Directional Eigentemplate Learning for Sparse Template Tracker

Hiroyuki Seto, Tomoyuki Taguchi, and Takeshi Shakunaga

Okayama University

Abstract. Automatic eigentemplate learning is discussed for a sparse template tracker.Using an eigentemplate learned from multiple sequences, a sparse template tracker can efficiently track a target that changes appearance. The present paper provides a feasible solution for eigentemplate learning when multiple image sequences are available. Two types of eigentemplates are compared in the present paper, namely, a single eigentemplate,and a set of directional eigentemplates.The single eigentemplate simply consists of all images learned from multiple sequences.On the other hand, directional eigentemplates are obtained by decomposing the single eigentemplate into three directions of the face poses. The sparse template tracker is also expanded to directional eigentemplates.Finally, the effectiveness of the provided solution is demonstrated in the learning and tracking experiments. The experimental results indicate that directional learning works well with small seed data,and that the directional eigentracker works better than the single eigentracker.

1 Introduction

Object tracking is one of the most significant problems in computer vision [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]. Considerable research on unknown objects and known objects has been conducted in a lot of applications.Among them,some robust algorithms were proposed for the tracking based on the eigenspace techniques [5] with combining iterative projections and outlier detection. The iterative projection approaches, however, often suffer from time-consuming implementation and the "breakdown point" problems. In order to solve these problems, a sparse eigentemplate tracker was proposed by Shakunaga et al. [10] based on a non-eigentemplate tracker [8]. In the tracker proposed by Shakunaga et al., a particle filter is used in order to aviod iterative calculations. Shakunaga-Noguchi [16] demonstrated that the tracker [8] could be converted to an adaptive tracker by combining their sparse template tracking and an on-line learning technique of Black-Jepson [1]. Sakabe et al. [17] used the adaptive tracker for learning the eigentemplate used in the sparse eigentemplate tracking and demonstrated that the effectiveness of the learning. A tracker with memory-based particle filter was developed from the sparse template tracker [8] by Mikami et al. [12,13].In their tracker, past history storage is used for robust tracking by calculating the posterior face position with this storage.However, this tracker requires a large

Y.-S. Ho (Ed.): PSIVT 2011, Part II, LNCS 7088, pp. 104–115, 2011.

amount of past data.In order to carry out robust eigentemplate tracking using a small amount of image data, the present research is based on the tracker proposed in [17].

In [17], an eigentemplate is learned from an image sequence. If the eigentemplate is learned from multiple sequences, then the tracker is expected to track more efficiently in case of changes in appearance. In the generation of eigentemplates, two types of eigentemplates, namely, a single eigentemplate and directional eigentemplates, are compared.Directional eigentemplates are obtained by classifying face templates into three directions. When the directional eigentemplates are used for eigentracker, the tracker selects appropriate eigentemplate in each frame and reduces the overmatching effect of the unified eigentemplate with respect to inappropriate poses. In addition, since the tracker evaluates the poses of the target with three eigentemplates, the tracker can avoid converging to a local minimum.

2 Learning Eigentemplate for Sparse Template Tracker

2.1 Adaptive Sparse Template Tracker

Automatic eigentemplate learning is possible, if an adaptive tracker is provided. If the tracker can carry out complete and accurate tracking for the case in which changes in appearance occur, then the problem of eigentemplate learning is reduced to a simple problem. However, since there is no such complete tracker, we must develop a learning method for a given tracker. The present paper basically uses an adaptive sparse template tracker formulated in Shakunaga-Noguchi [16]. The tracker is not complete but good since it combines the sparse template tracker and the WSL model proposed by Jepson et al. [1] for implementing an adaptive real-time tracker. In their formulation, the WSL model is applied to each pixel value, and an adaptive template, called the WSL template, is updated by the on-line EM algorithm.During the updating phase, an image estimated by the sparse template tracker is used to update the WSL model.Then, a dense template is constructed from the adaptive template, and the sparse template is updated. Thus, the tracker can carry out adaptive real-time tracking and sequential learning.

2.2 Learning Eigentemplate

This paper basically uses the eigentemplate learning formulated by Sakabe et al. [17]. Their method is summarized as follows: In tracking with the adaptive tracker, the estimated image is evaluated at each frame.

Their formulation uses the following notation.Let $\mathbf{Y}_t$ and $\tilde{\Phi} = [\Phi \;\; \overline{\mathbf{x}}]$ denote an input image and the eigentemplate at time t, respectively. Let $Q_i (i = 1, 2, 3, 4)$ denote partial indicator matrices which correspond to four quadrant regions of the entire template, respectively, and let $Q_0 = Q_1 + Q_2 + Q_3 + Q_4 = I$ hold. Then, for $i = 0, 1, 2, 3, 4$, a projection of a (partial) image $Q_i\mathbf{Y}_t$ onto the (homogeneous) eigentemplate, $\tilde{\Phi}$, is represented as $\mathbf{Y}'_{ti} = \tilde{\Phi}(Q_i\tilde{\Phi})^+\mathbf{Y}_t$. Thus, the correlation $C_i(\mathbf{Y}_t, \mathbf{Y}'_{ti})$ is calculated between $Q_i\mathbf{Y}_t$ and $Q_i\mathbf{Y}'_{ti}$.

Although Sakabe et al. [17] provided an image selection rule, a simpler rule is used in the present paper. That is, when the following condition is satisfied, the current input image $\mathbf{Y}_t$ is appended to the learning set. Otherwise, the current image is not appended to the learning set.

$$\min_{i=0,1,2,3,4} C_i(\mathbf{Y}_t, \mathbf{Y}'_{ti}) < 0.7 \tag{1}$$

2.3 Sparse Eigentemplate Tracker

When an eigenspace is constructed from a set of normalized template images, it is used as an eigentemplate. The formulation of sparse template matching [16] can be generalized to eigentemplate matching as follows:

Let $\overline{\mathbf{x}}$ and Φ denote the mean vector and a matrix composed of the m most significant eigenvectors. Let $\tilde{\Phi}$ denote $[\Phi \;\; \overline{\mathbf{x}}]$. Then, the eigentemplate matching problem is formulated as follows:

$$\arg\min_{T\in\{T\}} \epsilon = \arg\min_{T\in\{T\}} \widehat{\rho}(\frac{1}{\beta}P[\tilde{\Phi}\tilde{\mathbf{y}}^* - T\mathbf{Y}]), \tag{2}$$

where $\widehat{\rho}(\mathbf{x})$ indicates the summation of the Geman-McClure function, $\tilde{\mathbf{y}}^*$ is an $(m+1)$-vector calculated for each T as $\tilde{\mathbf{y}}^* = (P\tilde{\Phi})^+T\mathbf{Y}$, and β is a normalization parameter calculated for each T.

3 Learning and Tracking for Multiple Sequences

3.1 Eigentemplate Learning for Multiple Sequences

Once an eigentemplate is learned from an image sequence, the tracker can track similar sequences using the eigentemplate. If an eigentemplate consists of images learned from more varied image sequences, the tracker is expected to track against more varied changes of appearances. Therefore, in the present paper, two types of eigentemplates learning are considered for multiple sequences. In 3.2, simple expansion of Sakabe et al.'s method is discussed. The other expansion is discussed in 3.3, where a set of directional eigentemplates are learned from multiple sequences.

3.2 Simple Expansion of Sakabe et al.

As a simple expansion of the eigentemplate learning two types of learning should be considered for multiple sequences.

The first type is parallel learning, in which each sequence is first used to obtain a learning set of images independent of the other sequences. Then, the learning sets,selected from each sequence, are merged to generate a single eigentemplate.

Therefore, the result of parallel learning is invariant with respect to the order of image sequences.On the other hand, the result may be redundant since each parallel learning is carried out without any initial information.

The second type is cascade learning, which learns one sequence after another. Since cascade learning starts from the eigentemplate obtained from other sequences, only a small number of images are learned in each sequence. On the other hand, the results of learning may depend on the order of sequences used in learning. In the present paper, the parallel learning is used for multiple learning because no order problem is included in the learning.

3.3 Directional Eigentemplates

Once the single eigentemplate is learned from multiple sequences, the tracker is expected to track efficiently for all of the changes in appearance in multiple sequences. However, if the eigentemplate is constructed from too large a set of various images without considering the face poses, the tracker may excessively match inappropriate poses of the target. Actually, some combinations of multiple sequences often result in inefficient tracking. In such cases, the single eigentemplate causes unstable tracking when the pose estimation error is generated.

In order to avoid such a critical problem, we consider decomposing an eigentemplate into three directions of the face (front, left, right) as shown in Fig. 1. We call this set of eigentemplates “ directional eigentemplates”. By decomposing the eigentemplate, the tracker is expected to select an appropriate eigentemplate for the poses of the target.Therefore, the tracker will reduce overmatching and avoid unstable tracking.

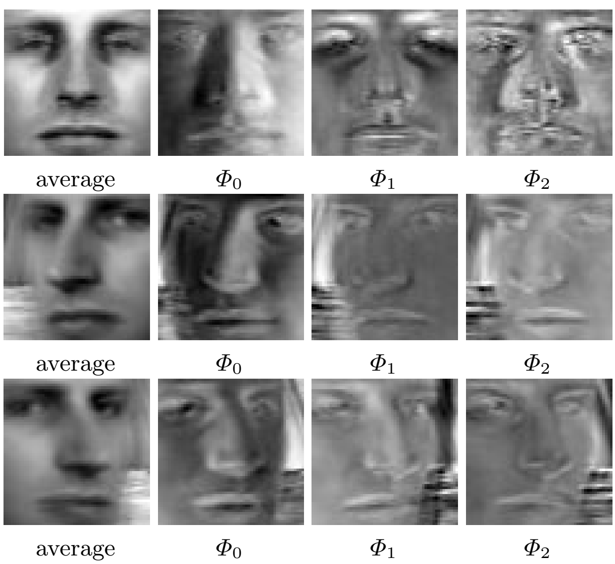


Fig. 1. Directional eigentemplates(front, left, right)

3.4 Automatic Learning of Directional Eigentemplates

Learning of directional eigentemplates consists of two parts, "direction judgment" and "learning judgment". In order to discuss how to select the learning set, we use the following notation. Let $[\Phi_d, \overline{\mathbf{x}}_d]$ denote the seed directional eigentemplates of each direction, and let $\mathbf{Y}_{0d}$ denote the initial image of each direction (d=f,l,r). The seed directional eigentemplates are provided in order to determine the direction of each frame. Each of the eigentemplates consists of a few images captured under different lighting conditions.

In the direction judgment, a projection $\mathbf{Y}'_{td}$ of input $\mathbf{Y}_t$ is made for each seed directional eigentemplate. Then, the correlation between the input and the projection, $C_0(\mathbf{Y}_t, \mathbf{Y}'_{td})$, is calculated in each direction d.The direction providing the highest correlation is determined as the direction of frame t.

After the direction judgment, the learning judgment is performed. Four correlation are calculated between each quadrant image$\mathbf{Y}_{ti}$(i=1,2,3,4) and the partial projection image$\mathbf{Y}'_{ti}$.Let the i-th correlation be denoted as $C_i(\mathbf{Y}_{ti}, \mathbf{Y}'_{ti})$. If the correlation satisfies the condition(1), then the current input image $\mathbf{Y}_t$ is appended to the learning set. Otherwise, the current image is not appended to the learning set.

3.5 Expansion for Eigentemplate Tracking

Next, the tracker is expanded to directional eigentemplates. Let $\overline{\mathbf{x}}_i$ and Φ_i(i=f,l,r) denote the mean vector and a matrix composed of the m most significant eigenvectors for each direction. Let $\tilde{\Phi}_i$ denote $[\Phi_i \;\; \overline{\mathbf{x}}_i]$. The particle filter first evaluates each particle and then selects the optimal particle in each direction as follows:

$$\arg\min_{T\in\{T\}} \epsilon = \arg\min_{T\in\{T\}} \widehat{\rho}(\frac{1}{\beta} P[\tilde{\Phi}_i \tilde{\mathbf{y}}^* - T\mathbf{Y}]), \quad (3)$$

Next, the proportion of each direction in the top particles are calculated by comparing ϵ, where top particles are a set of the best particles used to estimate the position of the next frame. Finally, the top particles are selected according to the proportion. In the selection phase, the top particles are basically selected from the direction that provides the highest proportion. When the highest proportion is less than 0.80, top particles are selected from the highest and the second directions. In this way, the tracker is expected to perform stable pose estimation as the pose of the target changes.

4 Experiments

4.1 Learning Directional Eigentemplates

Let us perform eigentemplate learning on the image sequences of Cascia et al. [7]. In this experiment, 30 trials of a set of directional learning were first carried out for each **jal** sequence. For directional learning, a set of directional images, as shown in Fig. 2, was provided for the seed directional eigentemplates. Since a

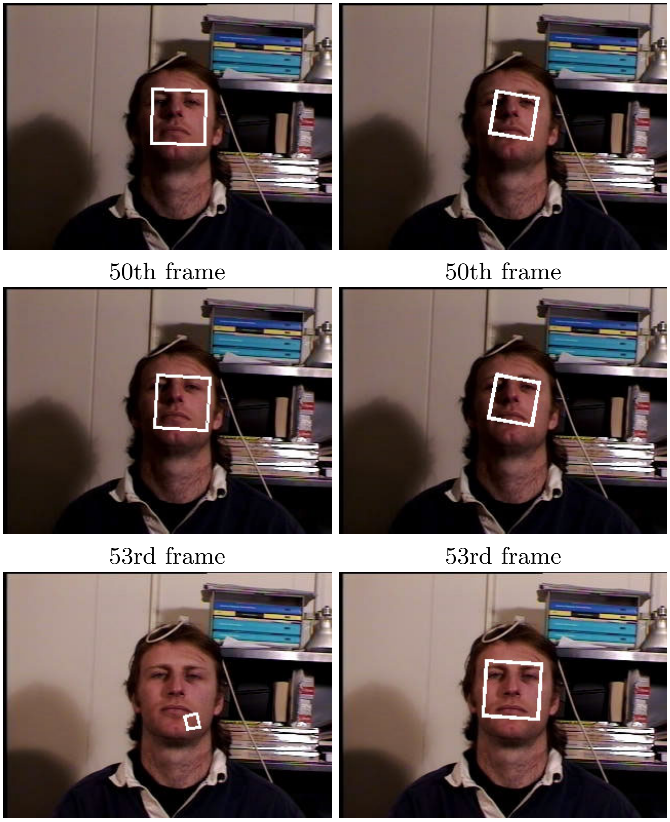
Fig. 2. Images for the seed directional eigentemplates. Three seed images are shown in each row. The front, left, and right directions are shown from top to bottom.

set of directional eigentemplates was constructed in each trial, a total of 30 sets of directional eigentemplates were constructed from each **jal** sequence.

After learning, each sequence was tracked by the sparse eigentracker with a set of learned eigentemplates. In tracking experiment, we used the eigentemplates learned from **jal3,4,5,6** and **jal9** because these sequences included appropriate changes in appearances. In other words, **jal3,4**, and **9** include up-and-down sequences, and **jal5** and **6** include right-and-left sequences. In some cases, the number of direc-

Table 1. Success rates(%) of learning and tracking with “s”ingle eigentemplate(S) and “d”irectional eigentemplates(D).(X) under “D” indicates direction(s) used for directional tracking(“f”ront(F),”l”eft(L),”r”ight(R)). Test sequences were tracked with the eigentemplate learned from each learning sequence.

	learning sequence									
	jal3		jal4		jal5		jal6		jal9	
test sequence	S	D (F)	S	D (F)	S	D (LFR)	S	D (LFR)	S	D (F)
jal1	30	17	91	93	74	97	72	50	61	70
jal2	28	13	87	73	100	100	99	100	100	100
jal3	46	43	81	90	8	47	3	57	84	53
jal4	71	70	97	97	0	60	0	57	73	33
jal5	0	0	2	3	100	100	85	43	81	57
jal6	0	7	2	10	83	90	100	100	33	33
jal7	51	20	53	77	100	100	100	100	99	97
jal8	22	7	85	87	100	100	100	100	93	90
jal9	8	10	88	83	0	100	0	100	95	93
average	28.4	20.7	65.1	66.7	62.8	87.8	62.1	78.5	79.9	69.6



50th frame 50th frame

53rd frame 53rd frame

65th frame 65th frame

Fig. 3. Comparison of tracking **jal4** with two types of eigenfaces learned from **jal6**(left: tracking with a single eigentemplate, right: tracking with directional eigentemplates)

tional eigentemplates varies from the results of the learning (the directions used are listed in Table 1). After tracking, the results were compared with the correct data for each sequence. In the evaluation, the averages of the estimation errors in distance [pixel], rotation [deg], and scale, are used for evaluation. If the averages satisfy $\overline{d}$<4.0 [pixel], $\overline{r}$<4.0 [deg], and $\overline{s}$<0.2, then the tracking is judged as "success". Table 1 shows the success rates of the tracking with directional eigentemplates and those with single eigentemplate. The success rates indicate how well the eigentemplates that are effective for tracking are learned.

In the learning phase, for the most part, the images were correctly learned with respect to the directions. No confusion occurred between the right and left directions. In a few cases, images that might be regarded as front images were learned as left images. However, the eigentemplate could cover new appearances that the eigentemplate did not cover before the learning.

As shown in Table 1, the results of the directional eigentemplates were sometimes lower than those of single eigentemplates. However, the average success

Table 2. Success rates (%) of learning and tracking.The test sequences were tracked with the eigentemplates learned from each learning set. The configuration of this table is the same as that of Table 1.

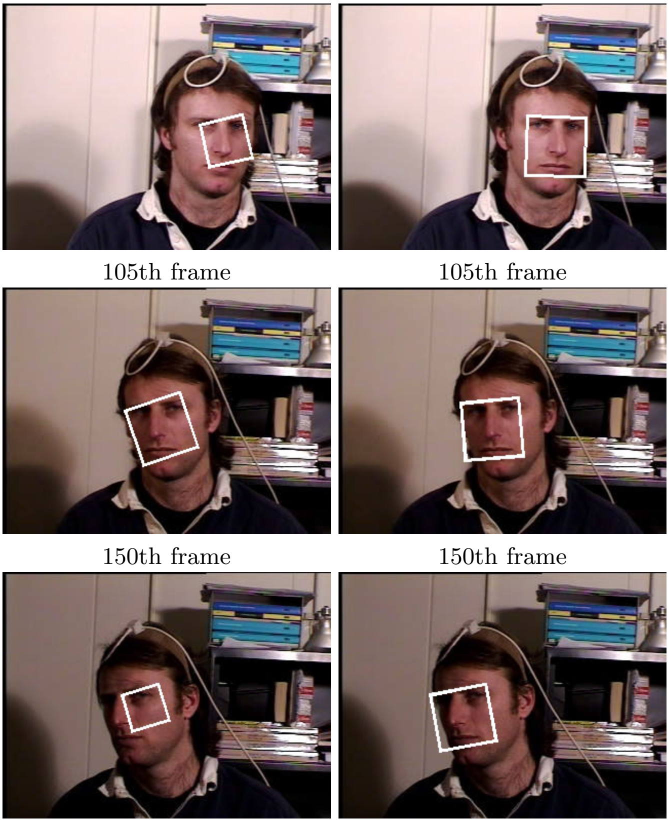
test sequence	learning sets: jal3+jal5		jal4+jal6		jal4+jal5 +jal6		jal4+jal6 +jal9		jal3+jal4 +jal5+jal6		jal4+jal5 +jal6+jal9		jal3+jal4 +jal5+jal6 +jal9	
	S	D	S	D	S	D	S	D	S	D	S	D	S	D
jal1	40	40	100	80	90	83	90	77	77	87	93	87	83	90
jal2	53	67	100	97	100	100	100	100	87	100	100	100	100	100
jal3	60	40	87	83	57	67	80	97	83	73	50	50	87	83
jal4	77	70	100	97	100	93	100	100	100	100	100	83	100	100
jal5	100	97	0	20	90	100	0	27	63	100	93	100	63	100
jal6	37	93	100	100	77	100	100	100	67	100	87	100	63	100
jal7	67	63	100	97	100	100	100	100	97	100	100	100	100	100
jal8	67	73	100	97	100	100	100	100	87	100	100	100	100	100
jal9	67	70	100	97	100	100	100	100	93	100	100	100	100	100
average	63.0	67.8	87.4	85.2	90.0	93.7	85.6	88.9	83.7	95.6	91.5	91.1	88.5	97.0

rates were better with directional eigentemplates than those with a single eigentemplate.In particular, the averages of **jal5** and **jal6** increase considerably. The directional eigentemplates improved pose estimation by selecting an appropriate eigentemplate in each frame.

Examples of the tracking with two eigentemplates are as shown in Fig. 3. With a single eigentemplate, tracking was stable until the 50th frame. However, the pose estimation error occurred at the 53rd frame, and the error continued.Finally, the tracker converged to local minimum at the 65th frame. On the other hand, the pose estimation error occurred until the 53rd frame in the tracking with directional eigentemplates. However, the tracker gradually corrected the pose of the target, and the error was resolved at the 65th frame. The results show that the directional eigentemplates are efficient for the pose estimation error on the tracking and provide efficient tracking.

4.2 Learning Eigentemplates from Multiple Sequences

The results of the previous experiment revealed that an eigentemplate learned from an image sequence can track other sequences to a certain extent. However, the eigentemplate often fails to track certain sequences because the eigentemplate includes information included in the learning sequence. In the single eigentemplate tracking, the eigentemplate learned from **jal3** could not track **jal6**, whereas the eigentemplate learned from **jal5** could track **jal6**.Therefore, if the eigentemplate is learned from **jal3** and **jal5**, the tracker is expected to carry out stable tracking **jal6**. In the following experiment, we tried to construct the eigentemplate from multiple sequences.



105th frame 105th frame

150th frame 150th frame

185th frame 185th frame

Fig. 4. Comparison of tracking **jal6**(left:tracking with directional eigentemplate learned from **jal3**,right:tracking with directional eigentemplates learned from **jal3+jal5**)

In the experiment, we also compared the two types of eigentemplates using tracking **jal** sequences. After learning each sequence, the images learned from some sequences were combined to include the information of other sequences. Therefore, the learned images included different poses, such as **jal3** + **jal5**.(Images learned from **jal3** include up-and-down information, and images learned from **jal5** include right-and-left information.) The combinations of the learned images were as shown in Table 2. The tracking was carried out 30 times for each sequence with each eigentemplate.

Table 2 compares the success rate of tracking using a single eigentemplate(S) and the directional eigentemplates(D). In the table, the results were evaluated similar to the manner described in 4.2.

The results of tracking are shown in Table 2, which indicates that the tracker can carry out stable tracking when the eigentemplate is learned from multiple sequences.For example, the results for tracking **jal6** were better with the

Fig. 5. Comparison of tracking **jal5** with two types of eigentemplates learned from **jal4+jal6+jal9**(left:tracking with single eigentemplate, right:tracking with directional eigentemplates)

eigentemplate learned from **jal3** and **jal5** than that learned from each sequence, as shown in Fig. 4. The eigentemplate adapted to new appearance changes. However, in some cases, the results became worse with additional eigentemplate learning. For example, although, **jal5** was tracked stably with the eigentemplate learned from **jal6**, the tracker perform out the stable tracking when the eigentemplate was learned from **jal4** and **jal6**. The images learned from **jal4** and **jal6** were inappropriate for tracking jal5, since the combinations of face positions and lighting conditions obtained from **jal4** and **jal6** were different from those of **jal5**. Therefore, the tracker could not estimate the correct position.

Comparing the results of tracking using two different types of eigentemplates, the tracking with directional eigentemplates worked better than that with a single eigentemplate. In some case, results of directional eigentemplates were lower than those of a single eigentemplate. However, the average success rates was higher than a single eigentemplate for most combinations, which indicates that the directional eigentemplates could use the information included in multiple sequences more effectively than the single eigentemplate.

The example shown in Fig. 5 indicates how the tracker worked using the two different types of eigentemplates. With single eigentemplate, a pose estimation error occurred at the 45th frame, after which the error continued.Finally, the tracker converged at a local minimum at the 60th frame. In contrast, for the same sequence, the tracker using directional eigentemplates could track the target correctly. In the sequence, when the target faced toward the right, the tracker selected the right eigentemplate. Therefore, the tracker could efficiently estimate the appropriate poses and track using the directional eigentemplates.

5 Conclusion

Directional eigentemplate learning was discussed for a sparse template tracker. In the learning phase, the adaptive tracker adaptively tracks a target for the eigentemplate learning. If an eigentemplate is decomposed into directional eigentemplates, then the sparse eigentemplate tracker can estimate the pose of the target with an appropriate eigentemplate.

The experimental results show that the directional learning worked well using a few initial images, and the tracking worked well using directional eigentemplates learned from single image sequences. In the second experiment, the tracker with directional eigentemplates was shown to work better than the single eigentemplate for multiple learning.In some cases, however, the tracker did not work well. In the future, we would like to solve the problems involved in these cases and develop a more stable on-line learning method.

This work has been supported in part by a Grant-In-Aid for Scientific Research (No.20300067) from the Ministry of Education, Science, Sports, and Culture of Japan.

References

1. Jepson, A.D., Fleet, D.J., El-Maraghi, T.F.: Robust online appearance models for visual tracking. IEEE Trans. Pattern Analysis and Machine Intelligence 25(10), 1296–1311 (2003)
2. Isard, M., Blake, A.: Condensation – conditional density propagation for visual tracking. International Journal of Computer Vision 29(1), 5–28 (1998)
3. Williams, O., Blake, A., Cipolla, R.: A sparse probabilistic learning algorithm for real-time tracking. In: Proc. ICCV, pp. 353–360 (2003)
4. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Analysis and Machine Intelligence 24(5), 603–619 (2002)
5. Black, M., Jepson, A.: Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. International Journal of Computer Vision 26(1), 63–84 (1998)
6. Avidan, S.: Ensemble tracking. IEEE Trans. Pattern Analysis and Machine Intelligence 29(2), 261–271 (2007)

7. Cascia, M.L., Sclaroff, S., Athitsos, V.: Fast,reliable head tracking under varying illumination: An approach based on robust registration of texture-mapped 3d models. IEEE Trans. Pattern Analysis and Machine Intelligence 22(4), 322–336 (2000)
8. Matsubara, Y., Shakunaga, T.: Sparse template matching and its application to real-time object tracking. IPSJ Transactions on Computer Vision and Image Media 46, 60–71 (2005) (in japanese no.sig9(cvim11))
9. Satake, J., Shakunaga, T.: Multiple target tracking by appearance-based condensation tracker using structure information. In: Proc. International Conference on Production Research, vol. 3, pp. 294–297 (2004)
10. Shakunaga, T., Matsubara, Y., Noguchi, K.: Appearance tracker based on sparse eigentemplate. In: Proc. Int'l Conf. on Machine Vision & Applications, pp. 13–17 (2005)
11. Oka, Y., Kuroda, T., Migita, T., Shakunaga, T.: Tracking 3d pose of rigid object by sparse template matching. In: Proc. the 5th International Conference on Image and Graphics, ICIG 2009, pp. 390–397 (2009)
12. Mikami, D., Otsuka, K., Yamato, J.: Memory-based particle filter for face pose tracking robust under complex dynamics. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), pp. 999–1006 (2009)
13. Mikami, D., Otsuka, K., Yamato, J.: Memory-Based Particle Filter for Tracking Objects with Large Variation in Pose and Appearance. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 215–228. Springer, Heidelberg (2010)
14. Murphy-Chutorian, E., Trivedi, M.M.: Hyhope: Hybrid head orientation and position estimation for vision-based driver head tracking. IEEE Trans. Pattern Analysis and Machine Intelligence 31(4), 607–626 (2009)
15. Oka, Y., Shakunaga, T.: Sparse eigentracker augmented by associative mapping to 3d shape. In: Proc. IEEE Conference on Automatic Face and Gesture Recognition (FG 2011), pp. 649–656 (2011)
16. Shakunaga, T., Noguchi, K.: Robust tracking of appearance by sparse template adaptation. In: Proc. 8th IASTED Int'l Conf. on Signal and Image Processing, pp. 85–90 (2006)
17. Sakabe, K., Taguchi, T., Shakunaga, T.: Automatic Eigentemplate Learning for Sparse Template Tracker. In: Wada, T., Huang, F., Lin, S. (eds.) PSIVT 2009. LNCS, vol. 5414, pp. 714–725. Springer, Heidelberg (2009)