# Avatar's Expression Control through Face Tracking

***AeKyung Yang, Yong Woon Park, GyeYoung Kim, HyungIl Choi,**
Soongsil University, Seoul, 500-712 Korea
**akyang@vision.ssu.ac.kr,woon5901@hanafos.com*
*gykim@computing.ssu.ac.kr,hic@computing.ssu.ac.kr*

## Abstract

We propose the method that avatar mimics the human facial expression and face behavior in the virtual environment. The proposed method is to extract the face and its features and transfer the rendering information to the avatar sequentially. We use deformable model to solve the problem. The face and its features are detected and tracked even when human face is hidden partially due to rotation. The deformable model is graph-based and it has two functions for face and facial features detecting and tracking by the energy minimization and the Kalman filter. The result of detecting and tracking is transferred to the face mesh model in the form of a feature point set. The feature point set is composed of minimum number of point for facial expression, so we can reduce display time.

**Key words**: Deformable Face Model, Kalman Filter, Face detecting, Tracking, Avatar

## 1. Introduction

The modeling or simulation about virtual environment and object becomes more realistic when we mix techniques of image processing and computer graphics. So the mixing techniques of the image processing and the computer graphics have many applications like as HCI(Human Computer Interface), virtual simulation, computer game, digital broadcasting etc. Face detecting and tracking technique is commonly used in these applications for searching human users.

There are many proposed method to detect and track a human face. Some of them are methods using template matching[1][2], eigenvector[3], optical flow[4][5] etc. Each method has merits and demerits. Template matching method depends on the template's attributes. The static template can cause many errors due to facial movements. An eigenvector has become a famous method recently for face detection and recognition, but its performance depends on segmentation of eigenspace and it requires large learning data. That is to say if the eigenspace does not include many cases of facial poses and features, the error is increased. And optical flow method takes a long time.

In this paper, we use graph-based deformable template matching to handle the case of rotational movements of a face. We fixed the number of nodes and arcs of a deformable template to keep the information of a hidden part caused by a face rotation. By minimizing the necessary number of nodes and arcs, we tried to reduce the overall time cost. Also the template becomes to have the minimum information and this information is sent to an avatar. From now on, we will call this model "deformable face model".

Our deformable face model is graph-based, and it detects and tracks the face and facial features by the energy minimization and the Kalman filter. An avatar is visualized by mapping a deformable face model into a face mesh model.

So overall, our deformable face model can solve the problem of partial occlusion and the processing time can be decreased by reducing the amount of information.

This paper is composed of six sections. In section 2, we describe the deformable face model. In section 3, the face detecting and tracking by the deformable face model is explained. Section 4 shows how the avatar is visualized as the result of face tracking and section 5 shows the experimental results. Lastly in section 6, we included the conclusions and the future work briefly.

## 2. Deformable Face Model

There are many face detection methods that use template matching techniques[1][2], but most of them uses a static template. The static template can yield many errors if a user moves. As an alternative, we can consider a dynamic template, but it can result an increasing of the processing time. In addition to, most of template matching methods require that all of facial components are visible at any direction.

In this paper, we propose a method that can obtain the facial information even when a face poses to any direction. Furthermore the structure of the deformable template is very effective and it can be represented efficiently. So we can reduce computation time for detecting and tracking a face.

Our deformable face model is a graph-based deformable template and it has two functions for detecting and tracking of a human face. The number of nodes and arcs do not alter even when the facial turning. So the deformable face model can detect and track face and facial features of partially hidden face robustly.

The deformable face model is composed of 4 nodes - two eyes, mouth, face – and adjacent arcs. Each node and arc has own property as follows.

*Properties of face node = {area, orientation, compactness, center of mass}*

*Properties of eye and mouth node = {texture, boundary, area, eccentricity, center of mass}*

*Properties of arc = {length, angle}*

The property of nodes and arcs represents position, shape and relation with an adjacent node. Properties are used for energy calculation.

The deformable face model detects the facial features by energy minimization, and the energy E is the sum of a node energy, an arc energy and a dummy energy as follows.

$$E = \alpha \cdot V_e + (1-\alpha)\cdot \beta \cdot V_e^* + \gamma \cdot A_e + (1-\gamma)\cdot A_e^* + D_e$$

$\alpha$ is the weight of the node energy $V_e$ and it denotes how many parts of a node is occluded. $\beta$ depicts the relationship with an adjacent node, $\gamma$ is the weight of an arc. $V_e$ is the energy of each node, $A_e$ is the energy of each arc, and $D_e$ is the energy about dummy. The dummy is to represent a hidden node caused due to face rotation. Here, * means the estimated value.

$V_e$ is computed as the differences of property values of each node against those of the reference model.

$$V_e = \frac{1}{n}\sum_{i=1}^{n} e\left(v_i^m, v_i\right)$$

$$e\left(v_i^m, v_i\right) = \frac{1}{6}\left( \begin{array}{l} \min\limits_j\left(\dfrac{\left|MER\left(v_i^m\right) - MER_j\left(v_i\right)\right|}{\max\limits_j\left|MER\left(v_i^m\right) - MER_j\left(v_i\right)\right|}\right) + \min\limits_j\left(\dfrac{\left|Cc\left(v_i^m\right) - Cc_j\left(v_i\right)\right|}{\max\limits_j\left|Cc\left(v_i^m\right) - Cc_j\left(v_i\right)\right|}\right) \\ + \min\limits_j\left(\dfrac{\left|N\left(v_i^m\right) - N_j\left(v_i\right)\right|}{\max\limits_j\left|N\left(v_i^m\right) - N_j\left(v_i\right)\right|}\right) + \min\limits_j\left(\dfrac{\left|Ec\left(v_i^m\right) - Ec_j\left(v_i\right)\right|}{\max\limits_j\left|Ec\left(v_i^m\right) - Ec_j\left(v_i\right)\right|}\right) \\ + \min\limits_j\left(\dfrac{\left|G\left(v_i^m\right) - G_j\left(v_i\right)\right|}{\max\limits_j\left|G\left(v_i^m\right) - G_j\left(v_i\right)\right|}\right) + \min\limits_j\left(\dfrac{\left|Cp\left(v_i^m\right) - = Cp_j\left(v_i\right)\right|}{\max\limits_j\left|Cp\left(v_i^m\right) - Cp_j\left(v_i\right)\right|}\right) \end{array} \right)$$

Where *n* is the number of nodes, $v_i^m$ means *i*-th node of the reference model, $v_i$ means *i*-th node of an input. $e(v_i^m, v_i)$ is an energy of each node and $e(v_i^m, v_i)$ is sum of differences between the reference node and the input node. The each term of $e(v_i^m, v_i)$ has the corresponding property of a node.

MER means a minimum enclosing rectangle of a node, Cc means a chain code sum as the boundary information, N means the area of a node, Ec means the eccentricity, G is the gabor wavelet coefficient as the texture information, and Cp means the center of mass.

$$A_e = \frac{1}{4}\left(e\left(a_{12}^m, a_{12}\right) + e\left(a_{13}^m, a_{13}\right) + e\left(a_{14}^m, a_{14}\right) + e\left(a_{23}^m, a_{23}\right)\right)$$

$$e\left(a_{ij}^m, a_{ij}\right) = \frac{1}{2}\left(\min_k\left(\frac{\left|Dist\left(a_{ij}^m\right) - Dist_k\left(a_{ij}\right)\right|}{\max\limits_k\left|Dist\left(a_{ij}^m\right) - Dist_k\left(a_{ij}\right)\right|}\right) + \min_k\left(\frac{\left|Angle\left(a_{ij}^m\right) - Angle_k\left(a_{ij}\right)\right|}{\max\limits_k\left|Angle\left(a_{ij}^m\right) - Angle_k\left(a_{ij}\right)\right|}\right)\right)$$

Arc energy $A_e$ is also composed of the sum of property values. $e(a_{12}^m, a_{12})$ depicts the energy of arc $a_{12}$, and $a_{12}$ denotes the arc between node 1 and node 2. *Dist* denotes the euclidean distance of an arc and the *Angle* is the angle of an arc.

The dummy energy $D_e$ is the hidden node's energy. Fig.2 shows an example where a left eye is hidden due to the face rotation. Whether a node is dummy or not depends on the area of a node and the length of an adjacent arc. If the area of a node and the length of an arc are smaller than the predefined threshold, the node is regarded as dummy. If the node becomes dummy, its energy E is calculated by an adjacent node and an adjacent arc instead of its own value.

$\alpha$ is the weight of a node and it is calculated by the ratio of areas. $\beta$ is weight of an adjacent node and it is calculated considering the number of adjacent nodes and their distances. $\gamma$ is the weight of an arc, it is calculated considering the angle of two adjacent arcs. If the occlusion happens, $\gamma$ becomes near to zero. For example, the difference angle of $r_{12}$ and $r_{13}$ gets near to zero if face turns to left or right direction as Fig.4.

When the energy E is minimized, the deformable face model does converge. The properties of detected facial features become the components of the state vector of the Kalman filter.

We use the novel Kalman filter to reduce the operating time. Generally, the Kalman filter takes many computations when the gain is calculated. So we update the Kalman gain equation using the mahalanobis distance for keeping the concept of Kalman gain and reducing the operating time[6].

The state vector of the Kalman filter is composed of 4 coordinates ; minimum enclosing rectangle(MER), center of mass(CP), and area of each node(N) and differential values as follows.

$$\begin{bmatrix} MER(top) \\ MER(bottom) \\ MER(left) \\ MER(right) \\ CP(x) \\ CP(y) \\ N \\ \Delta MER(top) \\ \Delta MER(bottom) \\ \Delta MER(left) \\ \Delta MER(right) \\ \Delta CP(x) \\ \Delta CP(y) \\ \Delta N \end{bmatrix}$$

The Kalman filter plays a role that estimates the next position of a node and keeps the hidden position of the dummy. The estimated value allows wider error range than the measured data(Fig.2). If the dummy is produced, the weight of the estimated value is going up. That is, it has the effect of keeping the information of hidden nodes. Fig.3 shows the difference between the case where the hidden node is considered in Fig.3(a), and the case where the hidden node is not considered in Fig.3(b) when computing the energy E. Fig 3 (b)'s energy is three time higher than that of Fig.3(a).

Up to now, we described the deformable face model as a graph-based deformable template which does detecting face and facial features by energy E minimization and tracking by an updated Kalman filter.

## 3. Face Detecting and Tracking

In this section, we describe how to employ the deformable face model to detect and track face and facial features.

First of all, we need the reference model to calculate energy E. The reference model is generated by the 1st frame. So a user is supported to look at the camera lens at first. Using this reference model, the deformable face model can work from the 2nd frame.

To generate the reference model, we need some preprocessing. The preprocessing includes binarization, labeling, projection to xy-axis. Threshold for binarization of an input image is decided by the 2nd step decision method as follows.

$$p(n,x,y) = \begin{cases} FCA(p) & \text{if, p(n)} \in \{\text{uncertain pixels}\}\ AND\ p(x,y) \in \{hole\} \\ NFCA(p) & \text{if, p(n)} \in \{\text{uncertain pixels}\}\ AND\ p(x,y) \notin \{hole\} \end{cases}$$

Where FCA means the face component area and FCA is decided by the variance of skin color. The uncertain pixel is an ambiguous pixel whether the pixel is a part of skin region or not. The hole is a pixel that is not a part of the skin region. At the 1st step, the minimum and the maximum mean value of skin and non-skin color are calculated. At the 2nd step, the threshold is decided by *p*. *p* reduces the term of the minimum mean and the maximum mean by the variance of skin color.

Through the preprocessing, candidates of a face and the facial features can be obtained. The candidates of facial features are used to form the deformable face model, and nodes are selected by energy minimization among candidates. After this step, the deformable face model is formed physically. The property of selected nodes becomes the items of state vector of the Kalman filter.

The deformable face model works repeatedly frame by frame by the energy minimization and the Kalman filter. The Kalman filter estimates next deformable face model and gives the information about dummy. And then the deformable face model is corrected by the energy minimization.

## 4. Avatar

For the facial expression presentation, there are many methods like facial action coding system[8] where the predefined facial expression is matched to the measured facial feature. Also, there are the methods using the face mesh where the boundary of measured face is matched to the boundary of virtual face[9][10]. The former is difficult to use because the psychological definition of the facial expression is not clear to adapt. The latter can be shown the natural expression but the more natural expression requires computation time.

In this paper, we use the latter approach, but we use minimum number of points needed for the facial expression and we do not match the boundary but feature points. The points are facial features' points computed from the deformable face model(Fig.5(a))

Points from the deformable face model are matched to the face mesh model by the matching metric of the set of feature points. The set of feature points is selected by following three conditions.

*a. Chain code's curvature changes steeply or the curvature is very slow.*

*b. Texture's similarity is higher than threshold*

*c, The point is within node of the deformable face model*

These conditions are met for nodes composing the deformable face model. Fig.5(a) shows the feature point set and Fig.5(b) shows the matched result of the face mesh model against the feature point set. Fig.5(c) shows the matched result between the face mesh model and the deformable face model.

After mapping, the face mesh model takes the differential value of each node and interpolates it to the next coordinate as follows.

$$u' = u + p\Delta u$$
$$v' = v + q\Delta v$$

Where u' and v' are moved coordinates, u and v are actual coordinates, p is the constant for a *x*-coordinate, q is constant for a *y*-coordinate, Δu and Δv is the small amount for change.

## 5. Experimental Results

To evaluate the proposed approach, we did several experiments. We used Pentium-III CPU, 24bit color CCD, and rendering library is Renderware 4.0. The frame rate is 20fps and the processing rate is approximately 1fps. A user should stare at the camera lens at first to generate the reference model (Fig 1). After the 2nd frame, a user is allowed to move freely.

Fig.6 shows the tracking result in case of the occlusion. The outer rectangle represents the estimated region, the inner rectangle denotes the detected region. The branch-like line denotes the arc of the deformable face model. After the occlusion, the left eye's position is kept.

Fig.7 shows the comparison between the case where we did not consider the hidden node as in Fig.7(a), and the case where we considered the hidden node as in Fig.7(b).

Fig.8(a) shows the initial state of an avatar and the reference model. The lower three small figures of Fig.8 shows the results of binarization, projection to xy-axis and labeling.

## 6. Conclusions and Future work

We proposed the deformable face model that can be used when we detect and track a human face, and we showed the experimental results where an avatar changes its position and direction accordingly. The deformable face model has the properties of facial features as well as functions of detecting and tracking. For detecting a face, we use the energy minimization of a graph model against the reference. The energy is composed of the property values of each node. The properties represent position and shape of a face such as area, texture information, center position, eccentricity, compactness, boundary, angle. For tracking a face, we use the updated Kalman filter to reduce an operating time. The Kalman filter plays a role that keeps the information of a hidden node by occlusion. An avatar is composed of a feature point set that affects the movement of facial muscles. The feature point set plays a role that matches a deformable face model against face meshes.

In the future, we will remove the user's restriction when a reference model is made for the first time. A user will not have to stare at the camera lens to make a reference model.

## References

1. Laurenz Wiskott, Labeled Graphs and Dynamic Link Matching for Face Recognition and Scene Analysis, Verlag Harri Deutsch, 1995.

2. Karin Sobottka and Ioannis Pitas, "Segmentation and Tracking of Faces in Color Images," International Conference on Automatic Face and Gesture Recognition, October, pp.236-241,1996.

3. Alex Penntland, Baback Moghaddam, and Thad Starner, "View-based and modular eigenspaces for Face recognition," IEEE Conference on Computer Vision and Pattern Recognition, Vol.1, pp84-91, June, 1994.

4. Irfan A.Essa and Alex P.Pentlan, "Coding, Analysis, Interpretation, and Recogniion of Facial Expressions," M.I.T. Media Laboratory Perceptual Computing Section Technical Report No.325, April 1995, Submitted for Review(April 1995)IEEE Transactions on Pattern Analysis and Machine Intelligence, April, 1995.

5. Yaser Yacoob and Larry S.Davis, Recognizing human facial expression, CAR-TR-706, May, 1994.

6. Yang,Aekyug,Park,JuChul, and Choi,HyungIl, "Lane Detectiion Using a Kalman Filter", IJCA, Vol.7, No.4, pp193-199, Dec,2000.

7. Yang,AeKyung and Choi,HyungIl, "Hierarchical Framework for Facial Expression Recognition", ICMI2000, Beijing, China, pp.184-190, Oct.2000.

8. P.Ekman and W.V.Friesen. Manual for the Facial Action Coding System. Cousulting Psychologists Press, 1977

9. Irfan. Essa and A. Pentland. ``Facial Expression Recognition using a Dynamic Model and Motion Energy.", In, Proceedings of the International Conference on Computer Vision 1995, Cambridge, MA., May 1995.

10. Akikazu Takeuchi and Steven Franks, "A Rapid Face Construction Lab," SCSL-TR-92-010, Sony Computer Science Laboratory Inc, 1992.

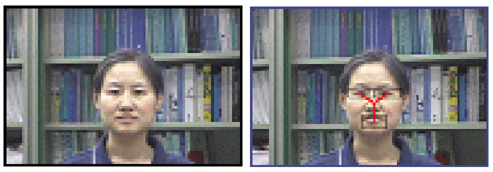
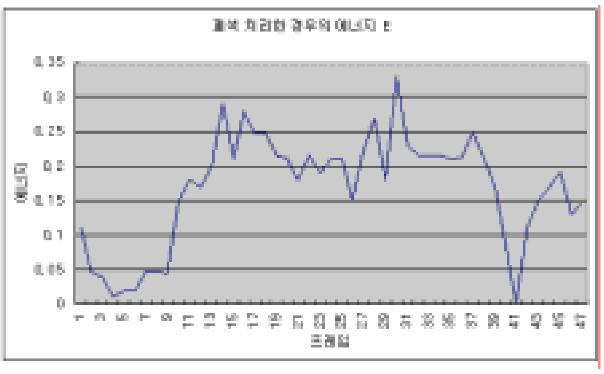
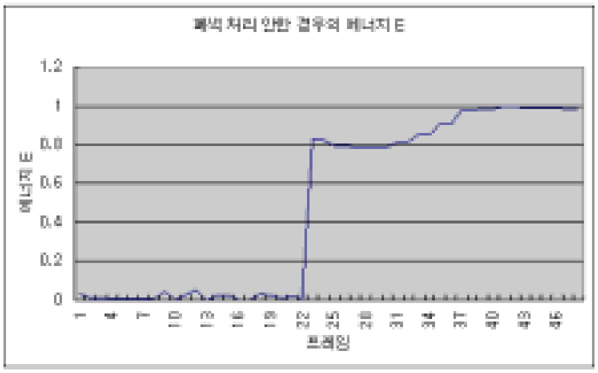

(a) (b)

Fig 1. Reference Face Model : (a) 1st image (b) Generation of face model.

Fig 2. Estimated region by Kalman Filter



(a)



(b)

Fig 3. The comparison of two energy: (a) Considered occlusion. (b) Not considered occlusion.

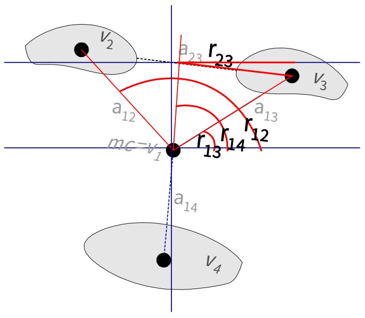
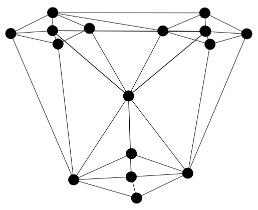
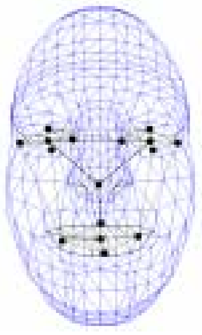


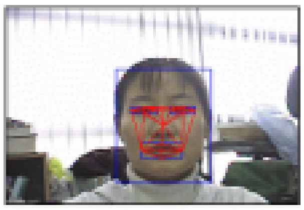
Fig 4. The angle between left eye and right eye
Angle = $r_{12}$- $r_{13}$



(a)



(b)



( c)

Fig 5. (a) Face Mesh Model composed of feature points
(b) Face Wireframe model with (a)
(c) Face Image matched with (a)

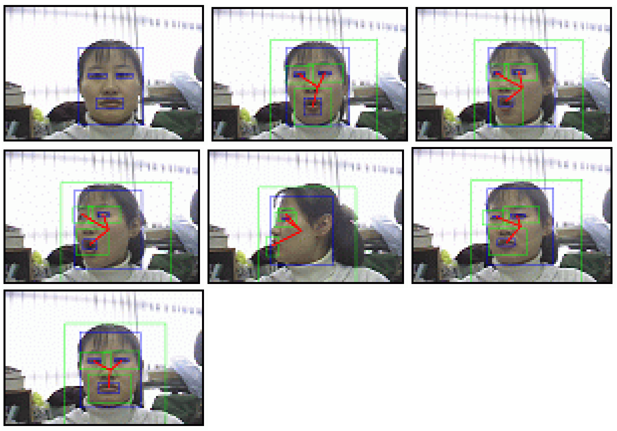
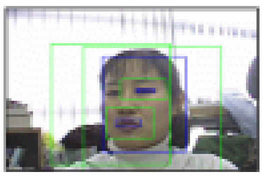
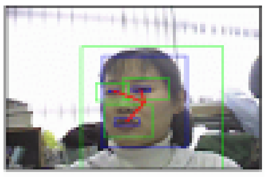


Fig 6. In case of occlusion, tracking result

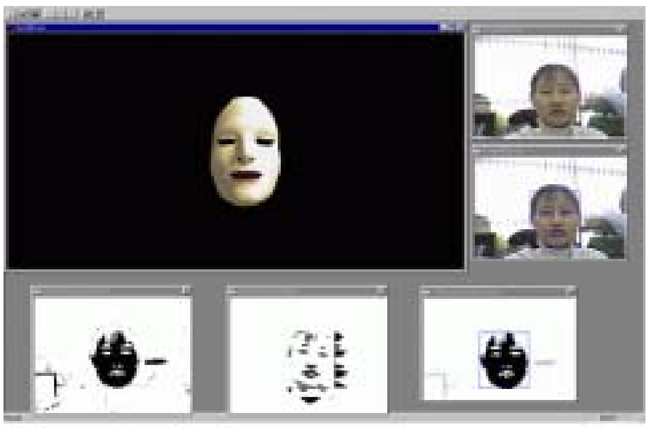


(a)

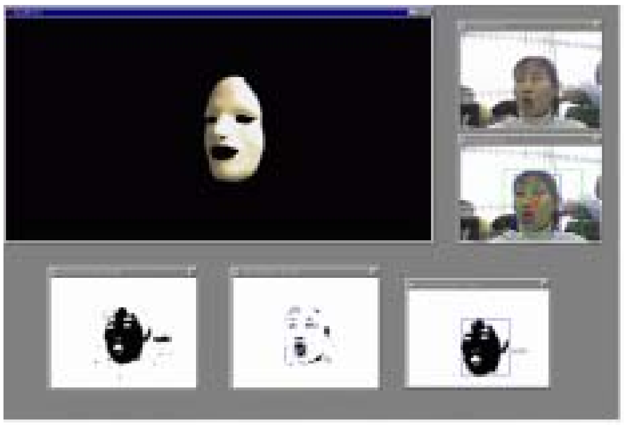


(b)

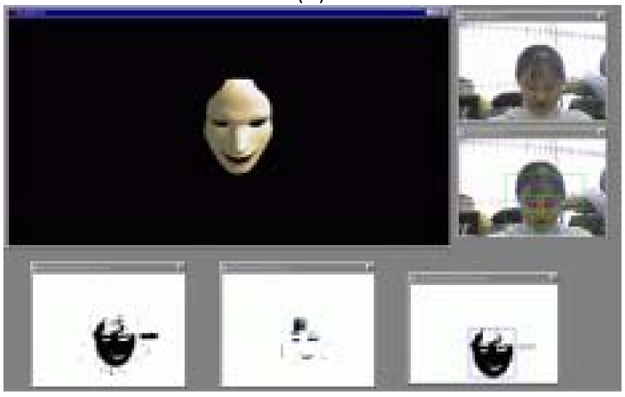
Fig 7. (a) not consider hidden (b) hidden consider hidden



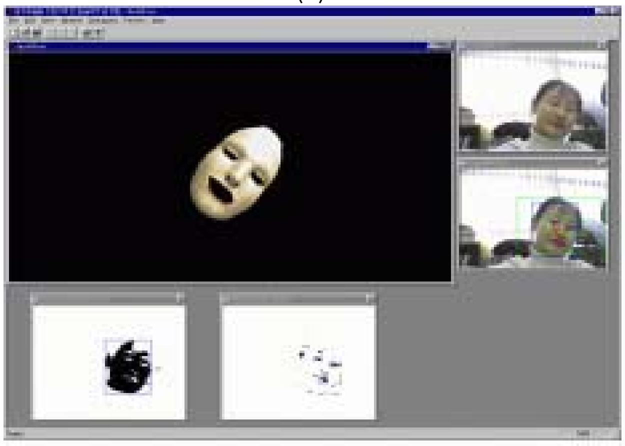
(a)



(b)



(c)



(d)

Fig 8 (a) Initial state of Avatar and reference model
(b) Surprise expression with rotated face
(c) With her head bowed
(d) Rolling face