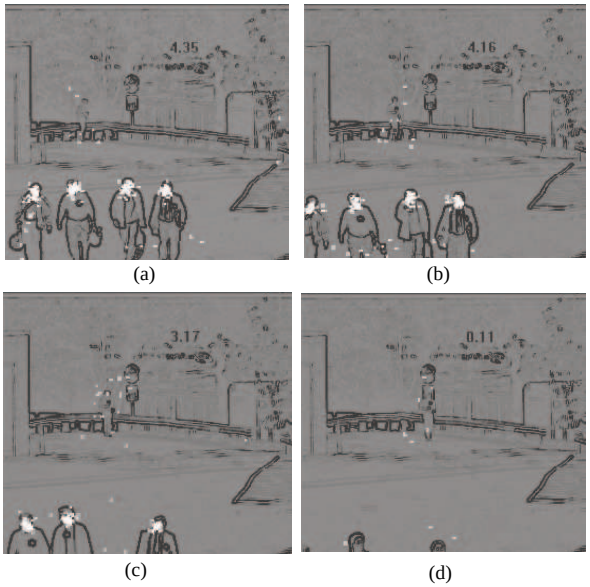


Fig. 8. The results of tracking multiple people.

6.2 Tracking Multiple People

Both algorithms have been tested on real video sequences. For tracking multiple people, a simple contour-plus-region template is designed (Fig. 5a). Only translation is modeled in the transformation. A frame is shown in Fig. 2a. Its corresponding motion blobs are shown in Fig. 2c. For this particular sequence, deletion is only allowed in the gray regions drawn in Fig. 2b. Fig. 8 demonstrates the tracking results in some frames. Four persons are simultaneously tracked. The number of persons in the scene is automatically estimated in the hierarchical algorithm.

7 Conclusions

The new representation proposed in this paper explicitly models multiple objects in a video frame as an object configuration. The events such as object addition, deletion, and occlusion are modeled in configuration-level dynamics. With this formulation, CONDENSATION-like tracking algorithms can be designed to propagate the configuration posterior. A hierarchical sampling algorithm is also proposed in this paper. Promising comparative experimental results of the CONDENSATION algorithm and the new algorithm on both synthetic and real data are demonstrated. Compared to the multiple-hypothesis tracking method, which is an approximation of the Viterbi algorithm based on local maximums of likelihood function, the proposed algorithm explores the likelihood function in the whole parameter space. However, the concept of configuration tracks needs to be introduced to fully model the data association over time.

References

[1] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257-286, Feb. 1989.

[2] Z. Ghahramani and G. E. Hinton, "Parameter estimation for linear dynamical systems," Technical Report CRG-TR-96-2, Univ. of Toronto, 1996.

[3] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Automatic Control*, vol. 24, no. 6, pp. 843-854, Dec. 1979.

[4] I. J. Cox, S. L.Hingorani, "An efficient implementation of Reid's multiple hypothesis tracking algorithm and tts evaluation for the purpose of visual tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, no. 2, pp. 138-150, Feb. 1996.

[5] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proc. European Conf. on Computer Vision*, pp. 343-356, Cambridge UK, 1996.

[6] M. Isard and A. Blake, "ICONDENSATION: unified low-level and high-level tracking in a stochastic framework," in *Proc. European Conf. on Computer Vision*, pp. 893-908, 1998.

[7] J. Sullivan, A. Blake, M. Isard, and J. MacCormick, "Object localization by Bayesian correlation," *Proc. Int. Conf. Computer Vision*, 1999.

[8] J. MacCormick and A. Blake, "A probabilistic exclusion principle for tracking multiple objects," *Proc. Int. Conf. Computer Vision*, 1999.

[9] N. A. C. Cressie, *Statistics for Spatial Data*, John Wiley & Sons Inc., 1991.

[10] A. Selinger and L. Wixson, "Classifying moving objects as rigid or non-rigid without correspondences," *Proc. DARPA Image Understanding Workshop*, pp. 341-347, Monterey, CA, Nov. 1998.

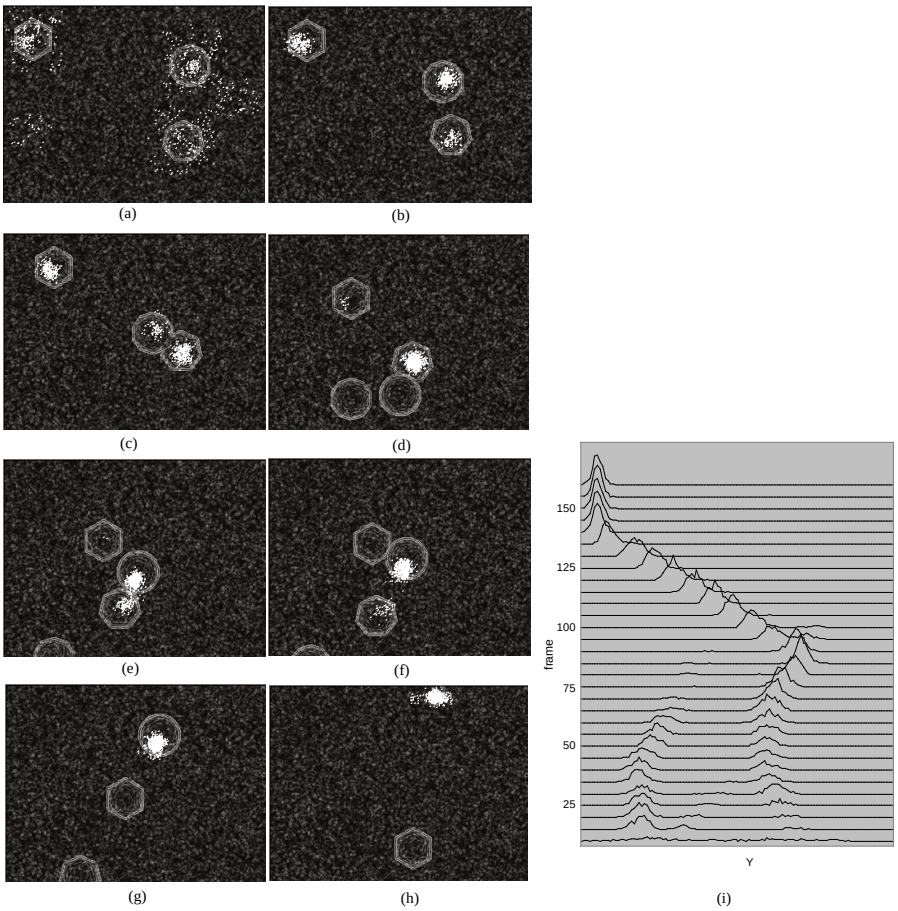


Fig. 9. Results of the CONDENSATION algorithm in frame (a) 1 (b) 10 (c) 25 (d) 65 (e) 85 (f) 90 (g) 110 (h) 140 and (i) the marginal sample distribution along the vertical image axis. Left side is the top of the images.

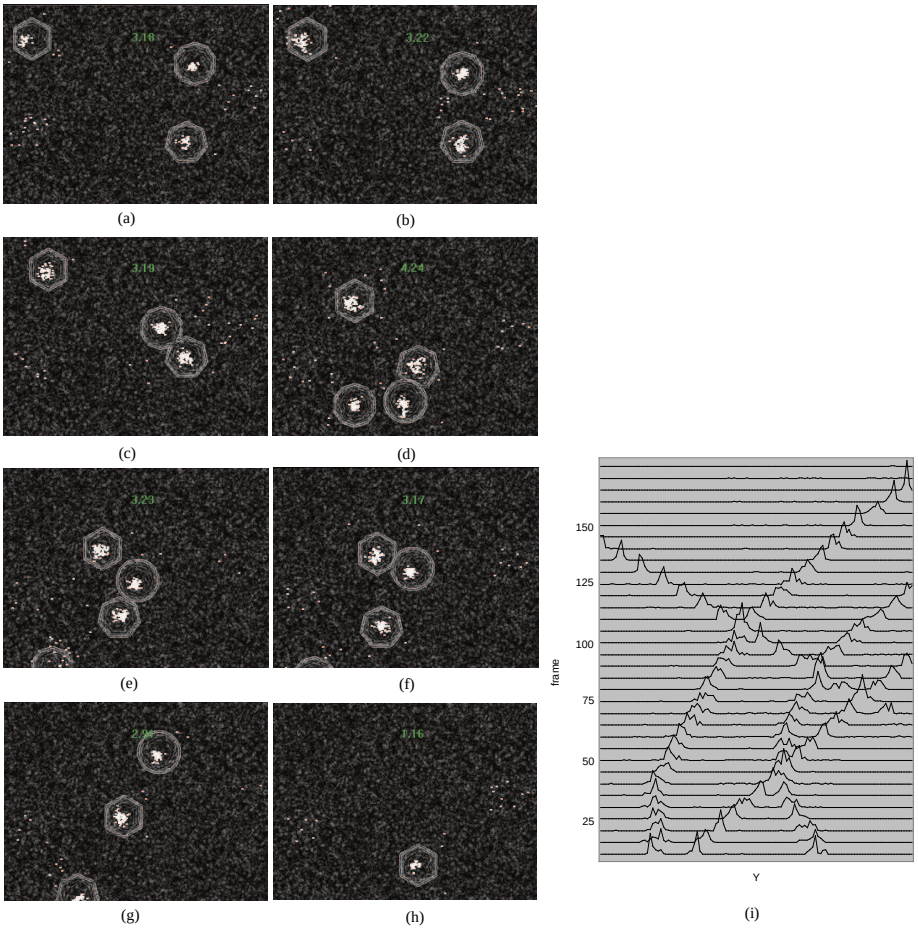


Fig. 10. Results of the hierarchical algorithm in frame (a) 1 (b) 10 (c) 25 (d) 65 (e) 85 (f) 90 (g) 110 (h) 140 (i) and the marginal sample distribution along the vertical image axis. Left side is the top of the images.