A NEW METHOD FOR TRACKING PERFORMANCE EVALUATION BASED ON A REFLECTIVE MODEL AND PERTURBATION ANALYSIS

Pan Pan[†‡]*, Fatih Porikli*[‡]*, Dan Schonfeld*[†]

University of Illinois at Chicago[†], Mitsubishi Electric Research Laboratories[‡]

ABSTRACT

In this paper, we present a novel methodology for tracking performance evaluation. Considering the continuity of the image sequences in a video, we define a new measurement called *tracking difficulty* which incorporates the local sequence information among a small image sequence centered at each frame. We subsequently use a reflective model to formulate tracking difficulty. Tracking difficulty curves can not only illustrate at which parts of the video one tracking algorithm performs well or poor, but also provide a way to compare the performance of different tracking algorithms. We further add perturbation analysis to the reflective model to examine how sensitive the tracking algorithm is to noise. Results on data sets are presented to show the effectiveness of our evaluation method.

Index Terms— Tracking performance evaluation, reflective model, perturbation analysis.

1. INTRODUCTION

Visual object tracking is one of the most important tasks in computer vision. The applications include video surveillance, traffic management, vehicle control systems, robotics, augmented reality, etc.

By far, a large number of algorithms are presented for visual object tracking in real applications, such as particle filters [1], mean-shift tracker [2], covariance tracker [3], etc. However, it would be premature to claim that a single technique can handle successfully any real world conditions. There are unfortunately many natural reasons to fail a tracker including irregular and fast object motion, partial and full occlusions, object appearance changes, drastic pose and size transformations.

Therefore, it is a natural question to ask how to evaluate the performance of tracking algorithms. Absolute error [1] and root mean squared error between ground truth and the tracking results at each frame are the most commonly used evaluation criteria. In [4] pseudo synthetic video is used to evaluate tracking performance. Algorithms are proposed in [5] to match ground truth tracks and system generated tracks and compute performance metrics based on these correspondences. In [6], several performance evaluation metrics were presented for detection and tracking. Tracking along forward and time reversed Markov chain is used in [7] to evaluate the tracking performance.

Most of the previous work uses the frame-based information as evaluation criteria, i.e. using the information at a certain frame to evaluate tracking performance at that frame. The fact is that the videos are continuous, and the performance of tracking algorithms at each frame also depends on the characteristics of the nearby frames. Therefore, it will be more interesting to use some sequence-based measurement which incorporates all of the information among a small segment of image sequences centered at each frame. Moreover, the current methods could not explore the rest of the video after a complete loss of tracking occurs and cannot tell the overall performance on the video. Furthermore, we would also wish to examine the robustness of tracking algorithms against noise, i.e. whether the tracking could still be recovered after the tracker made an error at one frame.

In this paper, we propose tracking difficulty as a novel measurement to assess the performance of tracking algorithms. Different from other measurement which utilizes the frame-based information, tracking difficulty incorporates the local sequence information among a small image sequence centered at each frame. We set up a novel reflective model to formulate tracking difficulty. The implementation of this model involves multi-directional tracking along forward and backward paths. Tracking difficulty curves can not only illustrate at which parts of the video one tracking algorithm performs well or poor, but also provide a way to compare the performance of different algorithms. We further add perturbation analysis to the reflective model to examine the sensitivity of tracking algorithms against noise.

2. TRACKING PERFORMANCE EVALUATION

We define interested objects as the objects we want to track in the video. It usually includes the moving objects and excludes the objects which belong to the background. Since the video is continuous, i.e. all the interested objects in the video have their past and future. Therefore, under the assumption that all the interested objects in the video are casual, tracking difficulty at frame k (Q_k) describes the average effort which has to make in order to get the accurate trajectories for all the interested objects among a small sequence centered at frame k.

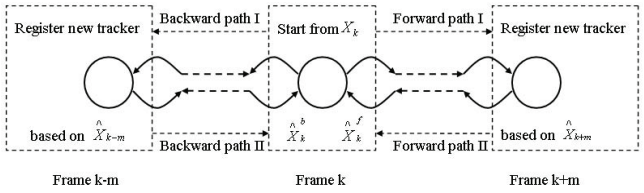


Fig. 1. Reflective model for single object tracking.

2.1. Reflective Model

For simplicity reason, we start from single object tracking. In order to access the tracking difficulty from frame $k-m$ to $k+m$ centered at frame k, where m is an integer usually very small, the common method is to run tracking from frame k to frames $k \pm m$ respectively and compare the tracker information, e.g. color histogram. However, it is more reasonable to reflect the output at $k-m$ and $k+m$ to frame k and make comparisons at frame k. The reasons are as follows. First, the information in the tracker may change with time, e.g. appearance change, which causes the comparison unfair. Second, because we are interested in the tracking difficulty at frame k, the comparison is better performed at frame k.

Therefore, a detailed description of the reflective model is given in Fig. 1. $\mathbf{x}_k$ is the state at frame k, known as input. The purpose of the reflector is to send back the information it receives. The implementation of the reflector could be done by registering the new tracker based on the tracking results at frames $k+m$ and $k-m$. We call the tracking path from frame $k, k+1, \ldots, k+m$ and then back $k+m, k+m-1, \ldots, k$ as the *forward path*, where the reflection occurs at frame $k+m$. Therefore, $\hat{\mathbf{x}}_k^f$ is the tracking result of the forward path. Similarly, the *backward path* is the tracking path from frame $k, k-1, \ldots, k-m$ and then back $k-m, k-m+1, \ldots, k$, where the reflection occurs at frame $k-m$. $\hat{\mathbf{x}}_k^b$ is the output of the backward path.

The differences between $\mathbf{x}_k$ and the returned $\hat{\mathbf{x}}_k^f$ and $\hat{\mathbf{x}}_k^b$ describe the effort which has to make in order to get the accurate trajectories for the interested object among $2m$ images centered at frame k, i.e. tracking difficulty at frame k. Therefore, we define *tracking difficulty at frame* k (Q_k) as the linear combination of the distance of the input and two reflected outputs in both forward and backward paths, i.e.

$$Q_k = \alpha_f D(\mathbf{x}_k, \hat{\mathbf{x}}_k^f) + (1-\alpha_f) D(\mathbf{x}_k, \hat{\mathbf{x}}_k^b), \qquad (1)$$

where α_f is the coefficient for the forward path; $D(\mathbf{x}_k, \hat{\mathbf{x}}_k^f)$ and $D(\mathbf{x}_k, \hat{\mathbf{x}}_k^b)$ measure the error between $\mathbf{x}_k$ and $\hat{\mathbf{x}}_k^f$, $\hat{\mathbf{x}}_k^b$ respectively, which will be defined later. Usually, because of the identical impact of error measurement for forward and backward paths, we set $\alpha_f = \frac{1}{2}$. From (1), we can see that if a tracking algorithm performs pretty good on a video segment centered at frame k, then the reflected tracking estimates should be pretty close to the input. Therefore, Q_k is approximately to zero. Otherwise, if the tracking performs poor on this small sequence, Q_k is large.

When there are multiple objects in the video, the extension of (1) depends on the description of the objects. If a jointly description is used, where a large joint vector $\mathbf{x}_k$ is used to describe all the objects within frame k, (1) does not need to be changed. However, the computational complexity of tracking algorithms will grow exponential with the number of objects. Nowadays, the distributed framework [8] to describe multiple objects gains popularity in tracking algorithms, because of the linear increase of the complexity along with the number of objects. Therefore, in this paper, we confine our expression under the distributed framework, where each object is tracked by a distributed tracker. We denote the state of the i^{th} object in the k^{th} frame as $\mathbf{x}_k^i$, where $i = 1, 2, \ldots, m$. Then Q_k could be expressed as

$$Q_k = \frac{1}{m}(\alpha_f \sum_{i=1}^{m} D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,f}) + (1-\alpha_f) \sum_{i=1}^{m} D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,b})), \qquad (2)$$

where we use the average distances over all objects within in one frame as the tracking difficulty for that frame.

The philosophy of our tracking difficulty can be considered similar with that in communication systems, where channel refers to the medium used to convey information from a sender to a receiver and in order to assess the properties of a channel, the information at the receiver is compared with the information at the sender by asking the receiver to send the received signal back, as our trackings along forward and backward paths.

2.2. Error Measure

The distance functions $D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,f})$ and $D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,b})$ capture the distance of the state $\mathbf{x}_k^i$ and the returned estimates $\hat{\mathbf{x}}_k^{i,f}$, $\hat{\mathbf{x}}_k^{i,b}$ respectively. Besides the spatial distance, we could also incorporate some feature distance. Therefore, the distance function is defined as

$$D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q}) = \omega_s Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q}) + (1-\omega_s) Df_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q}),$$

where $q = f$ or b; $Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$ and $Df_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$ denote the spatial and feature distances respectively; ω_s is the coefficient of the spatial distance. We find out by experiments that the spatial distance is more reliable than the feature distance, because the background may have the similar features to the objects. Therefore, ω_s is usually larger than $1/2$.

A good example of feature distance $Df_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$ is the Bhattacharyya distance of color histogram. For the spatial distance $Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$, we use a sigmoid function which consists of two segments depending on whether $\mathbf{x}_k^i$ and $\hat{\mathbf{x}}_k^{i,q}$ overlap or not. The non-overlapping ratio r is defined as $r = 1 - S_{\mathbf{x}_k^i \cap \hat{\mathbf{x}}_k^{i,q}} / S_{\mathbf{x}_k^i \cup \hat{\mathbf{x}}_k^{i,q}}$, where the operator $S_{\mathcal{R}}$ is defined as the number of pixels belong to region $\mathcal{R}$. As shown in Fig. 2,

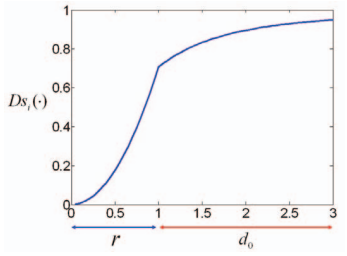


Fig. 2. Spatial distance $Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$.

$Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$ is defined as

$$Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q}) = \begin{cases} f(r) & 0 \le r < 1, \\ g(d_0) & r = 1, \end{cases}$$

where $f(r) = 0.7071r^2$, $g(d_0) = d_0/\sqrt{1+d_0^2}$, $d_0 = d/d_{min}$; d and d_{min} are the actual distance and minimum distance of the centers of $\mathbf{x}_k^i$ and $\hat{\mathbf{x}}_k^{i,q}$ respectively. When $\mathbf{x}_k^i$ and $\hat{\mathbf{x}}_k^{i,q}$ are overlapping, i.e. $0 \le r < 1$, $Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$ is a function of r. While when they are non-overlapping, $Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q})$ is a function of the normalized distance of the centers of $\mathbf{x}_k^i$ and $\hat{\mathbf{x}}_k^{i,q}$. It is easy to see that $Ds_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,q}) \in [0, 1]$ and is continuous and monotonic increasing.

Given the ground truth data $\mathbf{x}_k^i$, tracking difficulty curves can not only illustrate at which parts of the video one tracking algorithm performs well or poor, but also provide a way to compare the performance of different tracking algorithms. The main steps to obtain Q_k are summarized as follows:

1. For each frame, obtain $\mathbf{x}_k^i$ from the ground truth.
2. Track along the forward path till frame $k+m$, re-register trackers using tracking results at frame $k+m$, and track back to frame k to obtain $D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,f})$.
3. Track along backward path to obtain $D_i(\mathbf{x}_k^i, \hat{\mathbf{x}}_k^{i,b})$ similarly.
4. Get Q_k as shown in (2).

2.3. Perturbation Analysis

In order to examine whether the tracking could still be recovered after the tracker made an error at one frame, we incorporate perturbation analysis [9] to the reflective model. As shown in Fig. 3, after we initialize a tracker based on the ground truth $\mathbf{x}_k^i$, we add a small shift to the input, i.e. $\tilde{\mathbf{x}}_k^i = \mathbf{x}_k^i + \Delta\mathbf{x}$. The error ratio τ defined as

$$\tau = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\frac{\|\hat{\mathbf{x}}_k^{i,f} - \mathbf{x}_k^i\|}{\|\Delta\mathbf{x}\|} + \frac{1}{2}\frac{\|\hat{\mathbf{x}}_k^{i,b} - \mathbf{x}_k^i\|}{\|\Delta\mathbf{x}\|}\right)$$

is used to show how robust the tracking algorithm is against noise.

3. EXPERIMENTS

We use several test sequences with ground truth to demonstrate the effectiveness of our evaluation method. Fig. 4 gives a few sample images from the sequences used in the experimental tracking evaluation.

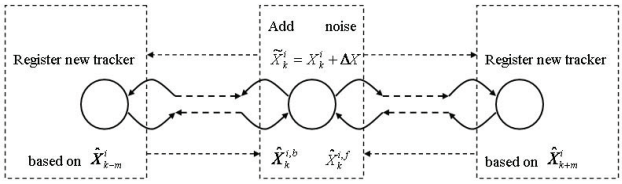


Fig. 3. Perturbation analysis of the reflective model.

Fig. 4. Sample images from sequences used in the experimental tracking evaluation.

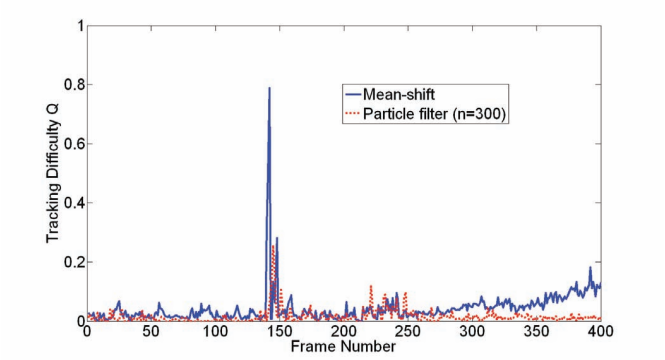
In the `Campus` sequence, we run the tracking using mean-shift tracker and particle filters. The boy in the video speeds up at frame 144, and then slows down. The tracking difficulty curves are shown in Fig. 5 (a). The root mean squared error between the tracking results and ground truth is shown in Fig. 5 (b). From Fig. 5 (a), we can see that in general particle filters with 300 particles performs better than mean-shift tracker, the same as shown in Fig. 5 (b). From Fig. 5 (a), we can see that around frame 144 mean-shift tracker performs much worse than other parts of the video, which is verified by the loss of tracking shown in Fig. 5 (b). Moreover, another advantage of our method is that Fig. 5 (a) can still evaluate the performance for the rest of the video after complete loss of tracking, as shown by mean-shift tracker in Fig. 5 (a).

We use a set of particle filters with different number of particles while keeping other parameters unchanged to run the tracking task on some CAVIAR sequences. The energy of the difficulty curves are calculated in Table 1. We can see the more particles are use, the better performance of the particle filters. And the performance improvement is not linear with the increase of the number of particles.

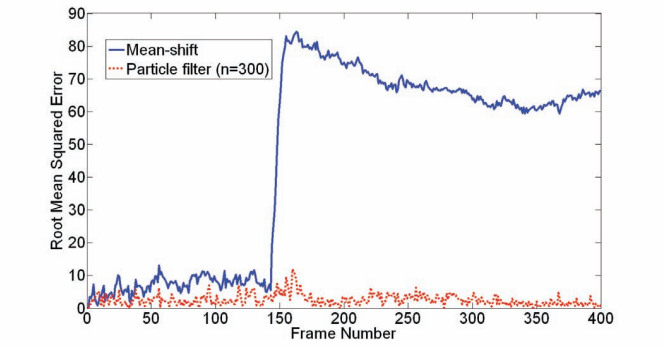
We further use perturbation analysis on the `Front` sequence. For fair comparisons, we keep the shift after initialization constant for all frames for all comparative tracking methods. The error ratios of tracking algorithms are shown in Table 2. We can see that the order of robustness against noise is PF (n=50) $>$ PF (n=30) $>$ Mean-shift. Also, the smaller

Table 1. Energy of tracking difficulty curves on CAVIAR data sets using particle filters. We can see tracking with more particles outperforms tracking with fewer particles.

Number of particles	10	30	50
Sequence I	0.4909	0.3602	0.3088
Sequence II	0.8853	0.7049	0.5968
Sequence III	3.0264	2.2521	1.6894



(a)



(b)

Fig. 5. Results on the `Campus` sequence. (a) Tracking difficulty curves obtained by mean-shift tracker and particle filter with 300 particles. (b) The root mean squared error between the tracking results and ground truth.

Table 2. Error ratios of perturbation analysis on the `Front` sequence.

Size of shift	Mean-shift	PF n=30	PF n=50
(0.5, 0.5)	10.0625	5.5796	5.3595
(1, 1)	5.0515	2.7921	2.6821
(1.5, 1.5)	3.3806	1.8630	1.7896

noisy shift will cause larger error ratio, which implies that tracking algorithms are more sensitive to small noises.

The choice of m decides the resolution of the difficulty curves. The larger m is, the more complete information it represents. However, large m will cause delay and a waste of computational resources. In the experiments above, we choose $m = 6$.

4. CONCLUSION

In this paper, we propose a novel tracking performance evaluation method based on a reflective model and perturbation analysis. Tracking difficulty is proposed as a new measurement which incorporates the local sequence information centered at each frame and is formulated using a reflective model. Perturbation analysis is applied to examine how the tracking algorithm performs against noise. Experimental results show the effectiveness of our evaluation method. Our method can tell (1) at which parts of the video the tracking algorithm performs well or poor. (2) how one tracking algorithm performs compared with other algorithms. (3) how sensitive one algorithm is against noise.

5. REFERENCES

[1] P. Pan and D. Schonfeld, "Dynamic proposal variance and optimal particle allocation in particle filtering for video tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 9, 2008.

[2] D. Comaniciu, V. Ramesh, and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, 2003.

[3] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[4] J. Black, T. Ellis, and P. Rosin, "A novel method for video tracking performance evaluation," in *IEEE VS-PETS Workshop*, 2003.

[5] L. M. Brown, A. W. Senior, Y. Tian, J. Connell, and A. Hampapur, "Performance evaluation of surveillance systems under varying conditions," in *IEEE PETS Workshop*, 2005.

[6] F. Bashir and F. Porikli, "Performance evaluation of object detection and tracking systems," in *IEEE PETS Workshop*, 2006.

[7] H. Wu, A. C. Sankaranarayanan, and R. Chellappa, "In situ evaluation of tracking algorithms using time reversed chains," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[8] W. Qu, D. Schonfeld, and M. Mohamed, "Real-time distributed multiobject tracking using multiple interactive trackers and a magnetic-inertia potential model," *IEEE Transactions on Multimedia*, vol. 9, no. 3, 2007.

[9] G. W. Stewart, *Matrix Algorithms, Volume I: Basic Decompositions*, SIAM, 1998.