Face Tracking by Maximizing Classification Score of Face Detector Based on Rectangle Features

Akinori HIDAKA
University of Tsukuba
1-1-1 Tennodai, Tsukuba-shi,
Ibaraki-ken, 305-8577 Japan
hidaka.akinori@aist.go.jp

Kenji NISHIDA
National Institute of Advanced
Industrial Science and Technology (AIST)
AIST Central 2, 1-1-1 Umezono,
Tsukuba-shi, Ibaraki-ken, 305-8568 Japan

Takio KURITA
National Institute of Advanced
Industrial Science and Technology (AIST)
AIST Central 2, 1-1-1 Umezono,
Tsukuba-shi, Ibaraki-ken, 305-8568 Japan

Abstract

Face tracking continues to be an important topic in computer vision. We describe a tracking algorithm based on a static face detector. Our face detector is a rectangle-feature-based boosted classifier, which outputs the confidence whether an input image is a face. The function that outputs this confidence, called a score function, contains important information about the location of a moving target. A target that has moved will be located in the gradient direction of a score function from the location before moving. Therefore, our tracker will go to the region where the score is maximum using gradient information of this function. We show that this algorithm works by the combination of jumping to the gradient direction and precise search at the local region.

1 Introduction

Efficient real-time tracking in complex environments is still a challenging task for the vision community. Target tracking is a key component in real-time applications such as surveillance, smart rooms, teleconferencing, and user interfaces. There are typical approaches for tracking moving objects. Feature-based tracking relies on the persistence of image features such as image curves [1, 2] or image appearance [3].Likewise, model-based tracking uses curves [4, 5] or appearance [3]. Particle filters [6, 7] or mean shift [8, 14] are often used to treat target dynamics. For face tracking in a crowded scene, however, the tracking system needs to rely more on target representation than on target dynamics.

On the other hand, object detection algorithms using support vector machines (SVMs) [9, 10] or boosting [11, 12] have become fast enough to run at almost any video rate. In particular, the face detector proposed by Viola and Jones is capable of processing images rapidly while achieving high detection rates. They introduced a new image presentation called Integral Image, which allows rectangle features used by the detector to be computed very quickly. A small number of rectangle features are selected to construct an efficient classifier by using the AdaBoost algorithm. However, the computational complexity of the target tracker is critical for most applications. Only a small percentage of system resources is allocated for tracking because the rest must be assigned to high-level tasks such as recognition, trajectory interpretation, and reasoning. Thus, it is important to develop a target tracking algorithm based on such fast object detectors. Recently, Avidan [13] proposed a tracking algorithm called Support Vector Tracker (SVT). In the algorithm, a target is tracked by maximizing a classification score computed by the target detector, which is designed using an SVM. He also proposed a tracking algorithm using mean shift [14].

Humans continuously direct their eyes toward interesting points to recognize an object. The moving objects are tracked by combining saccadic eye movement and smooth-pursuit eye movement in human vision. In saccadic eye movement, eyes are directed toward the rough location of the target and accurate positions are maintained by smooth-pursuit eye movement. We think that this combination of saccadic eye movement and smooth-pursuit eye move-

Proceedings of the Fourth IEEE International Conference on Computer Vision Systems (ICVS 2006)
0-7695-2506-7/06 $20.00 © 2006 IEEE

ment is probably important to achieve accurate and efficient tracking.

In this paper, we adapt an approach used in SVT and use a face detector proposed by Viola and Jones [11, 12] instead of an SVM-based face detector. This tracker is used as a rough-search-like saccadic eye movement and is combined with precise search in neighboring regions.

2 Boosted Detector Constructed by Rectangle Features

The face detector proposed by Viola and Jones [11, 12] became very successful because of its robustness and high-speed. We used three key features of their work, $integral\ image$, $rectangle\ features$, and $boosting$, for our face detector. We added two modifications to these features. First, the classification function is changed from the binary threshold function to the sigmoid function because we must obtain the derivative of the classification function for our tracking algorithm. Second, in the boosting algorithm, the two-phases optimization is reduced to the single-phase optimization to improve generalization performance. These features are introduced in this section.

2.1 Integral Image

Integral image, also called summed area table [15], is used to reduce the computational cost to determine the mean value of a rectangular region. Integral image is a reference table that contains the double integral of $i(x, y)$, which is the intensity distribution of an input image. Each point (x, y) of the input image is assigned an $ii(x, y)$ as

$$ii(x, y) = \int_0^y \int_0^x i(x, y)dx\, dy.$$

Therefore, using reference table $ii(x, y)$, the mean intensity of any rectangular region q in the input image is calculated as

$$\begin{aligned} mean &= \frac{1}{wh}\{ii(x_q + w, y_q + h) - ii(x_q + w, y_q) \\ &- ii(x_q, y_q + h) + ii(x_q, y_q)\}, \end{aligned}$$

where $q = \{(x, y)|x_q \le x \le x_q + w, y_q \le y \le y_q + h\}$, and (x_q+w, y_q+h), (x_q+w, y_q), (x_q, y_q+h), and (x_q, y_q) are vertices of q (Fig. 1). Hence, the mean intensity of rectangular regions of any size and position can be calculated from only four times of table reference.

2.2 Rectangle Features

A rectangle feature is constructed by n small rectangles that are the same size ($w \times h$ pixels) and neighbor each other (See left four patterns of Fig. 2.). Rectangle features classify the target image according the difference in mean intensity of these small rectangular regions. Let us consider a situation where target image I_q, which is to be classified, is located in local region q in whole image plane F, and small rectangles $\{s_k\}_{k=1}^n$ are in q, as shown in Fig. 2. s_k is defined as

$$\begin{aligned} s_k &= (x_k, y_k, w, h) \\ &= \{(x, y) \in q | x_k \le x \le x_k + w, y_k \le y \le y_k + h\}. \end{aligned}$$

Therefore, rectangle features has five degrees of freedom: small rectangle number n, place (x_k, y_k), and size (w, h). A sub-image located at s_k is defined as $I_{s_k} = \{i(x, y)|(x, y) \in s_k\}$. Therefore, the mean intensity of I_{s_k} can be defined as

$$\begin{aligned} m(I_{s_k}) &= \frac{1}{wh} \iint_{s_k} i(x, y)dx\, dy \\ &= \frac{1}{wh} \int_{y_k}^{y_k+h} \int_{x_k}^{x_k+w} i(x, y)dx\, dy. \end{aligned} \quad (1)$$

Each s_k is assigned label $c_k \in \{-1, 1\}$, which implies s_k is "white" or "black" as shown in Fig. 2. Now we represent a single rectangle feature as $r = \{s_k, c_k\}_{k=1}^n$. The feature value of r is the mean intensity of small black rectangles subtracted from the mean intensity of small white rectangles, so it can be defined as

$$f(r; I_q) = \sum_{k=1}^{n} c_k m(I_{s_k}). \quad (2)$$

Originally, Viola and Jones used a binary threshold function for the classification function,

$$h(r; I_q) = \begin{cases} 1 & (I_q \text{ is face) if } pf(r; I_q) > p\theta \\ 0 & (I_q \text{ is nonface) otherwise} \end{cases},$$

where p and θ are determined from machine learning. We replace it with a sigmoid function that is differentiable.

$$h(r; I_q) = \frac{1}{1 + e^{p(f(r;I_q)-\theta)}} \quad (3)$$

2.3 Boosting

Boosting is a learning algorithm that integrates multiple classifiers to yield a stronger classifier. The obtained classifier is called a $strong\ classifier$, and the classifiers used for components of the strong classifier are called $weak\ classifiers$. In Viola and Jones' method, a weak classifier is a single rectangle feature.

The algorithm is as follows.

- Enter training set $\{x_i, t_i\}_{i=1}^N$
  ($t_i = 1 \Leftrightarrow x_i$ is face. $t_i = 0 \Leftrightarrow x_i$ is nonface).

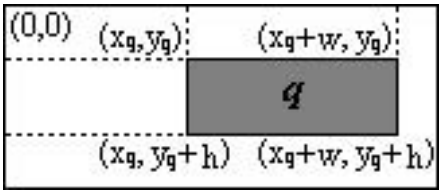

Figure 1. Integral image.

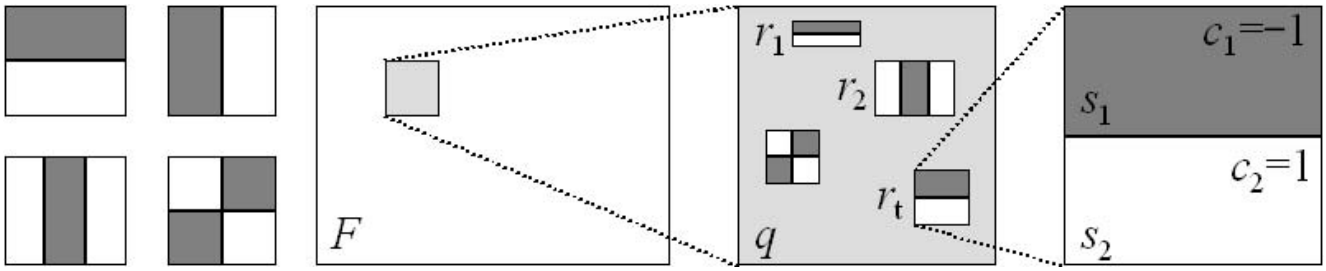

Figure 2. Rectangle Features.

- Initialize the weights of all images: if $t_i = 1$ then $w_{1,i} = \frac{1}{2p}$, else $w_{1,i} = \frac{1}{2q}$
  (p, q is number of faces, nonfaces).
- for $t = 1, \cdots, T$
  - Normalize the weights: $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{i=1}^{N} w_{t,i}}$
  - Training rectangle features: for each rectangle feature $r^{(j)}$, determine parameters p and θ that yield minimum weighted error $\epsilon^{(j)} = \sum_i^N w_i |h(r^{(j)}; x_i) - y_i|$.
  - Adopt rectangle feature $r^{(j)}$ having minimum $\epsilon^{(j)}$ in $h(r_t; x)$, which is the t-th weak classifier; set $\epsilon_{min} = \min\{\epsilon^{(j)}\}$.
  - Assign weights: $\alpha_t = \log(\frac{1-\epsilon_{min}}{\epsilon_{min}})$ for $h(r_t; x)$.
  - Update weights:
    $w_{t+1,i} = w_{t,i}(\frac{\epsilon_{min}}{1-\epsilon_{min}})^{1-|h(r_t;x_i)-t_i|}$.
- Construct strong classifier:

$$H(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^{T} \alpha_t h(r_t; x) \geq \Theta \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}, \quad (4)$$

  where Θ is a threshold determined manually.

In the training of weak classifiers, an image having a large $w_{t,i}$ is treated as more important training data that must not be misclassified as much as possible. In the final process of each iteration, $w_{t,i}$ will be updated according to the following rule: $w_{t+1,i}$ increases more than $w_{t,i}$ when $\{x_i, t_i\}$ is misclassified by the weak classifier, which is selected in the current iteration. Because of this rule, the weight of the data that has been difficult to classify until now becomes large after the next iteration. Therefore, at the later stages of boosting, a weak classifier that correctly classifies difficult training data is easy to select. Finally, a strong classifier will classify all training data correctly.

The strong classifier performs "weighted majority decision" by the weak classifiers. Weight α_t implies the "voting power" of the weak classifier $h(r_t; x)$.

2.4 Generalization Performance

In the original Viola and Jones' method, there are two optimization phases in the boosting algorithm:

1: determine the best parameters for each rectangle feature

2: select the best one in the pool of optimized rectangle features

However, this two-phases optimization may cause overfitting of the strong classifier to the training data. From an experiment, we found that generalization performance of the strong classifier may be improved by reducing the two-phases optimization to single-phase. In our method, we omit the optimization of the parameters of rectangle features. Thus, the weighted error of each rectangle feature is calculated under the parameters assigned randomly, and the rectangle feature having the lowest weighted error is selected.

We evaluated the classification performance of both learning algorithms, using 725 face images and 2,200 nonface images. These face/nonface images are divided into three sets (therefore each set includes about 240 face and 730 nonface images). One of the set is used for training, and remaining two sets are used for testing. We repeated this experiment three times while changing the set used for training, and plot the average classification rate in Fig. 3. The classification rate of the two-phase optimization converges faster than that of the single-phase optimization for the training set. This is a reasonable result because the parameters of each weak classifier are modified to well classify the training data. However, the two-phases optimization is not always superior for the test set. When the number of weak classifiers is small, the classification rate of the two-phases optimization is surely higher than the single-phase's one. But this relationship reverses when the weak classifiers is more than about 50, and finally the single-phase optimization converges more higher classification rate than two-phases optimization. This means that the single-phase optimization may be better in generalization performance. We consider this is a result of that the diversity of chosen weak classifiers became more wide by the randomness that added to the parameters of rectangle features.

Which algorithm should be used will depend on the required size of the strong classifier. If the strong classifier consists of a few weak classifiers (such as the early part of the cascade structure [11, 12]), the two-phases optimization should be used to strictly optimize each weak classifier. On the other hand, when more weak classifiers are used like our tracking method, the single-phase optimization may improve generalization performance of the strong classifier.

3 Tracking by Maximizing Score Function

In SVT, target tracking is performed by maximizing the target's score function. In the case of face tracking, the score function expresses the degree of similarity between a target image and faces. We expect that a local region where a face appears has the maximum score of all local regions throughout the image. If a face region moves anywhere in the image, then a peak score with this face should move in an image plane synchronously. Therefore, if a reliable score function is given, then the target tracking problem can be represented as peak tracking, which means searching for the region where the score is maximum. When the score function is differentiable, its derivative can be used to help the search. If the target moves to unknown region q from q_0, then q is in the direction of the gradient of the score function at q_0.

3.1 Score Function and Target Function

Our score function $E(I_q)$ is obtained from Eqs. 3 and 4, as

$$E(I_q) = \sum_{t=1}^{T} \alpha_t h(r_t; I_q) = \sum_{t=1}^{T} \frac{\alpha_t}{1 + e^{p_t(f(r_t;I_q)-\theta_t)}}. \quad (5)$$

This implies the "total voting power" of the majority decision by $h(r_t; I_q)$.

Let us consider images I' and I, which are consecutive frames in a video sequence including human faces and other objects. Suppose the face image that is located in local region q in frame I' is moving to unknown region q_f in current frame I. The old position q can be used as a clue to specify q_f. Let I_q be a subimage of current frame I at q. Now, we consider how $E(I_q)$ changes in a neighboring region of q. Let $q_{(u,v)}$ be a region that is (u, v) distant from q. To reduce computation, $I_{q_{(u,v)}}$ is approximated by first-order Taylor expansion at q as follows:

$$I_{q_{(u,v)}} \simeq I_q + uI_{qx} + vI_{qy}, \quad (6)$$

where $I_{qx} = \frac{\partial I_q}{\partial x}$, and $I_{qy} = \frac{\partial I_q}{\partial y}$. From this, Eqs. 1 and 2 are approximated as follows:

$$m(I_{s_k}) \simeq m(I_{s_k}) + um_x(I_{s_k}) + vm_y(I_{s_k})$$

$$f(r_t; I_q) \simeq \sum_{k=1}^{n} c_k \{m(I_{s_k}) + um_x(I_{s_k}) + vm_y(I_{s_k})\} \quad (7)$$

where $m_x = \frac{\partial m}{\partial x}$, and $m_y = \frac{\partial m}{\partial y}$ [1] We reset the right-hand side of Eq. 7 to $f^*(r_t; u, v)$. Replacing f with f^* in Eq. 5, we obtain the approximation of $E(I_q)$, which is the target function to be maximized. It is defined using movement (u, v) as follows:

$$E^*(u, v) = \sum_{t=1}^{T} \frac{\alpha_t}{1 + e^{p_t(f^*(r_t;u,v)-\theta_t)}}. \quad (8)$$

If the assumption that E^* as well as E becomes maximum at q_f is correct, the target will be in $q_{(u_m,v_m)}$ where (u_m, v_m) is a maximizer of $E^*(u, v)$. Therefore, the task of tracking q_f is represented by maximizing $E^*(u, v)$.

[1] Note that m_x and m_y as well as m can be calculated rapidly from Integral Image. For example, $m_x(I_{s_k})$ is represented intuitively as $mean(\{i(x_k + w, y)\}_{y=y_k}^{y_k+h}) - mean(\{i(x_k, y)\}_{y=y_k}^{y_k+h})$, where $mean(X)$ returns the mean value of elements of X, and $\{(x_k + w, y)\}_{y=y_k}^{y_k+h}$, $\{(x_k, y)\}_{y=y_k}^{y_k+h}$ means right/left vertical edges of rectangle s_k. Because edges can be considered as rectangular regions whose width or height is unit length, we can rapidly calculate m_x and m_y applying Integral Image to the edge regions.

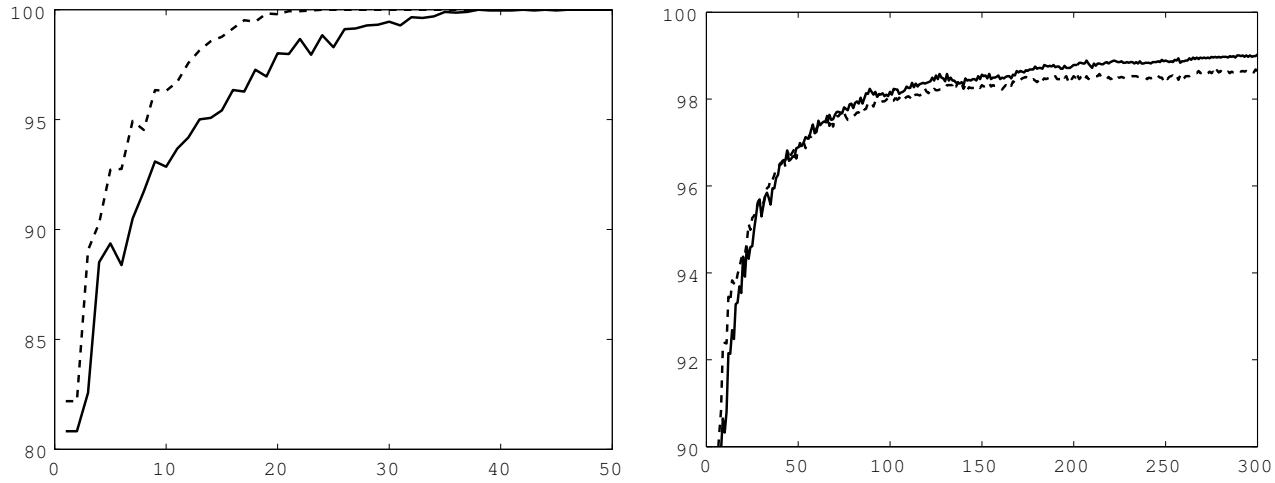

Figure 3. The classification rate curves for the training set (left) and the test set (right). The abscissa is number of weak classifiers (equal iterations number of boosting), and the ordinate is correct classification rate (%). Solid line and broken line shows the result of our single-phase optimization and the original two-phases optimization, respectively. For the test set, the two-phases optimization has higher classification rate than the single-phase optimization when the number of weak classifiers is small. However, when the number of weak classifiers is over about 50, the classification rate of the single-phase optimization is higher than that of the two-phases optimization.

For example, the steepest descent method is applied as follows:

$$\begin{aligned}\frac{\partial E^*}{\partial u} &= -\sum_{t=1}^{T}\alpha_t \frac{e^{p_t(f^*(r_t;u,v)-\theta_t)}}{(1+e^{p_t(f^*(r_t;u,v)-\theta_t)})^2} p_t f^*_{tu} \\ \frac{\partial E^*}{\partial v} &= -\sum_{t=1}^{T}\alpha_t \frac{e^{p_t(f^*(r_t;u,v)-\theta_t)}}{(1+e^{p_t(f^*(r_t;u,v)-\theta_t)})^2} p_t f^*_{tv} \\ (u,v) &\leftarrow (u+\eta\frac{\partial E^*}{\partial u}, v+\eta\frac{\partial E^*}{\partial v}),\end{aligned}$$

where $f^*_{tu} = \frac{\partial f^*(r_t;u,v)}{\partial u} = \sum_{k=1}^{n} c_k m_x(I_{s_k})$, $f^*_{tv} = \frac{\partial f^*(r_t;u,v)}{\partial v} = \sum_{k=1}^{n} c_k m_y(I_{s_k})$, and η is a step length that is selected manually.

3.2 Multilevel Blurred Images to Avoid Local Maxima

When using a gradient method, multimodality of the target function causes convergence to local maxima. To avoid this, we used multilevel blurred images, like Avidan's pyramid images [13]. In a blurred image, multimodality of the target function is toned down, but the precise position of the true maximum becomes slightly unclear (Fig. 4). Therefore, we start maximization at the most strongly blurred image using a start position with the target position in the previous frame. The position that the maximization achieved at is used as the start position of the maximization at the next level image. In this way, the algorithm approaches the true maximum gradually while avoiding convergence to local maxima. Finally, the maximized position in the original image is considered as the new position of the target in current frame.

The whole tracking algorithm is as follows.

1. Set $m \leftarrow 0$.
2. Set $m \leftarrow m+1$; Enter I^m which is the mth frame of the video sequence.
3. Run face detector on I^m; let q^m be a detected face position; if no face detected, go to 2.
4. (Tracking Mode)
   (a) Set $m \leftarrow m+1$; enter I^m which is the mth frame of the video sequence.
   (b) Create an L-level blurred image set $\{I^m_l\}_{l=0}^{L}$ from I^m(I^m_0: original image, I^m_L: most strongly blurred image).
   (c) Set $q^m_L \leftarrow q^{m-1}$.
   (d) For $l = L, \cdots, 1$
      i. Maximize Eq. 8 in blurred image I^m_l by gradient method, using start position q^m_l; let q^m_{l-1} be the position that the maximization achieved at.

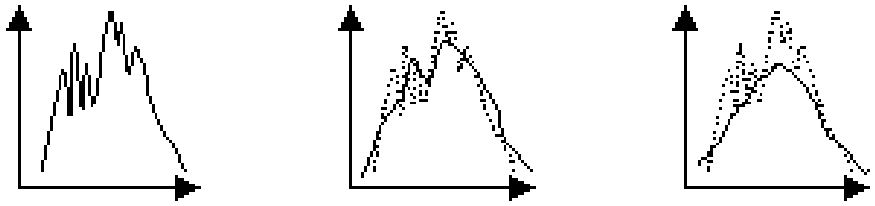

Figure 4. A model of Toning down multimodality of the score function. These graphs shows one-dimensional cross figure of the score function (left: original image, center: one-level blurred image, right: two-level blurred image). The abscissa is x axis of an image plane, and the ordinate is the score E of a local image locating in x (y axis is fixed in central position of a face existing in the whole image plane).

(e) Maximize Eq. 8 in original image I_0^m by gradient method, using start position q_0^m;let q^m be the position that the maximization achieved at.

(f) If target score at q^m in I_0^m is lower than θ_{track}, which is predefined, consider that the tracking failed and go to 2, otherwise, consider that tracking succeeded, and go to (a).

4 Experiments

The face detector used in the experiments is constructed by 200 iterations of boosting using 725 face images and 2,200 nonface images. The video sequence to be tracked consists of 1,286 frames, which are 320×240 pixels each. Each frame includes at most one person. A person moves around in a room that contains book shelves and a door as a background (Fig. 5). A person almost maintains an upright-frontal face of the same size, but sometimes rotates, turns to the side and comes closer to the camera.

When our tracker runs on the video sequence, mistracking for an almost upright-frontal face occurs 15 times throughout 1,286 frames. The cause of some mistakes is clear; the target moved too fast. The other cases of mistracking are more problematic. In some cases, the target face only slightly moved, rotated, and/or turned to the side while maintaining a almost frontal-upright face. We consider that these misses in such easy situations are caused by aborting the maximization. To reduce, and to obtain constant computational cost in each frame, the gradient method must be aborted when its iteration reaches a predefined number. Therefore, the tracked position in each frame may differ from the actual position a little bit, even if tracking seems to be good. This slight gap in tracking is a bad influence on the next tracking. If these gaps are accumulated in the same direction, the tracked position will gradually deviate from the center of the face. Even if gaps are small and appear in random directions, it leads noisefull tracking result. In our experiment, the window showing the tracked target position perturbed in an unsightly manner in each frame.

4.1 Jumping to gradient direction and precise search in neighbor region

These problems are caused when the gradient method does not converge to the maximum. However, that may be solved by a precise search for a local region. Hence, we use maximization by a gradient method and a local search appropriately, as follows. In each blurred image, the gradient method is used in the same way so far. A role of gradient method is to refine the position roughly using less computation by jumping to the gradient direction of the score function. In the original image, the position achieved by the jumping is used as the center of the neighborhood, which is searched by the face detector with brute force. The neighborhood has a predefined size. Let $N_s(q_0^m)$ be the neighborhood that is centered at q_0^m and has a predefined size s. Then, we replace the item (e) in the tracking algorithm by (e') to obtain the new algorithm:

(e') Find the position that has the maximum score in $N_s(q_0^m)$ at image I_0^m; let q^m be the achieved position.

As a result, tracking accuracy was improved (Tab. 1). Mistracking in the easy situations almost did not occur when using the combination of the gradient method and the neighborhood search. Furthermore, perturbations of the tracked position stopped almost entirely.

Fig. 6 shows a part of tracking result of NS algorithm and the combined algorithm. The combined algorithm often well worked in the point where NS algorithm failed. This implies that even if the maximum of the score function exist out of the neighborhood, our combined algorithm can run up the gradient and may reach near the top of the score function.

In this experiment, computation time of the static detection for single target image (320×240 pixels) is 0.1 to 0.2 fps per one detection size when using 200 rectangle

Figure 5. Part of video sequence and its two levels of blurred images. Left: original image, center: one-level blurred image, right: two-level blurred image.

Table 1. Tracking result using SDM (Steepest Descent Method), NS (Neighborhood Search) and its combination. 'window' means a detection window, and $p \times q$ represents the size of neighborhood as a set of windows. The combined algorithm greatly reduced the number of mistracking in comparison with both SDM only and NS only.

	SDM only (no blur)	SDM only (blur: two levels)	NS only (5×5 windows)	SDM+NS (blur: two levels) (5×5 windows)
number of misses	over 50	15	over 40	3

features. At the tracking mode, the combined algorithm worked in about 40 fps per one face on a Pentium4 2GHz machine. Probably, the rectangle-feature-based detector can calculate own classification function faster than the pixel-wise SVM, therefore it is expected that our tracker is faster than SVT. This is a benefit of that the differential of the weak classification function is fully calculated by the integral image.

5 Conclusion

We proposed a tracking algorithm that integrates a rectangle-feature-based boosted classifier and SVT. In our algorithm, tracking is performed by maximizing the score function. The algorithm was tested on an actual video sequence. When maximizing the score function, tracking accuracy is much better than that using the gradient method and local precise search together or than only using the gradient method. In future work, we will extend this algorithm to track a 2D affine-transformed object.

References

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," In Proc. Int. Conf. on Computer Vision, pp.259-268, 1987.

[2] A. Blake and M. Isard, *Active contours* Springer 1998.

[3] M. Bkackand A. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," In Proc. European Conf. on Computer Vision, Vol.1, pp.329-342, 1996.

[4] D. Lowe, "Robust model-based motion tracking through the integration of search and estimation," Int. J. Computer Vision, Vol.8, No.2, pp.113-122, 1992.

[5] A. Yuille and P. Hallianan, "Deformable templates," In A.Blake and A.Uille, editors, *Active Vision*, pp.20-38, MIT, 1992.

[6] M. Isard and A. Blake, "CONDENSATION – conditional density propagation for visual tracking," Int. J. Computer Vision, Vol.29, No.1, pp.5-28, 1998.

[7] J.-M. Odobez, S. Ba, and D. Gatica-Perez, "An implicit motion likelihood for tracking with particile filters," In Proc. of Brithish Machine Vision Conf., 2003.

[8] D. Comaniciu, V. Ramesh, P. Meer, "Kernel-based object tracking," IEEE Trans. on Pattern Analysis Machine Intell., Vol. 25, No. 5, 564-575, 2003.

[9] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: An application to face detection," In

Figure 6. A sub sequence of tracking result. Sequence (a) and (b) shows the tracking result of NS algorithm and the combined algorithm, respectively. Between the frame #1016 and #1017, NS algorithm begins to lose the center of the target face, because target's transition is larger than the predefined size of the neighborhood. A gap of this time is not revised after all, and the score becomes less than the threshold at the frame #1019. On the other hand, the combined algorithm keeps appropriate positions through this sequence.

Proc. of Conf. Computer Vision and Pattern Recognition, pp.130-136, 1997.

[10] S. Romdhani, P. Torr, B. Scholkopf, and A. Blake, "Computationally efficient face detection," In Prof. Int. Conf. on Computer Vision, Vol.2, pp.524-531, 2001.

[11] P. Viola and M. Jones, "Robust real time object detection," In IEEE ICCV Workshop on Statistical and Computational Theories of Vision, July 2001.

[12] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," In Proc. IEEE CS Conf. Computer Vision and Pattern Recognition, Dec. 2001.

[13] S. Avidan, "Support Vector Tracking", In IEEE CS Conf. Computer Vision and Pattern Recognition, Dec. 2001.

[14] S. Avidan, "Ensemble tracking" In IEEE CS Conf. Computer Vision and Pattern Recognition, Jun. 2005.

[15] F. C. Crow, "Summed-Area Tables for Texture Mapping", In Proc. of the 11th annual conference on Computer graphics and interactive techniques, 18(3):207-212, 1984.