VizWear-Active: Distributed Monte Carlo Face Tracking for Wearable Active Cameras

Takekazu Kato Takeshi Kurata Katsuhiko Sakaue
Intelligent Systems Institute,
National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1 Umezono, Tsukuba, Ibaraki, 305-8568 JAPAN
t.kato@aist.go.jp

Abstract

In this paper, we discuss a Distributed Monte Carlo (DMC) tracking method which achieves real-time and accurate face tracking for wearable active vision systems. The DMC is an extension of sequential Monte Carlo approaches to a client-server distributed architecture. The client feeds back the results of tracking for use in the control of the wearable active camera with minimal delay, and the server accurately tracks the faces at the same time. In our method, the client is able to complete the tracking processes even when it is unable to communicate with the server. Furthermore, more accurate results are obtained when it is able to communicate with the server. We have implemented and demonstrated this method on the VizWear-Active system which includes a wearable active camera.

1. Introduction

Wearable systems have the potential to assist wearers by understanding their context. Computer vision is particularly useful as a means for understanding visual context in real-world environments. We are researching wearable systems, interfaces and applications using computer vision [1, 5].

Information about meeting people is contextual information that is particular interest to wearers. Person recognition by a wearable system is applicable as part of augmented-memory [2] and security applications. A key issue in recognition is the location of subjects person and the acquisition of their features. However, these tasks are made more difficult for a wearable system by the frequent changes in environment which are a product of separate movements of the wearer and the subject.

We are developing the *VizWear-Active* [4], which is a wearable assistant using a wearable active camera [6]. *VizWear-Active* actively obtains information on the subjects in real-time while coping with the movements of the wearer and subject. This paper proposes a Distributed Monte Carlo (*DMC*) tracking method which is an extension of the Con-Densation algorithm [3] to distributed architectures. In this method, faces are tracked by processes distributed between the wearable client and the vision server. The *DMC* tracking method achieves real-time face tracking with the control of the active camera.

2. *VizWear-Active*

Many current wearable systems include wearable cameras that are fixed to the wearer's body or head. A head-mounted camera naturally observes the subjects that the wearer is looking at. This is because, for the most part, it shares the wearer's field of view. However, there are still problems with thus mounting; the input images become unstable when the wearer moves, and the camera often loses sight of subjects when the wearer is not looking at them. In particular, it is difficult to maintain a constant focus on another person because the subject and the wearer are moving independently of each other. To solve such problems, we use a wearable active camera in *VizWear-Active*. The control of its field of view is independent of the wearer's motions.

We have constructed a prototype of the *VizWear-Active* system. It consists of a wearable client, a vision server and a wireless-LAN connection, and is shown in Figure 1. The wearable client includes a wearable PC (mobile-PentiumIII 500MHz), an active camera (SONY EVI-G20) and inertial sensors (InterSense InterTrax^2). The active camera is mounted on the wearer's shoulder, and its elevation and panning are controlled by the wearable PC. Inertial sensors are attached to the active camera and measure the posture of the camera so that the system is capable of stabilizing the input image to cope with the wearer's motion [4].

The wearable client is able to process the information ob-

1051-4651/02 $17.00 © 2002 IEEE

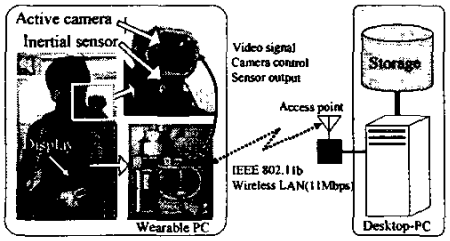


Figure 1. Prototype *VizWear-Active* system.

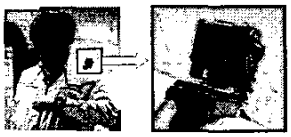


Figure 2. Wearable active camera.

tained by the camera and inertial sensors with a stand-alone. Some vision tasks are, however, too computationally intensive for existing stand-alone wearable computers. In our system, the wearable client and the vision server then cooperatively process these tasks when they can communicate each other via a wireless LAN network. In our prototype system, a desktop PC (dual Xeon 1.7GHz) is used for the vision server.

In current prototype systems, the active camera is too big to be considered wearable. We are thus constructing the new system in which the much smaller wearable active camera shown in Figure 2 is used.

We aim to develop the visual augmented-memory [2] applications for *VizWear-Active* that are based on face recognition. It assists the wearer in recalling episodes in wearer's life. In realizing such applications, the robustly detection and tracking of face images in real-time in real-world settings is important.

3. Distributed Monte Carlo Tracking

In the control of an active camera to track a person, the results of tracking must be fed back to the active camera with the minimum of delay in order to keep the person in the camera's field of view. It is, however, difficult for a stand-alone wearable client to accurately and robustly track a person in real-time, because its computational resources are often restricted by measures for reduced power consumption and wearability. It is also difficult for the vision server to immediately respond to the results of tracking because of delays in the wireless LAN network.

To solve this problem, we propose a Distributed Monte Carlo (*DMC*) tracking method which extends the ConDensation algorithm [3] to distributed architectures. In this section, we discuss the *DMC* tracking method, after briefly describing the original ConDensation algorithm.

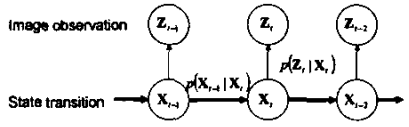


Figure 3. State transition diagram of the ConDensation algorithm.

3.1. ConDensation algorithm

In a dynamic system, the states of the subject and image observation are respectively represented by $\mathbf{X}_t$ and $\mathbf{Z}_t$. Previous states and observations are denoted by $\mathcal{X}_t = (\mathbf{X}_1, \ldots, \mathbf{X}_t)$ and $\mathcal{Z}_t = (\mathbf{Z}_1, \ldots, \mathbf{Z}_t)$, respectively. The tracking problem can thus be formulated as a problem of inferring a prior density $p(\mathbf{X}_t|\mathcal{Z}_t)$. With the application of Bayes' rule, we obtain

$$p(\mathbf{X}_t|\mathcal{Z}_t) \propto p(\mathbf{Z}_t|\mathbf{X}_t)p(\mathbf{X}_t|\mathcal{Z}_{t-1}), \quad (1)$$

where $p(\mathbf{Z}_t|\mathbf{X}_t)$ represents the likelihood of observation. When the dynamic model is given as a transition probability $p(\mathbf{X}_t|\mathbf{X}_{t-1})$, we have

$$p(\mathbf{X}_t|\mathcal{Z}_{t-1}) = \int p(\mathbf{X}_t|\mathbf{X}_{t-1})p(\mathbf{X}_{t-1}|\mathcal{Z}_{t-1})d\mathbf{X}_{t-1}. \quad (2)$$

Therefore, the prior $p(\mathbf{X}_t|\mathcal{Z}_t)$ can be estimated from the likelihood of observation $p(\mathbf{Z}_t|\mathbf{X}_t)$ for each time-step. It is, however, difficult to evaluate the likelihood of observation for every $\mathbf{X}_t$. Accordingly, these are discretely estimated by random sampling.

The density $p(\mathbf{X}_t|\mathcal{Z}_t)$ is then represented by a set of discrete random samples $\{\mathbf{s}_t^{(1)}, \ldots, \mathbf{s}_t^{(N)}\}$, each having a probability that is proportional to its weight $\{\pi_t^{(1)}, \ldots, \pi_t^{(N)}\}$. Here, the weight $\pi_t^{(j)}$ is defined from the likelihood of observation as $\pi_t^{(j)} = \frac{p(\mathbf{Z}_t|\mathbf{X}_t=\mathbf{s}_t^{(j)})}{\sum_{i=0}^{N} p(\mathbf{Z}_t|\mathbf{X}_t=\mathbf{s}_t^{(i)})}$. The sample set at time t is generated from the previous sample set at time $t-1$ as follows:

1. A sample $\mathbf{s}_{t-1}^{'(n)}$ is randomly selected from the previous sample set $\{\mathbf{s}_{t-1}^{(1)}, \ldots, \mathbf{s}_{t-1}^{(N)}\}$ according to the sample weights $\{\pi_{t-1}^{(1)}, \ldots, \pi_{t-1}^{(N)}\}$.

2. The sample $\mathbf{s}_t^{(n)}$ at time t is generated from the selected sample $\mathbf{s}_{t-1}^{'(n)}$ according to the dynamic model $p(\mathbf{X}_t|\mathbf{X}_{t-1} = \mathbf{s}_{t-1}^{'(n)})$.

3. N samples are generated by iterations of 1. and 2.

Here, the initial sample set $\{\mathbf{s}_0^{(1)}, \ldots, \mathbf{s}_0^{(N)}\}$ is generated according to a prior $p(\mathbf{X}_0)$ that is given in advance.

The expected value of a state $\mathbf{X}_t$ can be estimated from the weighted mean as

$$\mathcal{E}(\mathbf{X}_t) = \frac{\sum_{n=1}^{N} \pi_t^n \mathbf{s}_t^{(n)}}{\sum_{n=1}^{N} \pi_t^n}. \quad (3)$$

3.2. A distributed form of the ConDensation algorithm

In the *DMC* tracking method, the ConDensation algorithm is applied to the coarse and rapid tracking of the subject by the wearable client, and the results of tracking are fed back for use in the control of the active camera with the minimum delay. On the other hand, the vision server accurately estimates the state of the subject according to the results it receives from the wearable client.

The wearable client applies the original ConDensation algorithm with the lowest possible number of samples that allows of to keep tracking the subject. Let $\{\mathbf{s}_t^1, \ldots, \mathbf{s}_t^N\}$ and $\{\pi_t^1, \ldots, \pi_t^N\}$ be the sets of samples and weights for the wearable client. The vision server then accurately estimates the state of the subject from a sufficient number of samples at the same time. The vision server generates the M samples according to the probability density represented by the sample set $\{\mathbf{s}_t^1, \ldots, \mathbf{s}_t^N\}$ and their weights $\{\pi_t^1, \ldots, \pi_t^N\}$. Let $\{\mathbf{s}_t^{N+1}, \ldots, \mathbf{s}_t^{N+M}\}$ and $\{\pi_t^{N+1}, \ldots, \pi_t^{N+M}\}$ be the sets of samples and weights for the vision server.

In the wearable client, the active camera is controlled to keep the position estimated by equation (3) as the center of each input image. The client uses only those results of tracking that it has obtained in providing feedback with the minimum delay. In the vision server, the accurate state of the subject is estimated from the weighted mean of samples from both the wearable client and the server as

$$\mathcal{E}(\mathbf{X}_t) = \frac{\sum_{n=1}^{N+M} \pi_t^n \mathbf{s}_t^{(n)}}{\sum_{n=1}^{N+M} \pi_t^n}. \quad (4)$$

A more accurate result can be obtained by applying repeated processing by the vision server, as is shown in Figure 4.

With the *DMC* tracking method, the wearable client is capable of continuing to track the subject on a stand-alone basis even when it is unable to communicate with the vision server. This is because the stand-alone wearable client is also able to complete the tracking process. Furthermore, move accurate results are obtained when it is able to communicate with the vision server.

4. Implementing Distributed Monte Carlo Tracking on the *VizWear-Active* System

In this section, we give the detailed implementation of the *DMC* tracking method on the *VizWear-Active* system.

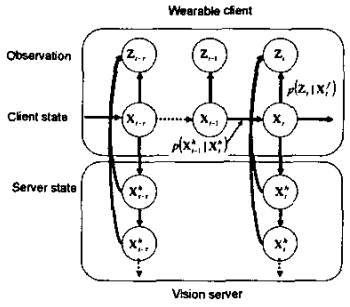


Figure 4. State transition diagram of *DMC* tracking.

4.1. Head Tracking Using an Elliptical Model

We approximate a head shape by an elliptical model. The shape is represented by five parameters in the form

$$\mathbf{X}^h = (x^h, y^h, s^h, r^h, \theta^h), \quad (5)$$

where (x^h, y^h) are the center positions, s^h is the scale, r^h is the aspect ratio, and θ^h is the angle of rotation of the elliptical model in an image. The dynamic model $p(\mathbf{X}_t^h|\mathbf{X}_{t-1}^h)$ is defined by normal distributions.

The likelihood of observation for each sample is evaluated by using features on color and shape. The color and shape observations are alternately evaluated. We generate and evaluate the samples on the wearable client in the following way:

1. For color observation, N_c samples are generated according to the previous shape samples $s_{t-1}^{s,(n)}$ and their weights $\pi_{t-1}^{s,(n)}$. Let $s_t^{c,(n)}$ be the sample for color observation.

2. The weight $\pi_t^{c,(n)}$ of the color sample $s_t^{c,(n)}$ is obtained from the likelihood of color observation.

3. For the shape observation, N_s samples are generated according to the color samples $s_t^{c,(n)}$ and their weights $\pi_t^{c,(n)}$. Let $s_t^{s,(n)}$ be a sample generated for shape observation.

4. The weight $\pi_t^{s,(n)}$ of the shape sample $s_t^{s,(n)}$ is obtained by contour-edge detection.

In the vision server, the samples for color observation are generated according to the shape samples and weights by the wearable client $s_t^{c,(n)}$ and $\pi_t^{c,(n)}$, and the rest of the processes, 2, 3, and 4, are the same as on the wearable client.

The color observation may be used to eliminate samples of the background that are of different color from a standard

color that has been learned in advance. This provides robustness of tracking in the face of noise and complex backgrounds and quick tracking because the color observation is computationally cheaper than the shape observation.

The color observation is evaluated by a color histogram. Histograms of the YUV color space are used as models for the head color. The likelihood is defined as intersection between these color histograms: the standard histogram H and image histogram I^s. H is learned in advance from the heads of numerous people, and I^s is obtained from the elliptical region.

The shape observation is evaluated by the contour edges [3]. Edge detection is performed in 1-D following the normal lines of the sample shape. Let P_c be a point on the elliptical contour of the sample $\mathbf{s}_t^{(n)}$, and $\epsilon_1^c, \ldots, \epsilon_E^c$ be the distances between P_c and the edge points on the normal line through P_c. We then obtain

$$p(\mathbf{Z}_t|\mathbf{X}_t^h = \mathbf{s}_t^{(n)}) = \prod_c^C \left(\alpha + \frac{1}{\sqrt{2\pi}\sigma} \sum_e^E \exp -(\frac{\epsilon_e^{c2}}{2\sigma^2}) \right), \quad (6)$$

where α is a constant that denotes the probability of false edges due to clutter and σ is the standard deviation of the edge distances.

4.2. Tracking with Camera Motion

In the wearable client, when the camera moves with the movements of the wearer and according to the camera control, the position of the subject in the input images may greatly vary. It is difficult to represent a dynamic model as a static transition probability $p(\mathbf{X}_t^h|\mathbf{X}_{t-1}^h)$. The dynamic model is therefore corrected for the motion of the camera as estimated on the basis of the parameters of camera control and the inertial sensors.

Let θ_t^c, ϕ_t^c be the pan-tilt parameters of the camera's control and $\theta_t^m, \phi_t^m, \psi_t^m$ be the camera posture that is measured by inertial sensors. Changes in camera angle $\dot{\theta}_t, \dot{\phi}_t, \dot{\psi}_t$ are denoted as

$$\begin{aligned} \dot{\theta}_t &= (\theta_t^c - \theta_{t-1}^c) + (\theta_t^m - \theta_{t-1}^m), \\ \dot{\phi}_t &= (\phi_t^c - \phi_{t-1}^c) + (\phi_t^m - \phi_{t-1}^m), \\ \dot{\psi}_t &= \psi_t^m - \psi_{t-1}^m. \end{aligned} \quad (7)$$

The center position and rotation angle are corrected by using these angles. Here, the other parameters, the scale and aspect ratio, are ignored. This is because they are only slightly affected by changes in camera angle. The corrected parameters $(x_t'^h, y_t'^h), \theta_t'^h$ are denoted by

$$\begin{aligned} x_t'^h &= f \tan(\arctan(\frac{x_t^h}{f}) + \dot{\theta}_t), \\ y_t'^h &= f \tan(\arctan(\frac{y_t^h}{f}) + \dot{\theta}_t), \\ \theta_t'^h &= \theta_t^h + \dot{\psi}_t, \end{aligned} \quad (8)$$

where f is the focal length. The dynamic model is corrected by these parameters.

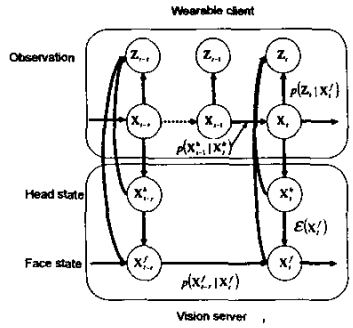


Figure 5. State transition diagram of face tracking.

4.3. Face Tracking

In the vision server, a more accurate position and shape of the face is estimated on the basis of the head state obtained by the ConDensation algorithm for the face recognition. This is shown in Figure 5.

The shape of the face depends on the shape of the head and the direction of the face in relation to the camera. Here, the facial region is defined by a rectangle, and the states of the face are defined by the following five parameters:

$$\mathbf{X}_t^f = (\phi_t^f, s_t^f, x_t^f, y_t^f, \theta_t^f), \quad (9)$$

where ϕ_t^f is the direction of the face in relation to the camera, s_t^f is the relative scale of the face, (x_t^f, y_t^f) is the center position of the face region and θ_t^f is the angle of rotation.

Let $\mathcal{E}(\mathbf{X}_t^h) = (\overline{x}_t^h, \overline{y}_t^h, \overline{s}^h, \overline{r}_t^h, \overline{\theta}_t^h)$ be the result of tracking at time t estimated by equation (4). The parameters for the face region, width W_t^f, height H_t^f, center position (X_t^f, Y_t^f), and angle of rotation Θ_t^f, are respectively denoted by

$$\begin{aligned} &W_t^f = \overline{s}_t^h s_t^f, \quad H_t^f = \overline{s}_t^h \overline{r}_t^h \cos\phi_t^f, \quad \Theta_t^f = \overline{\theta}_t^h + \theta_t^f, \\ &X_t^f = \overline{x}_t^h + \overline{s}_t^h (x_t'^f \cos\Theta_t^f - y_t^f \sin\Theta^f), \\ &Y_t^f = \overline{y}_t^h + \overline{s}_t^h (x_t'^f \sin\Theta_t^f + y_t^f \cos\Theta^f), \\ &x_t'^f = x_t^f + \sin\phi_t^f. \end{aligned} \quad (10)$$

The likelihood of observation of the face region is estimated by using the Eigenface method with normalization of the shape, brightness, and contrast [7, 8]. Here, let $\mathrm{DFFS}(s_t^{f,(n)})$ be the distance between the eigenface and the face image that is extracted from a sample $s_t^{f,(n)}$. The likelihood of observation is defined as

$$p(\mathbf{Z}_t|\mathbf{X}_t^f = s_t^{f,(n)}) = \frac{1}{\sqrt{2\pi}\sigma_f} \exp\left(-\frac{\mathrm{DFFS}(s_t^{f,(n)})}{2\sigma_f^2} \right), \quad (11)$$

where σ_f is the standard deviation of DFFS. Finally, the results of face tracking are evaluated by using the weighted mean of $s_t^{f,(n)}$ in the same way as in equation (3).

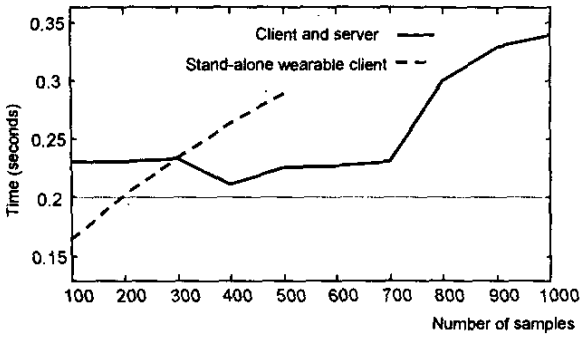
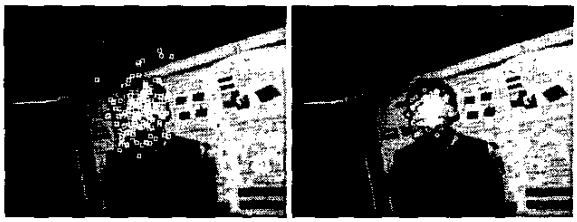


Figure 6. Comparison of processing times.



Wearable client Vision server

Figure 7. Distribution map of samples for shape observation.

5. Experimental Results

This section shows experimental results for the proposed method that is implemented on the prototype *VizWear-Active* system, the details of which were given in Section 2.

5.1. Performance in Distributed Monte Carlo Tracking

The processing times for varied numbers of samples about both the wearable client and the vision server were compared. Figure 6 shows the processing time per frame. The broken line indicates the processing time when the number of samples for shape observation on the wearable client was varied. The number of samples for color observation is here fixed to 300. This result shows that processing time increases linearly with the number of samples. Experientially, we know that tracking with camera control is possible as long as the processing is completed within 0.2 seconds per frame. Therefore, the number of samples for shape observation was set to 200 on the wearable client in the following experiments.

The solid line indicates the processing time when the number of samples for shape observation on the vision server was varied. The number of samples on the wearable client is here fixed to 200. This results shows that the processing time hardly changes until the server has 700 samples because the processing by the wearable client takes 0.2 seconds, and the delay of the wireless LAN must also be taken into account. Therefore, the number of samples for shape observation on the vision server was set to 700 in the following experiments.

We compared the performance of our method with the case of that the wearable client only sent the input image and whole of tracking process was applied on vision server alone. Then, about 0.4 seconds passed between capture of the image and reception of the result on the wearable client. This is caused by the delay of the wireless LAN. This delay will become still larger as a result of roaming and noise. In such a case, our method is capable of controlling a camera to track a person without being affected by the conditions of the wireless LAN.

5.2. Distribution of samples

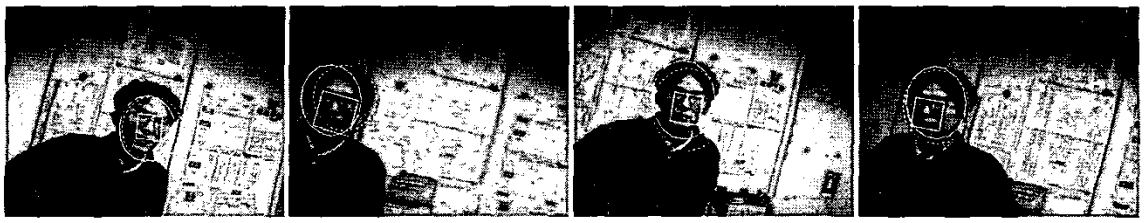
Figure 7 shows an example of a map of the distribution of samples for shape observation on the wearable client and the vision server. Each point indicates the center position of the ellipse in each sample. The samples on the wearable client were distributed sparsely and over a wide range. On the other hand, the samples were distributed over a narrow range on the vision server. This is because of the difference between the respective dynamic models. The wearable client thus rapidly and coarsely tracks the person, while the vision server accurately estimates the posture of the person.

5.3. Results of Tracking

Figure 8 shows the results of head and face tracking. The broken ellipses indicate the tracked head regions from the wearable client and the solid ellipses indicate the tracked head regions by the proposed method. Figure 9 shows the errors in center position between the tracked position and manually obtained correct position. The broken line indicates the errors in the results obtained by the stand-alone wearable client and the solid line indicates the errors in the results obtained by the proposed method. We can see that there were large errors in the results from the wearable client, but these errors were reduced in the results obtained by the proposed method. The rectangles in Figure 8 indicate the tracked face region. Although there were some errors, stable face tracking was achieved in the most of frames.

$t = 5$ $t = 13$ $t = 16$ $t = 19$ (seconds)

(a)



$t = 14$ $t = 17$ $t = 21$ $t = 24$ (seconds)

(b)

Figure 8. Results of tracking.

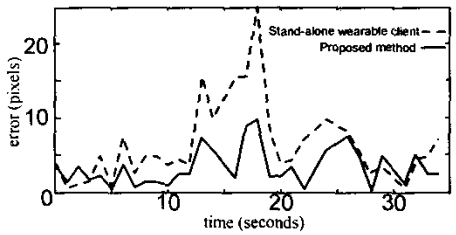


Figure 9. Errors in head tracking.

6. Discussions

In this paper, we have proposed a Distributed Monte Carlo (*DMC*) tracking method which tracks faces by processes that are distributed between the wearable client and the vision server. The wearable client coarsely and rapidly tracks the head, and its results are fed back to control the active camera. The vision server more accurately estimates the face region according to the results of the wearable client. The advantage of our method over simple parallel processing is that the tracking process completable by the stand-alone wearable client. This makes the wearable client capable of continuing to track without missing the subject even when it is unable to communicate with the vision server.

In the experiments, we confirmed that stable head tracking can be achieved in real-time with control of the active camera. However, the face tracking often failed when the wearer or subject was moving quickly. In future work, we will attempt to identify the reasons for and correct these errors by using more accurate post processing, such as facial feature analysis.

Acknowledgments

This work is supported by Special Coordination Funds for Promoting Science and Technology of MEXT of the Japanese Government.

References

[1] http://www.is.aist.go.jp/vizwear/.

[2] J. Farringdon and V. Oni. Visual augmented memory. In *4th International Symposium on Wearable Computers (ISWC2000)*, pages 167–168, 2000.

[3] M. Isard and A. Blake. Condensation – condentional density propagation for visual tracking. *International Journal of Conputer Vision*, 29(1):5–28, 1998.

[4] T. Kato, T. Kurata, and K. Sakaue. Face registration using wearable active vison systems for augmented memory. In *DICTA2002*, pages 252–257, Jan 2002.

[5] T. Kurata, T. Okuma, M. Kourogi, T. Kato, and K. Sakaue. VizWear: Toward human-centered interaction through wearable vision and visualization. In *PCM2001*, pages 40–47, 2001.

[6] W. Mayol, B. Tordoff, and D. Murray. Wearable visual robots. In *4th International Symposium on Wearable Computers (ISWC2000)*, pages 95–102, 2000.

[7] Y. Sugiyama and Y. Ariki. Facial region tracking and recognition by subspace method. In *VSSM'96*, pages 225–230, Sept. 1996.

[8] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, vol.3(no.1):71–86, Jan. 1991.