Mesh-Shrink: Real-Time Fast Moving Object Tracking with Sporadic Occlusion

Wei Quan, Jim X. Chen, and Nanyang Yu

Abstract—**This paper proposes a new method for real-time object tracking in scenarios where the target moves fast but its appearance does not change quickly. By creating a mesh on the current frame, we set nodes of the mesh as candidate positions of the target. Several adjacent nodes which are more similar to the target constitute a new smaller search region, on which a new finer mesh is created. The approach iterates until certain conditions are satisfied. Finally, one of the nodes is identified as the target location. Unlike existing tracking methods, this approach achieves tracking of fast moving object in real time and is capable of recovering tracking when the target is missing due to full occlusion, or moving out of the image area and reappearing in the near future frames. The method does not require complicated computation and thus can be applied to the environment which permits only limited computing resources. The capabilities of the tracking based on our method are demonstrated by several image sequences.**

Index Terms—**Mesh based search, fast moving object tracking, recovering tracking, target representation and localization.**

I. Introduction

Object tracking is a key problem of automated video analysis that has been applied to many significant applications in computer vision, such as intelligent surveillance, human-computer interfaces, augmented reality, driverless vehicles, object-based video compression and 3D modeling. The task of a tracker is to generate the trajectory of an object over time by marking its position in every frame of the video. An ideal tracker is capable of tracking a target robustly for a long duration, handling occlusion, and resolving entry and exit of the target. Moreover, it is not too complex to apply to some scenarios where the computing power of a device is much lower than that of a normal PC. That is, the tracker is also practically applicable when the tracking environment permits only limited resources in remote locations or confined spaces.

Effective methods have appeared for object tracking [1], [2]. Two major categories of tracking methods are probabilistic tracking and deterministic tracking. The former derives the target position by spatio-temporal estimation. For example, Unscented KF (UKF) is a method that utilizes a set of definite samplings to approximate posterior probability density function [3], while particle filter (PF) uses random particles [4], [5]. For these methods, a large number of samplings increasing with the complexity of the applications are required, which incurs significant computational cost and is the major problem in real-time applications.

The later treats the tracking task as an optimization procedure, which establishes a cost function of the observation model and determines the target position while the cost function happens to have a maximum or minimum value. Mean shift (MS) is a typical method of deterministic tracking [6, 7], which acquires efficient tracking results at an interactive video rate. The central computational module of MS is based on the MS iterations and finds the most probable target position in the current frame. An improved method for MS is proposed to deal with the difficulties that MS faces by using a new, simple-to-compute and more discriminative similarity measure in spatial-feature spaces [8]. However, as MS requires an original or estimated location of the target for its iterations in advance to achieve tracking in a confined region, it cannot recover tracking when it loses the target. A robust appearance filter is proposed to update the target template and achieve fast occluded object tracking [9]. But it cannot handle full but partial occlusions. In order to track an object with a complex shape, such as a human or an animal, silhouette based approaches have been proposed [10], [11]. Depending on the object model generated using the previous frames in the form of a color histogram, object edges, or contour, direct shape matching is used in [10] and contour evolving is used in [11]. Nevertheless, the problems related to silhouette trackers, including object topology changes, occlusions, and fast moving object tracking, need to be addressed further.

On the other hand, many researchers use self-learning [12]-[17] to perform adaptive object tracking, which is related to classification problem and able to adapt the tracker to new appearance and background by updating the model with positive and negative examples in the vicinity of the current location of the target. In order to address the problem of drifting caused by the errors introduced accumulatively while updating the tracker, many research efforts have been taken, e.g. Semi-supervised learning [18]-[21], where the processing of unknown data is guided by some supervisory information [22] such as relationships or constraints [23], and MIL (Multiple Instance Learning) [24], where the training samples are delivered by spatially related units rather than independent ones. To continue tracking correctly when the

Manuscript received October 15, 2012; revised November 18, 2012. This work was supported in part by the National Natural Science Foundation of China (Grant No. 40672203) and the Doctoral Innovation Foundation of Southwest Jiaotong University.

Wei Quan is with the School of Information Science and Technology, Southwest Jiaotong University, Chengdu, Sichuan 610031, PR China (e-mail: xweiquan@gmail.com).

Jim X. Chen is with the Department of Computer Science, George Mason University, Fairfax, Virginia 22030, U.S.A (e-mail: jchen@gmu.edu).

Nanyang Yu is with the School of Mechanical Engineering, Southwest Jiaotong University, Chengdu, Sichuan 610031, PR China (e-mail: yunanyang@hotmail.com).

tracker makes a mistake, Yu et al. [25] proposed co-training, a generative and discriminative classifiers during tracking. Due to the re-detection capability of their tracking algorithm, it performed well in comparison to self-learned trackers. Another approach is combing adaptive tracking with object detection [26], where the tracking algorithm is based on P-N learning, also known as growing and pruning events. However, these methods are computationally complex and thus hard to apply to the context in which the capability of processors is relatively low, such as an embedded system.

In this paper, we present a new method for real-time object tracking in scenarios where the target moves fast but its appearance does not change quickly, which is called *mesh-shrink* for convenience of description. Mesh-shrink searches for the target from the entire image using an initial mesh covering it, derives a smaller region by those mesh nodes which are more similar to the target, and then creates a new finer mesh on the smaller region. The algorithm iterates the search procedure until certain conditions are satisfied and finally determines the location of the target. It is, therefore, not laborious to implement the algorithm due to the mesh based search without complicated computation. To measure the similarity between the object and the target candidates, many visual features including color, texture, and silhouette can be employed.

The remainder of this paper is organized as follows. Section 2 presents the principle of tracking based on mesh-shrink. Section 3 presents the implementation details, including object representation and similarity measurement, target localization, and adaptation. Experimental results are presented in Section 4 and finally some observations are discussed in Section 5.

II. The Mesh-Shrink Principle

The main idea of mesh-shrink is using the mesh and its nodes to narrow the search region and finally find the location of the target. A certain similarity measurement is required to estimate the distance between the target and the target candidate.

A. Analysis of the Target Location

As shown in Fig. 1, rectangle T is overlapped with four rectangles denoted by A, B, C and D, which have the same size as T and the overlapped areas are denoted by S_A, S_B, S_C, and S_D, respectively. The overlapped areas are used to analyze the distances of the corresponding rectangles to the target. Apparently, $S_B > S_A > S_C > S_D$, and therefore the position of B is the closest to that of T.

For a moving object in an image sequence, T is the target, and A, B, C, D are the target candidates in the current frame. There are two types of information in a target candidate: One is from the target, and the other is from the background. The overlap of the target and the target candidate provides the partial information of the target. That is, the larger the overlap is, the more target information the candidate has, and therefore the closer it is to the target.

Some visual features including color, texture, or silhouette can be used to measure the similarity between the target and the target candidate. Several candidates more similar to the target constitute a smaller local region which contains the target, and then new candidates are set in this region for the next search. Hence these new target candidates are gradually centralized to the target.

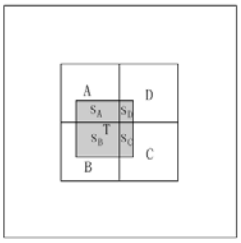


Fig. 1. Overlapping between two rectangles.

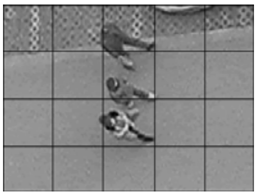


Fig. 2. Mesh on the image.

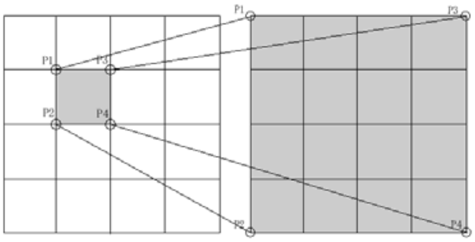


Fig. 3. New region for search.

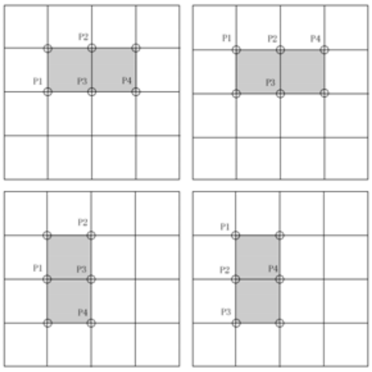


Fig. 4. Other probable distributions of 4 chosen nodes.

B. Mesh Based Search

To update the target candidates with the positions closer to the target as described above, a mesh is created on the current frame whose nodes are spaced proportionally with cells having the same size, as shown in Fig. 2. The nodes of the mesh are set as the target candidates, and the number of them is adapted according to the size of the target to ensure that there are always some candidates overlapping with the target. That is, the smaller the target is, the finer the mesh is.

By calculating the similarity between the target and the target candidate, several nodes which are more similar to the target are obtained. In this paper, as an example, 4 nodes are adequate for our method. So the target is in the region composed of these 4 nodes. As shown on the left of Fig. 3, P1, P2, P3, and P4 denote these 4 nodes, and the target is in the cell enclosed by them. Fig. 4 shows other possible distributions of these 4 nodes when the target is not inside the cell, but just on the edge of it, and the target is in the rectangle region of the 2 cells. The distribution at the top left of Fig. 4 shows that the target is on the side P2-P3 and closer to P3, and similarly that at the bottom right shows the target is on the side P2-P4 and closer to P2. Then a new mesh is created on the local region for the next search as shown on the right of Fig. 3, and the target candidates are updated using the nodes of this new mesh. The search operation is repeated until the required criteria in terms of the distance between the two nodes, the similarity value, or the number of iterations is satisfied. Then the best node in the last mesh search is set as the location of the target.

It is worth noting, if the similarity value is low all the time during the iterations of search, the target is considered missing and the tracking can proceed into the next frame.

III. Implementation

A. Object Representation and Similarity Measurement

Here the color histogram is used to represent object appearance, which suffices for our explanation. Let $\{x_i\}_{i=1 \ldots n}$ be the pixel locations of the target. The function b (b : $R^2 \rightarrow$ {1…m}) associates the pixel at location x_i to the index b(x_i) of the histogram bin corresponding to the color of that pixel. Hence the probability of the color u (u = 1… m) is given by

$$p_u = \frac{1}{n}\sum_{i=1}^{n}\delta(\mathrm{b}(\mathrm{x_i}) - \mathrm{u}) \tag{1}$$

where δ is the Kronecker delta function. Then the normalized p_u is computed as

$$p_u = \frac{p_u}{\sqrt{\sum_{j=1}^{m} p_j^2}} \tag{2}$$

Thus,

The target: $P^t = \{p_u^t\}_{u=1 \ldots m}$ and the target candidate: $P^c = \{p_u^c\}_{u=1 \ldots m}$

Let S_d be the similarity between the target and the target candidate, which is given by

$$S_d = \sum_{u=1}^{m} p_u^t p_u^c \tag{3}$$

The geometric interpretation of Eq. (3) is the cosine of the angle between the m-dimensional normalized vectors (p_1^t,…, p_m^t) and (p_1^c,…, p_m^c). Hence the distance D_s between these two distributions corresponding to the dissimilarity between theses two objects can be defined as

$$D_s = 1 - S_d \tag{4}$$

In the next section, the distance D_s will be used to search the location of the target.

B. Target Localization

According to Section 2, four best nodes are chosen by calculating the distance between the target and the target candidate to narrow the search range. The search operation based on the mesh is repeated until the target location is found as the last best node. The target localization algorithm is presented below.

Given the distribution $\{p_u^t\}_{u=1 \ldots m}$ of the target model:

1) Create a mesh covering the entire image of the current frame, and initialize the positions of the mesh nodes as those of the target candidates. Denote the width and height of the image by w_{img} and h_{img} and those of the target by w_{obj} and h_{obj}. Therefore the number of columns n_{col} and the number of rows n_{row} of the mesh are given by

$$n_{col} = w_{img} \,/\, w_{obj} + 1 \tag{5}$$

$$n_{row} = h_{img} \,/\, h_{obj} + 1 \tag{6}$$

2) Calculate the distances between the target and the candidate by Eqs. (3) and (4).
3) Rank the nodes into a sequence in ascending order according to their distances. There are several cases:
   a) If the distance of the first node is smaller than a certain threshold ε, the location of this node is that of the target in the current frame, stop and initialize the next frame. Go to Step 1.
   b) If the distance of the first node is larger than ε and the number of iterations is also larger than a certain number *k*, the target is missing, stop and initialize the next frame. Go to Step 1.
   c) Otherwise go to Step 4.
4) Choose 4 nodes at the beginning of the sequence to compose a new search region. Create a new mesh on this region and update the positions of target candidates according to this new mesh. Go to Step 2.

Since the search range goes from the entire image to a smaller and smaller region, the location of the target can be evaluated gradually except that it is missing. Thus, the target tracking is achieved by running the search algorithm presented above for each frame.

C. Target Adaptation

Considering that the appearance (here is color) and the scale of the target often changes in time, the target adaptation scheme is achieved by updating the target model and modifying its size.

Let the previous and the current targets be P_{prev}^t and P_{cur}^t, respectively, then the new target is calculated as follows:

$$P_{new}^t = (1 - \lambda_p)\, P_{prev}^t + \lambda_p\, P_{cur}^t \tag{7}$$

where λ_p is 0.1 or smaller to avoid over-sensitive appearance adaptation.

Let the previous size of the target be S^t_{prev}, then the current size of the target S^t_{best} is measured by running the localization algorithm three times, with sizes $S^t = S^t_{prev}$, $S^t = S^t_{prev} + \Delta S$, and $S^t = S^t_{prev} - \Delta S$ where typically $\Delta S = 0.1 S^t_{prev}$. S^t_{best} is one of these three S^t which yields the smallest distance. Similarly,

$$S^t_{new} = (1 - \lambda_s)\, S^t_{prev} + \lambda_s\, S^t_{best} \qquad (8)$$

where S^t_{new} is the new size of the target.

IV. Experiments

To evaluate the performance of the proposed method, the mesh-shrink-based visual tracker has been implemented using C# and applied to many sequences. Meanwhile, the mean shift tracker is also implemented according to [6] as a comparison with the tracker. Since the emphasis in this paper is object tracking, the required target is selected manually. The color histogram of the target with 32 bins is derived by quantizing the image intensity. The distance threshold ε is 0.08, and the maximum iteration number is 5. In all of the experiments the frame size is 360×240 pixels. The algorithm runs comfortably on a normal 2.2 GHz PC at around 20fps on average depending on the size of the target selected. Here some representative examples are presented.

The first example is the Road sequence as shown in Fig. 5 which has 862 frames. The target is a woman marked by a hand-drawn rectangular region in the first image in Fig. 5. The frames 472 and 473 show that the tracker (b) performs the task of tracking well when the camera is moving quickly from left to right (correspondingly the target is moving fast from right to left in the image). The capability of recovering tracking is shown in the frames 505 (where the woman moves out of the image area) and 512 (where the woman reappears in the image). In contrast, the mean shift tracker (a) fails to keep tracking in the same scenario.

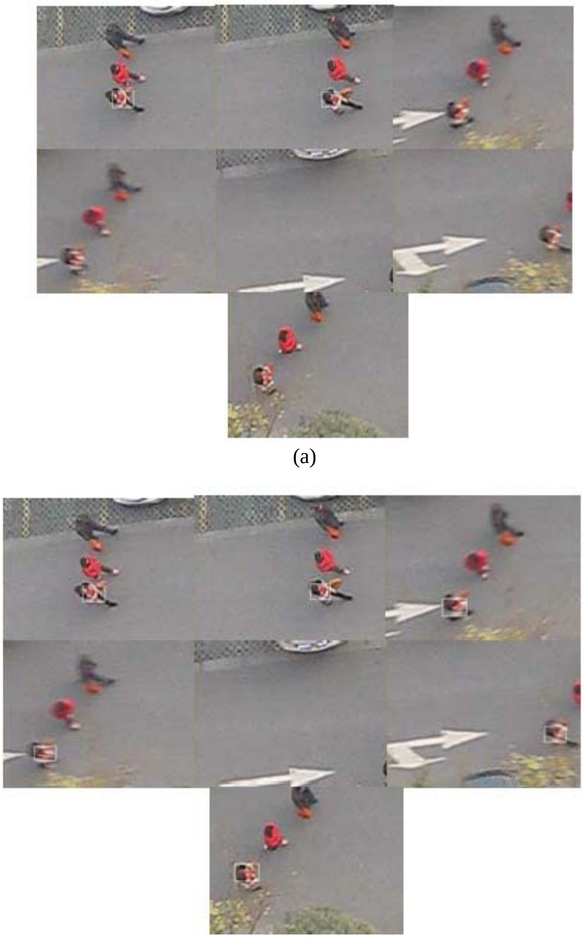
The Park sequence (Fig. 6) is the second example which has 532 frames where the camera is close to the target comparatively. The tracker also achieves satisfactory tracking results, while the mean shift tracker loses the target from the frame 32 due to the exit of target even if the target reenters the field of view later.

In addition, the tracker is applied to track a toy as shown in Fig. 7. The Toy sequence is obtained by extracting the frames from the video at 2fps, and thus the target moving distance between the frames 16 and 17 is very long (almost half of the image height). The frames 31 and 32 show that the tracker recovers the tracking when the target is fully occluded and then reappears at any location of the image. Apparently, in this special context, the mean shift tracker can not perform tracking effectively.

The Orange sequence (Fig. 8) is used to demonstrate the procedure of search for the target localization. Each search region is marked by a green rectangle in the frame. The target is an orange on the ground as shown in the first image of Fig. 8. To find the location corresponding to the target in the current frame, the tracker executes 3 iterations (3 green rectangles, big to small) to search in the frame 29, and 4 in the frame 107.

V. Conclusion

We have proposed a new tracking method whose main idea of the approach is using the mesh and its nodes to narrow the search region and finally find the location of the target. Since the method does not need to initialize an estimated location of the target but searches for the target from the entire image to a smaller and smaller local region, it can track fast moving object and recover the tracking when the target is missing and reappears in the next frames. The method does not require complicated computation and thus can be applied to the environment which permits only limited computing resources.


(a)

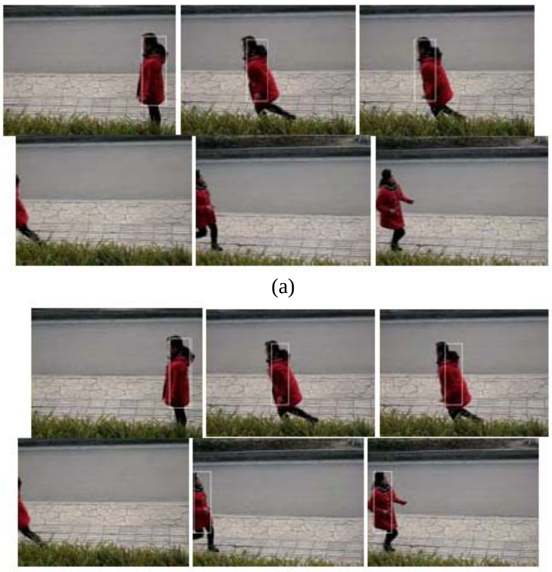
(b)

Fig. 5. Road sequence. The frames 1, 50, 472, 473, 505, 512, and 521 are shown. (a) The mean shift tracker. (b) The mesh shrink tracker.

For demonstration purposes, here only color is used to represent the target, which limits the performance of the tracker when object appearance and illumination changes. Particularly, there may be some objects very similar to the target in the image. However, the approach can be improved by adding and estimating other visual features or devising a corresponding target description with these visual features.

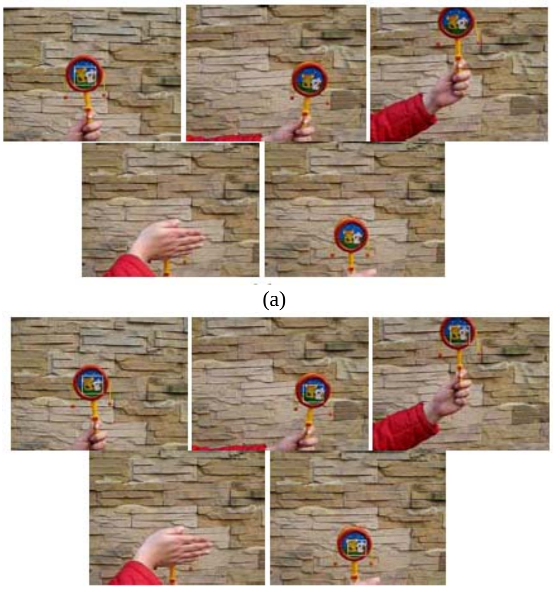
In many applications (surveillance and monitoring, for example), even if the target moves extremely fast, the target location changes slowly in the successive frames because of the high sampling rate of the video. Therefore, by simply calculating a part of the nodes of the mesh in the interested or estimated region, the computational cost can be reduced dramatically, especially when the target is relatively small.

Although only one target is tracked in our experiments carried out for this paper, the proposed approach can also be used to track multiple targets through determining a corresponding mesh for each target and then searching for them in parallel. Whereas, its efficiency needs to be investigated further and its combination with current multiple target tracking methods is a promising line of future work.


(a)

(b)

Fig. 6. Park sequence. The frames 1, 15, 16, 32, 57, and 60 are shown. (a) The mean shift tracker. (b) The mesh shrink tracker.


(a)

(b)

Fig. 7. Toy sequence. The frames 1, 16, 17, 31, and 32 are shown. (a) The mean shift tracker. (b) The mesh shrink tracker.

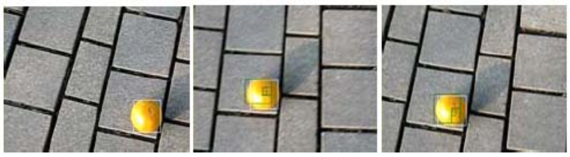


Fig. 8. Orange sequence. The frames 1, 29, and 107 are shown.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Grant No. 40672203) and the Doctoral Innovation Foundation of Southwest Jiaotong University.

References

[1] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” *ACM Computing Surveys*, vol. 38, no. 4, pp. 13, Dec. 2006.

[2] F. Porikli, "Achieving real-time object detection and tracking under extreme conditions," *Journal of Real-time Image Processing*, vol. 1, no. 1, pp. 33-40, Oct. 2006.

[3] Y. Chen, T. Huang, and Y. Rui, “Parametric contour tracking using unscented kalman filter,” in *Proc. of Int’l Conf. Image Processing*, vol. 3, no. 3, pp. 613–616, 2002.

[4] K. Nummiaro, E. Koller-Meier, and L. Van Gool, “An adaptive color-based particle filter,” *Image and Vision Computing*, vol. 21, no. 1, pp. 99-110, 2003.

[5] M. Isard and A. Blake, “CONDENSATION—Conditional density propagation for visual tracking,” *Int’l J. Computer Vision*, vol. 29, pp. 5-28, May 1998.

[6] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-Based object tracking,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564-577, May 2003.

[7] D. Comaniciu, V. Ramesh, and P. Meer, “Real-Time tracking of non-rigid objects using mean shift,” in *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 142–149, June 2000.

[8] C. Yang, R. Duraiswami, and L. Davis, "Efficient spatial-feature tracking via the mean-shift and a new similarity measure,” in *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, 2005.

[9] H. T. Nguyen and A. W. M. Smeulders, "Fast occluded object tracking by a robust appearance filter,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1099-1104, Aug. 2004.

[10] D. P. Huttenlocher, J. J. Noh, and W. J. Rucklidge, “Tracking non-rigid objects in complex scenes,” *IEEE Int’l Conf. Computer Vision*, pp. 93-101, 1993.

[11] A. Yilmaz, X. Li, and M. Shah, “Contour-based object tracking with occlusion handling in video acquired using mobile cameras,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1531-1536, Nov. 2004.

[12] S. Avidan, “Ensemble tracking,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261–271, 2007.

[13] R. Collins, Y. Liu, and M. Leordeanu, “Online selection of discriminative tracking features,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.

[14] J. Lim, D. Ross, R. Lin, and M. Yang, “Incremental learning for visual tracking,” Neural Information Processing Systems (NIPS), 2005.

[15] H. Grabner and H. Bischof, “On-line boosting and vision,” *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 260–267, 2006.

[16] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, “On-line random forests,” *IEEE Int’l Conf. Computer Vision (ICCV), WS on On-line Learning for Computer Vision*, 2009.

[17] A. Wang, G. Wan, Z. Cheng, and S. Li, “An incremental extremely random forest classifier for online learning and tracking,” *IEEE Int’l Conf. Image Processing (ICIP)*, pp. 1449-1452, 2009.

[18] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” *European Conference on Computer Vision (ECCV)*, 2008.

[19] S. Stalder, H. Grabner, L. van Gool, E. Zurich, and K. Leuven, “Beyond semi-supervised tracking: tracking should be as simple as detection, but not simpler than recognition,” *IEEE Int’l Conf. Computer Vision (ICCV), WS on On-line Learning for Computer Vision*, 2009.

[20] C. Leistner, A. Saffari, J. Santner, and H. Bischof, “Semi-supervised random forests,” *IEEE Int’l Conf. Computer Vision (ICCV)*, 2009.

[21] C. Leistner, M. Godec, A. Saffari, and H. Bischof, “On-line multi-view forests for tracking,” *DAGM-Symposium*, pp. 493-502, 2010.

[22] O. Chapelle, B. Scholkopf, and A. Zien, editors, *Semi-Supervised Learning*, MIT Press, Cambridge, MA, 2006.

[23] Y. Abu-Mostafa, “Machines that learn from hints,” *Scientific American*, vol. 272, no. 4, pp. 64–71, 1995.

[24] B. Babenko, M. H. Yang, and S. Belongie, “Visual tracking with online multiple instances learning,” *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2009.

[25] Q. Yu, T. Dinh, and G. Medioni, "Online tracking and reacquisition using co-trained generative and discriminative trackers," *European Conference on Computer Vision (ECCV)*, 2008.

[26] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N learning: bootstrapping binary classifiers by structural constraints," *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2010.

Wei Quan received the BS and MS degrees in control theory and engineering from Southwest Jiaotong University, Chengdu, China, in 2004 and 2007, respectively. He is currently a PhD candidate at Southwest Jiaotong University. His research interests include computer vision, pattern recognition, machine learning, and image processing. He was awarded the excellent student fellowship from 2000 to 2004 at Southwest Jiaotong University. He was also an excellent bachelor graduate of Southwest Jiaotong University in 2004. He holds two patents and has 6 more patents pending.

Jim X. Chen received the BS and MS degrees from Southwest Jiaotong University, Chengdu, China, in 1983 and 1986, respectively. In 1995, he received his PhD degree in Computer Science from the University of Central Florida. He is currently Professor of Computer Science, and the director of the Computer Graphics Lab at George Mason University (GMU), Fairfax, Virginia. He is a senior member of IEEE and a professional member of ACM. His research interests include computer graphics, virtual reality, computer vision, pattern recognition, image processing, visualization, networking, and simulation. He has authored 4 books, edited 2 conference proceedings, published over 100 research papers, and acquired 3 patents.

Nanyang Yu received the BS degree from Jiangxi Normal University, Nanchang, China, in 1983. He received the MS and PhD degree in applied physics and environmental science from Southwest Jiaotong University, Chengdu, China, in 1986 and 2004, respectively. He is currently Professor of Mechanical Engineering at Southwest Jiaotong University, Chengdu, China. His research interests include intelligent control, video analysis and environment protection. He holds 2 patents and has 5 more patents pending.