Bi-directional Tracking using Trajectory Segment Analysis

Jian Sun Weiwei Zhang Xiaoou Tang Heung-Yeung Shum
Microsoft Research Asia, Beijing, P. R. China
{jiansun, weiweiz, xitang, and hshum}@microsoft.com

Abstract In this paper, we present a novel approach to keyframe-based tracking, called *bi-directional tracking*. Given two object templates in the beginning and ending keyframes, the bi-directional tracker outputs the MAP (Maximum A Posterior) solution of the whole state sequence of the target object in the Bayesian framework. First, a number of 3D *trajectory segments* of the object are extracted from the input video, using a novel trajectory segment analysis. Second, these disconnected trajectory segments due to occlusion are linked by a number of inferred occlusion segments. Last, the MAP solution is obtained by trajectory optimization in a coarse-to-fine manner. Experimental results show the robustness of our approach with respect to sudden motion, ambiguity, and short and long periods of occlusion.

1 Introduction

Visual tracking is one of the fundamental problems in computer vision. Given the *observations*, i.e. a video sequence, tracking infers the *states* of the target object(s). Applications range from video surveillance, human-computer interfaces, and augmented reality to digital video editing.

Most tracking approaches work in a recursive way: estimating object location at the current time t based on the observations up to time t. In a Bayesian framework, the tracking problem is commonly formulated as a recursive estimation of a time-evolving posterior distribution $P(x_t|y_{1:t})$ of state x_t given all the observations $y_{1:t}$. Recursive estimation has two major advantages: 1) it is efficient in computation, and 2) it naturally fits into real-time or *online tracking* applications.

Many real world applications such as event statistics in video surveillance, object-based video compression, home video editing, video annotation, and visual motion capture can be regarded as *offline tracking* where all the frames from the input video sequence can be used. In offline tracking, moreover, a long video sequence can be decomposed into short ones by specifying a few keyframes, which is also called *keyframe-based tracking*. Each keyframe contains an object template which can be given by hand or by using some automatic object detection methods.

To utilize the information from these keyframes, a straightforward method is to apply the recursive approach from keyframes going forward or backward. One problem of this approach is that when tracking fails in the middle of the sequence, we have to add another keyframe at the failed location. However, it is very difficult to predict when the method may fail, thus we have to add the keyframe in a trial-and-error manner which is prohibitively time consuming. The second problem is that the recursive method only uses information in one keyframe while ignoring information in the other keyframe.

Recent work on rotoscoping [1] tracks the contours in video for animation using user-specified contours in two or more frames. Rotoscoping makes full use of the information in the keyframes to improve the performance of contour tracking. However, rotoscoping is limited to tracking only parameterized curves, which is difficult to apply to other tracking applications.

In this paper, we develop a *bi-directional tracking* algorithm of generic objects by taking advantage of the information in both keyframes. Formally, given a video sequence and two states x_1 and x_T in the beginning and ending keyframes, we compute the MAP solution of the whole state sequence:

$$P(x_{2:T-1}|y_{1:T}, x_1, x_T) \sim P(y_{1:T}|x_{1:T})P(x_{2:T-1}|x_1, x_T) \quad (1)$$

The success of our algorithm depends on whether it can overcome the following two challenges.

One challenge is to provide an efficient optimization algorithm to obtain the MAP solution. In visual tracking, the whole continuous state sequence space usually has an enormous number of local minimums due to nonlinear dynamics and non-gaussian observations. Gradient-based methods will often become stuck at a local minimum. The MAP solution can be also computed by Viterbi algorithm using a discrete HMM (Hidden Markov Model) representation. However, the the quantized state space is very large even for a simple state representation for a 320×240 video.

The other challenge is to handle partial or complete occlusions. Short-time occlusions can often be handled by an appropriate dynamics model. However, for more complex occlusions, such as long-time occlusions or occlusions by similar objects, previous methods often fail. How to handle various difficult occlusions using the information in two keyframes is of both theoretical and practical interest in the bi-directional tracking.

In order to overcome the above difficulties, our bi-directional tracking uses a novel trajectory segment representation. Trajectory segments are a number of small fractions of possible object trajectories in the 3D video volume.

Trajectory segments are extracted from the input video using a spectral clustering method. With this representation, the MAP solution can be efficiently obtained in a coarse-to-fine manner by a discrete HMM model. More important, at the trajectory segment level, we propose an occlusion reasoning algorithm to robustly infer possible occlusion trajectory segments of the target object.

2 Previous Work

Tracking remains a very difficult vision problem due to several reasons, for example sudden motion, ambiguity and occlusion. The sudden motion of object may be caused by unexpected dynamic changes of the object itself or abrupt camera motion. When the target object comes close to a similar object, tracking algorithms often fail to locate the correct one due to ambiguity. The target object may be partially or completely occluded. Occlusion can be of short or long. A number of approaches have been proposed to alleviate these problems.

Direct optimization The direct optimization approaches [12, 2, 7, 4] estimate the motion parameters between two neighboring frames by minimizing a deterministic cost function. The direct optimization approach assumes slow motion between two frames. This kind of approach is efficient but not very robust in situations with rapid sudden motion, ambiguity, and long-time occlusion.

Particle filtering Condensation [10] is the first particle filtering [6, 11] based algorithm introduced in visual tracking. Particle filtering approximates the posterior distribution using a set of "weighted particles". The particle filtering algorithm has advantages on handling sudden motion and short-time occlusion. However, it often difficult to handle ambiguity or long-time occlusion. Maccormick & Black proposed a "probabilistic exclusion principle" [13] to address the ambiguity problem. But their approach is limited to a special observation model for contour tracking.

Offline tracking Offline tracking exploits all the information in the video sequence. In [9], the optical flow over the entire sequence is estimated simultaneously using a rank constraint on the rigid motion. Torresani & Bregler [17] track 3D points using a low rank constraint on a 3D morphable model and importance sampling in trajectory space. Multiple hypothesis tracking (MHT) was proposed by Reid [16] and improved by Cox & Hingorani [5] for multiple objects tracking. They give a Bayesian formulation for determining the probabilities of measurement-to-target association hypotheses. Recent work in [8] optimizes a MAP solution of the joint trajectories of objects for multiple object tracking. Their approach severely relies on background substraction and object detection, and no explicit occlusion reasoning mechanism is presented.

3 Framework

In this paper, we chose a very basic state model and observation model to demonstrate our bi-directional tracking approach in the keyframe-based framework.

State The target object is represented as a rectangle $R = \{p, s*\widehat{w}, s*\widehat{h}\}$, where p is the center rectangle and s is the scaling factor. $\widehat{w}$ and $\widehat{h}$ are a fixed width and height of the object template, respectively. So, we denote the state of the object as $x = \{p, s\} \in \mathcal{X}$, where $\mathcal{X}$ is the state space. In the bi-directional tracking, the state x_1 in the first keyframe I_1 and the state x_T in the last keyframe I_T are known.

Observation The observation is the color statistics of the target object. The object's color model is represented as a histogram $\mathbf{h} = \{h_1, ..., h_H\}$ with H (typically, $H = 8 \times 8 \times 8$) bins in RGB color space. The Bhattacharyya distance between the associated histogram $\mathbf{h}(x_0)$ of the state x_0 and the associated histogram $\mathbf{h}(x_i)$ of the state x_i is defined as: $B^2[\mathbf{h}(x_0), \mathbf{h}(x_i)] = 1 - \sum_{j=1}^{B} \sqrt{h_j(x_0) h_j(x_i)}$. This model only captures global color statistics. A more sophisticate multi-part color model [15] can be used if there is a certain spatial configration of the target object.

Trajectory Optimization The posterior of the whole state sequence $X = \{x_2, ..., x_{T-1}\}$ for a given video sequence $Y = \{y_1, ..., y_T\}$ and known two states $\{x_1, x_T\}$ can be represented as follows under the first order Markov independence assumption:

$$P(X|Y, x_1, x_T) = \frac{1}{Z} \prod_{i=2}^{T-1} \psi(y_i|x_i, x_1, x_T) \prod_{i=1}^{T-1} \psi(x_i, x_{i+1}), \tag{2}$$

where the local evidence $\psi(y_i|x_i, x_1, x_T)$ is defined using the Bhattacharyya distance:

$$\psi(y_i|x_i, x_1, x_T) \sim \exp(-\min\{B^2[\mathbf{h}(x_i), \mathbf{h}(x_1)], B^2[\mathbf{h}(x_i), \mathbf{h}(x_T)]\}/2\sigma_h^2), \tag{3}$$

where σ_h^2 is the variance parameter. It measures the similarity between the color histogram $\mathbf{h}(x_i)$ of the state x_i to the closest color histogram between $\mathbf{h}(x_1)$ in the keyframes I_1 or $\mathbf{h}(x_T)$ in I_T. The potential function $\psi(x_i, x_{i+1})$ between two adjacent states is defined as:

$$\psi(x_i, x_{i+1}) \sim \exp(-D[x_i, x_{i+1}]/2\sigma_p^2), \tag{4}$$

where $D[x_i, x_{i+1}] = ||p_i - p_{i+1}||^2 + \beta||s_i - s_{i+1}||^2$ is the similarity between state x_i and x_j. σ_p is a variance to control the strength of smoothness and β is a weight between location difference and scale difference. It is a smoothness constraint on the whole trajectory of the target object.

The goal of the bi-directional tracking is to obtain the MAP solution of Equation (2). To efficiently perform the optimization and handle possible occlusion, we present a novel approach based on trajectory segment analysis. Figure 1 shows the basic flow of our approach:

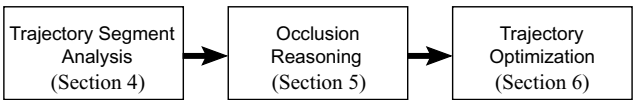


Figure 1: Flowchart of bi-directional tracking.

1. Trajectory segment analysis. For a given video sequence and object templates in two keyframes, trajectory segment analysis extracts a number of small 3D trajectory segments in the video volume using a spectral clustering method.

2. Occlusion reasoning. To handle both short-time and long-time occlusions, we connect disjointed trajectory segment pairs where an occlusion segment may exist in between.

3. Trajectory optimization. A number of discrete states in each frame are sampled from the segments obtained in step 2. The MAP solution of Equation (2) is obtained by a discrete HMM model in a coarse-to-fine manner.

4 Trajectory Segment Analysis

Trajectory segment analysis consists of two steps: *2D mode extraction* in each frame independently and *3D trajectory segment extraction* in the whole video simultaneously.

4.1 2D mode extraction

The purpose of 2D mode extraction is to significantly reduce the whole state space so that further analysis on a sparse state set is tractable. For each frame, we can compute an evidence surface using Equation (3). The 2D modes are peaks or local maxima on this surface. A 2D mode represents a state x' whose observation is similar to the object templates in the keyframes. Namely, the local evident $\psi(y|x', x_1, x_T)$ is high.

To efficiently find these modes, we adopt the mean shift [4] algorithm which is a nonparametric statistical method seeking the nearest mode of a point sample distribution. Given an initial location, mean shift can compute the gradient direction of the convoluted evidence surface by a kernel G [4]. With this property, the mean-shift algorithm is a very efficient iterative method for gradient ascent to a local mode of the target object.

To perform 2D mode extraction, we uniformly sample the location in the image and the scale (3-5 discrete levels) to obtain a set of starting states. The spatial sampling interval is sightly smaller than half the size of the object. Then, the mean shift algorithm runs independently from each starting state. After convergence, we get a number of local modes. Finally, we reject the state mode x' whose local evidence $\psi(y|x', x_1, x_T) \leq 0.5$ and merge very close modes to generate a sparse set of local 2D modes in each frame, as shown in the bottom row of Figure 2.

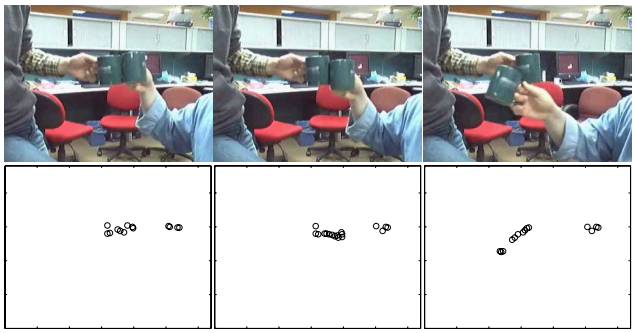


Figure 2: 2D mode extraction. Top: three frames in the **"cup"** sequence. Bottom: each black circle is a 2D mode whose local evidence is high. For example, in the first column, the left cluster corresponds to two green cups in the image and the right two circle corresponds to a background region with similar color statistics of the object templates in the keyframes.

4.2 3D trajectory segment extraction

Figure 3 shows the "circles" example containing two occlusions (one arm occludes a cup from frame 16 to 35, and from frame 98 to 132) and one ambiguity (two cups meet from frame 52 to 86). Figure 4(a)(b) shows the all extracted 2D mode points in a 3D volume. We found that the "true" object trajectory is a curved structure, which may contain discontinuities caused by occlusion or is very close to other "irrelevant" points due to ambiguity. It is not trivial to extract it at the point level. Therefore, we first extract a number of trajectory segments from all points. An ideal segment extraction should have two properties: 1) a segment represents a fraction of a "meaningful" trajectory, and 2) the length of each segment is as long as possible or the number of segments is small. In this paper, we use spectral clustering methods [14] to perform this task.

4.2.1 Spectral clustering

Each 2D mode is a 3D point $\mathbf{m}_n = [p_n, t_n]$ in the video volume, where p_n and t_n are the spatial location and the temporal location (frame number) respectively. Given a set of points $\mathbf{M} = \{\mathbf{m}_n\}_{n=1}^N$ in $\Re^3$, spectral clustering builds an affinity matrix A and then clusters data points based on the eigenvector analysis of the Laplacian matrix of A. Variants of spectral clustering algorithms analyze the eigenvectors in sightly different ways. In this paper, we use Ng's algorithm [14] which uses K eigenvectors simultaneously for K-class clustering. In our trajectory segment analysis, the basic affinity matrix $A \in \Re^{N \times N}$ is defined as follows:

$$A_{ij} = \exp(-||p_i - p_j||^2/2\sigma_p^2 - ||t_i - t_j||^2/2\sigma_t^2), \quad (5)$$

where the scaling parameters σ_p and σ_t control how rapidly the affinity A_{ij} falls off with the distance between two points $\mathbf{m}_i$ and $\mathbf{m}_j$ in space and time, respectively. To encourage more compact trajectory segments, we use an ap-

pearance dependent definition in this paper:

$$A'_{ij} = \alpha A_{ij} + (1-\alpha)\exp(-B^2[\mathbf{h}(\mathbf{m}_i), \mathbf{h}(\mathbf{m}_j)]/2\sigma_h^2), \quad (6)$$

where the last term measures the similarity between the appearances (color histogram in this paper) of two modes. α is a weighting factor (typically 0.5). The process to partition the points into K clusters is as follows:

1. Build the affinity matrix A according to Equation (6).
2. Construct the matrix $L = D^{-1/2}AD^{-1/2}$ where D is a diagonal matrix ($D_{ii} = \sum_{j=1}^{N} A_{ij}$).
3. Compute the matrix $E = [e_1, ..., e_K] \in \Re^{N\times K}$, where e_k is the normalized K largest eigenvectors of L.
4. Treat each row of E as a point in $\Re^K$, and cluster them into K clusters by K-means algorithm. Assign the original point to cluster k if row i of the E was assigned to cluster k.

After spectral clustering, we treat all 3D points in cluster k as a trajectory segment Tr_k. Namely, we get a number of K trajectory segments $\mathbf{Tr} = \{Tr_1, ..., Tr_K\}$. Figure 4(e) shows the extracted trajectory segments on the "circles" sequence. Spectral clustering successfully produces a number of "meaningful" trajectory segments.

4.2.2 Why use spectral clustering?

We get less "meaningful" results from a standard k-means clustering. The reason is that the "true" trajectory is usually highly curved and some partition of it may not be a convex region, but every cluster of k-means has to be a convex region. Figure 4(a)(b) shows two k-means results using different scaling factors of the time variable t. In fact, we found that k-means always gives unsatisfactory results no matter what the scaling factor is for this example.

In contrast, in spectral clustering, 3D data points are embedded on the surface of a unit sphere in another K dimensional space spaned by the K largest eigenvectors of L. In this space, curved trajectories or manifolds in the original 3D space can be well separated. Clustering in the embedded space using spectral analysis is the key to our trajectory segment analysis. We refer the reader to [14, 3] for more details and comparisons.

5 Occlusion Reasoning

If there is no occlusion of the target object, trajectory segments extraction is already a very good "proposal" for state space sampling in trajectory optimization. However, due to partial or complete occlusion occurring in the input video, the occlusion (trajectory) segment (the states during occlusion stage) does not exist in already extracted segments. The occlusion segment should be inferred and sampled between object trajectory segments. This section presents a simple but effective occlusion reasoning algorithm at the trajectory segment level.

After analyzing the trajectory segments on a number of video sequences, we have several observations:

A. The trajectory segment including object templates in the keyframes must be in the "true" object trajectory.

B. The trajectory segment parallel to the segment which contains object templates should be excluded.

C. No occlusion segment exists between two overlapping trajectory segments along the time axis.

D. There are certain speed and time limits on an occlusion segment.

In observation B, two segments are parallel if the overlapping time and the shortest distance between them are not more than certain empirical thresholds. For example, in Figure 4(e) the vertical segment (red) in the center will be excluded because it is parallel to two segments (cyan and dark-green) containing object templates.

5.1 Occlusion reasoning algorithm

Based on the above observations, we propose an bi-directional, tree-growing algorithm for occlusion reasoning as follows:

1. Build two trees T_A and T_B. Each tree has an empty root node. Then, add one trajectory segment containing an object template in the keyframe to each tree as an *active* node. The remaining segments are denoted as a candidate set.
2. Exclude the trajectory segment from the candidate set using the current two trees according to observation B.
3. For each *active* leaf-node (node without child) in T_A, add the Q-best occlusion segments from the candidate set or the *active* leaf-nodes in T_B as its child nodes, according to observations C and D. The newly added child node is set to *active* if it comes from the candidate set. Otherwise, it is set to *inactive* in both trees.
4. The tree T_B grows one step in a similar way.
5. If there is no *active* leaf-node in both trees, stop; otherwise, go step 2.

Occlusion trajectory generation For two disjoint trajectories Tr_1 and Tr_2 in time, we want to fill in the missing occlusion segment O in between, as shown in Figure 5. Given all points $\{\mathbf{m}_j = [p_j, t_j]\}_{j=1}^{N'}$ in Tr_1 and Tr_2, we fit a B-spline $\mathbf{r}(s) = \sum_{n=0}^{N_B} B_n(s)\mathbf{q}_n$ using weighted least squares:

$$\min_{\{\mathbf{q}_n\}} \sum_{j=1}^{N'} w(\mathbf{m}_j)||\mathbf{r}(s'_j) - \mathbf{m}_j||^2, \quad (7)$$

where $s'_j = (t_j - t_1)/N'$ is a temporal parametrization of the B-spline in frame t_j. Although it is an approximation of

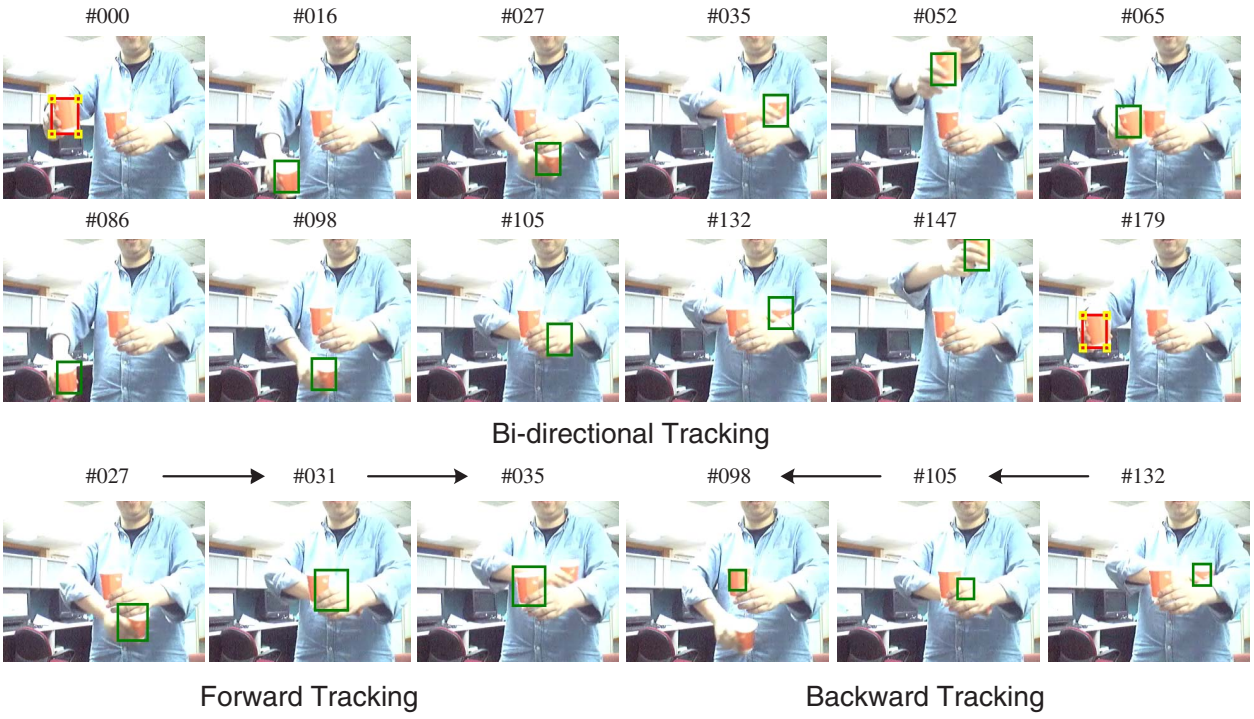


Figure 3: The **"circles"** example. Bi-direction tracker successfully tracked the whole sequence. Forward tracker and backward tracker failed at frame 031 and 105 respectively. The keyframes 000 and 179 contain two object templates.

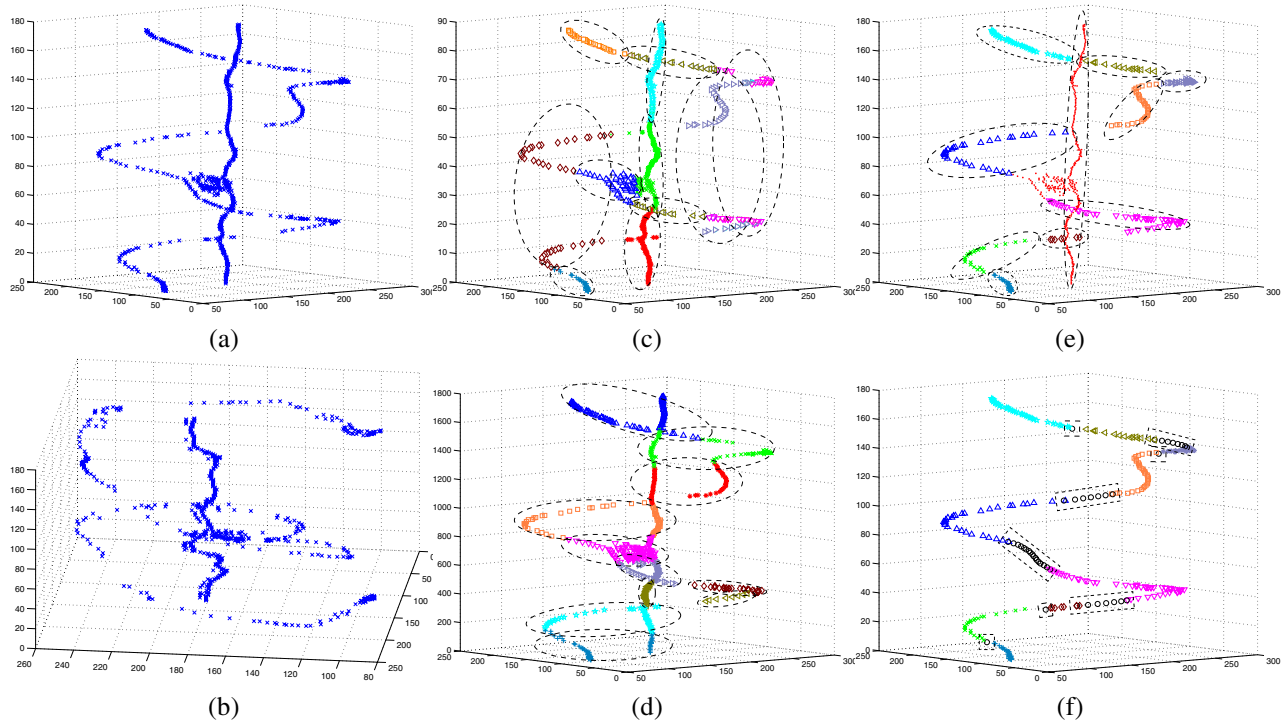


Figure 4: Trajectory segments analysis on **"circles"** sequence. (a)(b) two views of all 2D mode points in 3D. The vertical axis is the frame number in the sequence. (c)(d) two k-means results with different time scaling factors. K-means does not provide very meaningful "segments" in terms of trajectory. (e) meaningful "segments" from spectral clustering. (f) result after occlusion reasoning. Black circles in dashed rectangles are filled-in occlusion segments. *Please view the electronic version for a better illustration in color.*

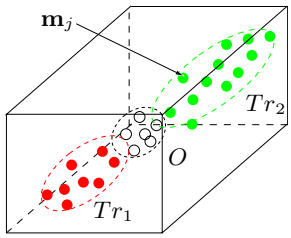


Figure 5: Occlusion trajectory generation. Point $\mathbf{m}_j$ is an 2D local mode and $\{Tr_1, Tr_2\}$ are 3D trajectory segments in the 3D video volume. O is an inferred occlusion (trajectory) segment between Tr_1 and Tr_2.

the parametrization of a B-spine in a strict sense, we found that it is good enough in our application. The weighting function $w(\mathbf{m}_j)$ is defined as:

$$w(\mathbf{m}_j) = \begin{cases} \exp(-\sigma_w^{-1}||t_j - t_A^e||) & j \in Tr_1 \\ \exp(-\sigma_w^{-1}||t_j - t_B^s||) & j \in Tr_2, \end{cases} \quad (8)$$

where t_A^e and t_B^s are the last frame number in Tr_1 and the first frame number in Tr_2. The scaling parameter σ_w is set to 20 in our experiments. Using a weighting function, we can give a larger weight to a point nearby t_A^{end} and t_B^{start}. Finally, we treat the fitted B-spline curve between Tr_1 and Tr_2 as the occlusion segment O.

Q-best hypothesis For an active leaf node Tr^a, we search forward for all possible trajectory segments $\mathbf{Tr}^*$ that satisfy the observation C. Then, we generate the occlusion segments between Tr^a and every trajectory segments in $\mathbf{Tr}^*$. The Q-best occlusion segments are selected based on $(L_O + \gamma S_O)$, where $\gamma = 10$ is a weighting factor. L_O and S_O are the length (pixel) and maximum speed (pixel $*$ frame^{-1}) of the hypothesized B-spline between two connected trajectory segments. As mentioned in the last section, a large K is selected in our system. A long trajectory segment may be divided into two very close segments. Therefore, we add a *dominant grouping* process in the Q-best hypothesis: we just keep one trajectory segment if $(L_O + \gamma S_O)$ of this segment is significantly smaller than others.

Figure 4(f) shows the final trajectory segments and occlusion segments inferred by occlusion reasoning. The vertical segment in the center (red) is excluded in the first iteration using observation B. Curved occlusion segments are successfully generated by our weighted least squares fitting.

6 Trajectory Optimization

After getting a set of trajectory segments and occlusion segments, a single optimal trajectory between two keyframes is computed by trajectory optimization. In order to obtain a more accurate tracking result, we perform trajectory optimization of Equation (2) in a coarse-to-fine manner. Two-levels is sufficient for all examples in the experiments. In the coarse level (spatially down-sampled only), we uniformly sample M (500-1000) states around (in a small radius, e.g. 5 pixels) the segments using three discrete scaling factors s in each frame. A optimal trajectory is computed in this level using a discrete HMM model. In the fine level, we sample M states around the optimal solution obtained from the coarse level using five discrete scaling factors in each frame. For a 10 second video, the trajectory optimization took about 8 seconds.

7 Experimental Results

In this paper, we compare our approach with standard particle filtering (PF) with a first-order dynamics with 500 particles. The observation model in PF tracker is exactly the same as the likelihood in our bi-directional tracker.

Parameter setting In 2D mode extraction, G is a Gaussian kernel whose standard deviation is about $1/6$ the size of the target object in the keyframe. In the 3D trajectory extraction, scaling parameters σ_p and σ_t are set to 10 and 20. We set the clustering number to $K = 7$ or $K = 10$ for all examples shown in this paper. Adaptive selection of K may be addressed in future work. In trajectory optimization, the variance parameters σ_h and σ_p are 10 and 1, respectively.

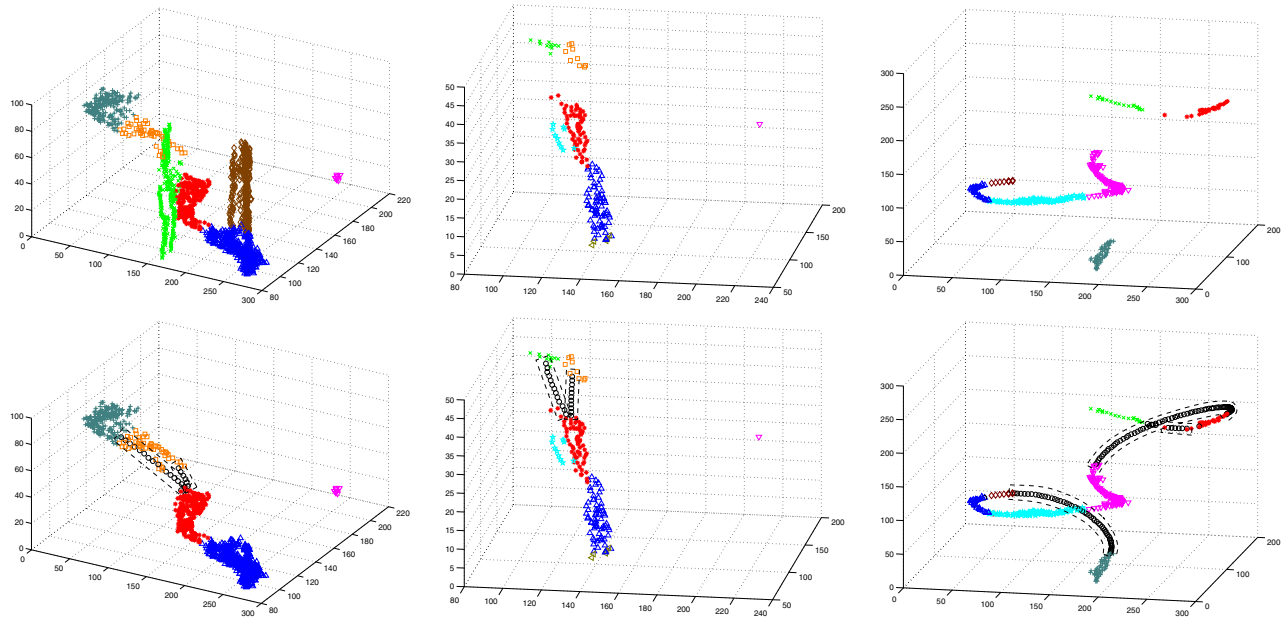
In the first-order dynamic $x_i = x_{i-1} + cv(i)$ of PF, $c = diag(c_x, c_y, c_s)$ and $v(t) \sim N(0, 1)$ is a normal distribution. In our experiment, we set the parameters as: c_x is 8 pixels/frame, c_y is 8 pixels/frame, and c_s is 0.1 /frame. We have also tested a second-order dynamic and turned the parameters video by video. But we found that the improvements are marginal on our test sequences.

"Cup" sequence includes two almost identical objects. The target object passes close to the other from frame 33 to 66. This ambiguous event corresponds the red asterisk trajectory segment in Figure 6(a). Neither forward PF nor backward PF can correctly track the target after this event. To solve this ambiguity, our occlusion reasoning generates two hypotheses and trajectory optimization selects a smoother one, as shown in Figure 6(a).

"Leg" sequence shows a complete occlusion from frame 35 to 45. This event can be easily identified in Figure 6(b). Occlusion reasoning hypothesizes two occlusion segments. The correct path is picked by trajectory optimization. Again, forward PF and backward PF is incorrect from frame 36 and 40 respectively because the background's color is more similar to the target than the leg's color.

"Toy" sequence shows two long-time occlusions from 23 to 70, and from 155 to 209. Two curved occlusion segments are inferred by our B-spline based estimation, as shown in Figure 6(c). The tracking results are shown in Figure 7.

"Magic" sequence shows a more ambiguous event. Two indistinguishable Pepsi cans enter and then leave a blind area. For the target object on the left side in frame 0, it can go back the left side or go to the right side in frame 127. To



(a) Cup (b) Leg (c) Toy

Figure 6: Trajectory segments analysis results (Top) and occlusion reasoning results (Bottom). The black circles in dash rectangles are inferred occlusion segments. (a) two segments corresponding to the cup in the center and a green region on the background are excluded. (b) two possible occlusion segments are hypothesized. (c) two highly curved occlusion segments are estimated.

solve this ambiguity, our bi-directional tracker can give two reasonable guesses by specifying two kinds of keyframes, as shown in Figure 7.

8 Conclusion

In this paper, we have presented a bi-directional tracking approach based on trajectory segment analysis. Curved target object trajectories are successfully extracted by trajectory segment analysis and connected by the occlusion reasoning algorithm. With a trajectory segment representation, more challenging visual tracking tasks can be well handled.

There are many opportunities to improve and generalize our approach, such as automatic selection of clustering number, handling large appearance changes between two keyframes, integrating more visual cues, developing other state representations, and bi-directional tracking of multiple objects.

References

[1] A. Agarwala, A. Hertzmann, D. Salesin, and S. Seitz. Keyframe-based tracking for rotoscoping and animation. In *Proceedings of SIGGRAPH 2004*, 2004.

[2] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. *CVPR*, 1998.

[3] M. Brand and K. Huang. A unifying theorem for spectral embedding and clustering. *Proceedings of Inte. Conf. on AI and Statistics*, 2003.

[4] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *CVPR*, 2000.

[5] I.J. Cox and S.L Hingorani. An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Tran. on PAMI.*, 8(2):138–150, 1996.

[6] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings on Radar and Signal Processing.*, 140:107–113, 1993.

[7] G.D. Hager and P.N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Tran. on PAMI*, 20(10):1025–1039, 1998.

[8] M. Han, W. Xu, H. Tao, and Y. H. Gong. An algorithm for multiple object trajectory tracking. *CVPR*, 2004.

[9] M. Irani. Multi-frame optical flow estimation using subspace constraints. *ICCV*, 1999.

[10] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *ECCV*, 1996.

[11] J. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *J. Amer. Statist. Assoc.*, 93:1032–1044, 1998.

[12] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *Proceedings of the Int. Joint Conf. on AI.*, pages 593–600, 1981.

[13] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. *ICCV*, 1999.

[14] A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *NIPS*, 2002.

[15] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *ECCV*, 2002.

[16] D.B. Reid. An algorithm for tracking multiple targets. *IEEE Tran. on Automatic Control*, 24(6):843–854, 1979.

[17] L. Torresani and C. Bregler. Space-time tracking. *ECCV*, 2002.

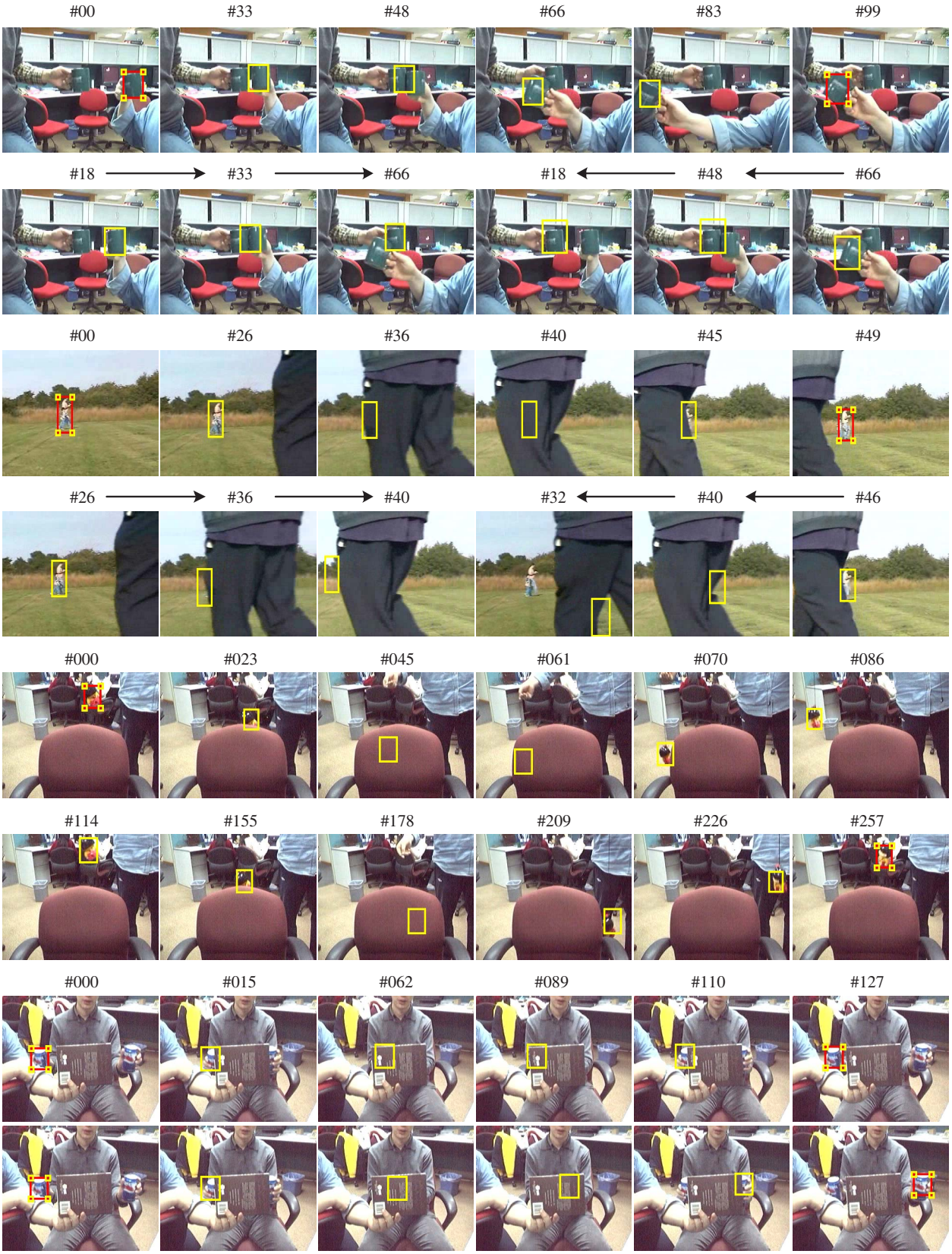


Figure 7: **"Cup"**, **"Leg"**, **"Toy"**, and **"Magic"** examples (from top to bottom). In "cup" and "leg" examples, we compare bi-direction tracking result with forward PF and backward PF. The image containing a red rectangle is the keyframe.