3b2v7 IMAGE : 581– Prod.Type: typ pp.1–17(col.fig.:NIL) ED:elizabeth PAGN: vimala SCAN: sujatha

ARTICLE IN PRESS

ELSEVIER

Signal Processing: *Image Communication* ▮ (▮▮▮▮) ▮▮▮–▮▮▮

SIGNAL PROCESSING: IMAGE COMMUNICATION

www.elsevier.com/locate/image

Facial feature extraction for quick 3D face modeling

Taro Goto*, Won-Sook Lee, Nadia Magnenat-Thalmann

MIRALab, University of Geneva, 24, rue du General-Dufour, CH-1211, Geneva 4, Switzerland

Received 13 February 2001; received in revised form 20 June 2001; accepted 21 July 2001

Abstract

There are two main processes to create a 3D animatable facial model from photographs. The first is to extract features such as eyes, nose, mouth, and chin curves on the photographs. The second is to create 3D individualized facial model using extracted feature information. The final facial model is expected to have an individualized shape, photograph-realistic skin color, and animatable structures. Here, we describe our novel approach to detect features automatically using a statistical analysis for facial information. We are not only interested in the location of the features but also the shape of local features. How to create 3D models from detected features is also explained and several resulting 3D facial models are illustrated and discussed. © 2001 Published by Elsevier Science B.V.

Keywords: Face modeling; Facial feature extraction; Statistical analysis; Facial animation

1. Introduction

The 3D modeling of the human face has wide applications from video conferencing to facial surgery simulation. However, cloning a real person's face has practical limitations in the sense of how long it takes to obtain a result, how simple the equipment is and how much of a realistic shape can be obtained.

Currently, there exist two mainstream approaches to create a 3D facial model. Firstly, special equipments like 3D scanners are used to capture the 3D shape of human heads. In order to animate the scanned data inside a virtual reality environment, the shape must also be combined with an animation structure. However, this animation process requires additional information that the scanning process cannot provide directly. Therefore, this is not suitable for creating virtual 3D models by unskillful users. The second method uses images for the creation of a 3D model. Some researchers are using a stereo pair of images or video sequences [4,5] to reconstruct a face by finding pixel correspondence between successive frames or stereo-pair. Others use front and side view photographs to reconstruct a 3D facial model [1,10,12–14]. The difficulty when using two photographs is that the pixels on images do not contain any three-dimensional information. Only the X and Y positions from the front view, and Y and Z positions from the side view can be obtained. It is possible to match between characteristic points such as the eyes, nose, mouth and chin curves but 3D positional values on the forehead or cheeks where no features appear cannot be matched on

*Corresponding author.

E-mail addresses: goto@miralab.unige.ch (T. Goto), wslee@miralab.unige.ch (W.-S. Lee), thalmann@miralab.unige.ch (N. Magnenat-Thalmann).

0923-5965/01/$ - see front matter © 2001 Published by Elsevier Science B.V.
PII: S0923-5965(01)00021-2

two photographs. However, features on a face give a lot of information of the person's characteristic and the features provide the cue to build the 3D surface of the person's face. Therefore, the feature detection on photographs and how to make correspondence between two photographs have been key issues when making a reliable facial model. To detect 2D features on photographs is explored by several researchers [1,10,12–14] by manually, semi-automatically or automatically. The semi-automatic method [13] is robust, but the users must be instructed to understand what features mean and it takes some time to locate a few features manually. If we like to make a usable facial construction system, automatic process is a must. However, no automatic process has been proved to be robust for any given photographs [1,10]. The limitation comes from the variety of skin color and facial features depending on ethnic group and ages. Here we focus our effort on making more reliable and robust automatic feature detection and natural 3D feature construction from 2D features.

There are many different approaches to recognize feature positions and shapes on the face. Cootes et al. [3] use parameters of the shape variety to locate face features. This technique is more for tracking the movement of the pre-studied facial expression. Brunelli et al. [2] use a feature-based face recognition technique. They first use a template matching method by means of a normalized cross correlation coefficient. Then after the matching, gradient information of both the horizontal and vertical is used to recognize the edges. The face outline is extracted using the gradient intensity map of an elliptical shape. This method is quite useful since all human beings have the same facial structure, but can be weak at noise component, like moustaches or cuts/scratches. On the other hand, Intrator et al. [9] use a neural network model. The last method has the robustness against irregular individual features and noise. They use a generalized symmetry transform method, as the preprocessing and then warp the face to a standard location using these points. After this process, a back-propagation-type network is used to classify the features. The extraction of the feature positions (eyes and mouth) is robust enough, but the detailed shape is not considered in this method. Another neural network-based approach is done by Reinders et al. [18]. They use a neural network model to locate facial micro features. Each feature, such as an eye, can be considered a structural assembly of micro features, which are small parts of the feature that are being sought. They construct the detailed shape from the micro features. They also try to adapt the 3D face model to an image [17] using a knowledge-based feature selection mechanism. Akimoto et al. [1] try to detect feature points automatically from a knowledge-based model. They use simple filtering techniques, using the Sobel operator and constant threshold, and try to extract feature points. Ip et al. [10] propose a method for an automatic side feature detection using concave and convex curve information. This method works well for limited human races of Asian, but other races are not discussed. There is also another knowledge-based approach where the configuration of the features can help localize features and find the face position, since we have the knowledge that the features cannot appear in arbitrary arrangements. For example, given the positions of the two eyes, we know that the nose and the mouth can only lie within specific areas of the image. Leung et al. [15] use such a knowledge-based approach to develop a method called random labeled graph matching that permits detection of facial feature positions. Garcia et al. [6] recognize the global position of features using wavelet analysis and knowledge of the human face. However, they did not work on feature shape extraction.

Lee et al. [13,14] use a structured snake method to fit curves to features. The strong feature shapes are extracted in their method. However, an operator has to locate each terminal point of feature sets manually. Thus, the method is far from being automatic. In this paper, we explain how we improve on this method to make it automatic and also shorten the processing time in creating a 3D model. An operation to create a model is now only to indicate the face region of the front and side view. The facial feature positions and shapes are extracted in short time.

2. Automatic cloning outline

This section is devoted to the description of the method and the system used to realize the virtual cloning of a real person's face. We intend to produce a photo-realistic virtual human facial 3D model for real-time animation. The main idea is to modify a shape of an already well-prepared model to be individualized using feature information detected on photographs of the person. As a pre-process, we prepare generic features, a generic 3D model, generic feature region mask and an animation structure. The features definition is shown in Fig. 1. The feature definition has to fulfill several conditions as follows:

(i) The features must be recognizable on the front and side view photographs—a small number of feature points are the best candidates. Shapes of cheek or forehead are not considered as recognizable. However we add four points on cheeks because they are considered as control points in shape modification step. The positions of them are automatically assigned by other recognizable feature points.
(ii) The set of features must contain characteristics of the target person without depending on ethnic/aged/gender—a large number of feature points are the best candidates since more information from photographs guarantee better adaptation of an individual.
(iii) The features detected on the front view must be linkable to the features on the side view in certain way—a small number of feature points are the best candidates.

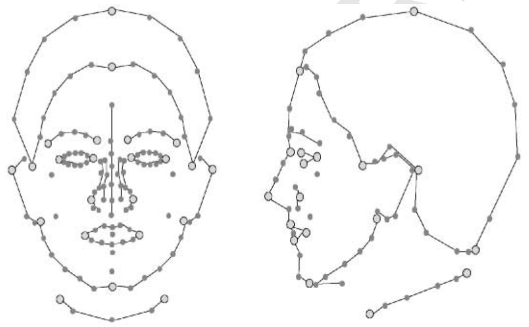


Fig. 1. Features of the frontal view and on the right side of the side view of the person.

The features are defined optimally from our experiment with various ethnic/aged/gender groups of people.

We intend to take the positional information of features out of photographs and then modify the shape of the generic 3D model to make individualized 3D model. Fig. 2 shows the flow of how the cloned face model is made from photographs. The indicated numbers on the figure show the following items:

(1) We take photographs from front and side views as input. Then an operator indicates the facial region on the images by drawing one square for each image. A generic model we use is fit in these squares.
(2) The feature locations on the face images are extracted automatically. This method is called a global matching. We use a generic model to fit to the facial image and use information of statistic feature position. This part is explained in Section 3.
(3) The feature locations are then used to detect the shapes more detail. Thus, the front face feature shape and position information is created. This is explained in Section 4.
(4) The feature shape extraction of the side face is similarly done with the front feature information.
(5) 3D features are created from two 2D features detected on two photographs. Then we deform a 3D generic model to create a 3D cloned model. After this process, we put texture generated from the photographs on the surface of the model. This is explained in Section 5.

Manual work needed are taking photographs and indicating facial regions. Then the 3D facial model of an individual is fully automatically created. Also, as the reconstructed 3D-face inherits the same structure from the generic model with all animation/functional information, it can be animated immediately with parameters extracted from the video captured face image [7,8].

3. Global matching

Before detecting feature shapes, it is necessary to know the feature positions. Feature positions like

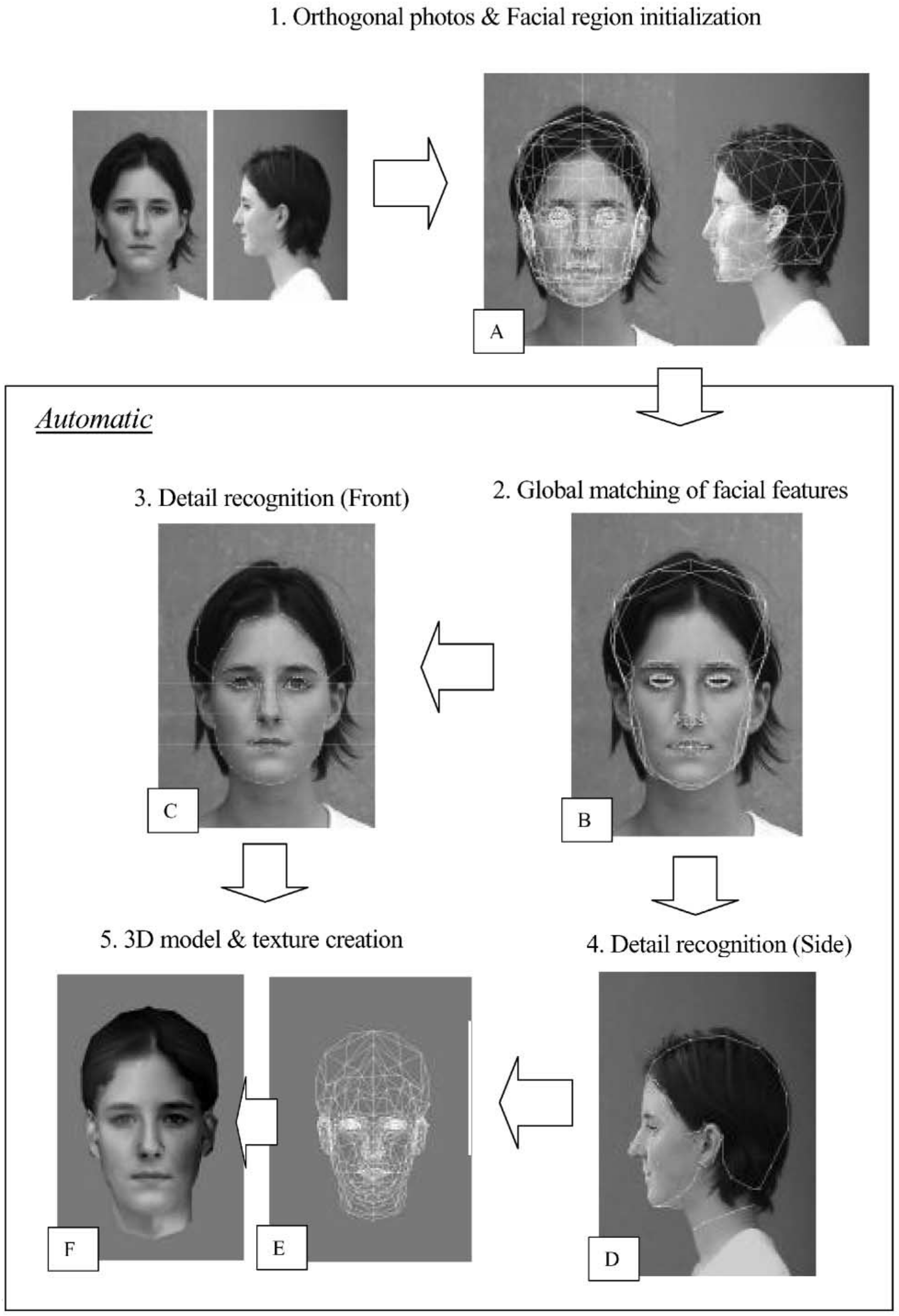

1. Orthogonal photos & Facial region initialization
A
Automatic
2. Global matching of facial features
3. Detail recognition (Front)
B
C
4. Detail recognition (Side)
5. 3D model & texture creation
D
E
F

Fig. 2. Automatic face feature detection.

an eye position or nose position are different for each individual. As we explained, we use photographs from two views. A photograph does not convey motion information, but edge information, and also sometimes color information. Thus, we have to recognize feature positions from the static image information. In general, a vision system constructs a face from this kind of image by the combination of the segment division of the features on the face and the detail region recognition after the segmentation. The first one, the segmentation of the face, is based on knowledge of the human face. The facts like 'there are two eyes around the center of the face', or 'the mouth is below the nose' helps the segment construction (a knowledge based method). The second one, the detail recognition, is based on the combined information of edge appearance, color difference (a vision based method).

To mimic this visual system, we divide our feature detection into two steps; (i) global matching to find location of features (ii) detail matching to find each feature shape by a specific method designed for each feature.

In this section, we explain the algorithm for global matching.

3.1. Statistical approach for location detection

A generic 3D facial model is used to perform this global matching in this system. An operator has to locate the generic face model on the face image by drawing a square for each photograph. This process to put the generic model on an image is done manually, because the search of the face position from the background image is not the main aspect in this research. However, the global matching after indicating the facial region is fully automatic. Fig. 2(A) shows the photograph after the generic face model is superimposed on the image. Every feature region mask that is designed by partition on the generic model as pre-process is also displayed at this stage. The face features on the photograph and the generic face model do not fit each other because feature locations of an individual face are unique for each person. This global matching helps in avoiding feature position matches to noise-like wrinkles or mustache.

As a pre-process, the generic model is separated into several feature regions like eyes, nose, eyebrows, mouth, and jaw regions as shown in Fig. 2(B). By the global positioning of the generic model on the photographs, the positions of the features, like eyes, are approximated even though they are not accurate. The feature positions are different by the age, human race, gender, etc. Our strategy to find the feature positions is by using statistical data for image processing. Image analysis only has limitation of the accuracy, but knowledge of a face helps the processing. Therefore, we analyze 200 people to know the distribution of the facial features. Fig. 3 shows the distribution graph of feature positions. The X-axis is a position of feature shown by ratio. The position of a 100% indicates the tip of jaw, 0% indicates the top of the head. The Y-axis is a number of people. We randomly picked up people from our database to get the graph. This graph shows that the distribution can be considered as a standard distribution. Therefore, they can be estimated by Gaussian curves of small dispersion, except at the position of the forehead. Thus, we can set constant distribution value, which is calculated from the curve, to the equation to estimate positions. Since most humans have almost symmetric faces, only the vertical distribution is used to calculate the feature positions. Let α be a parameter for each region and σ^2 be

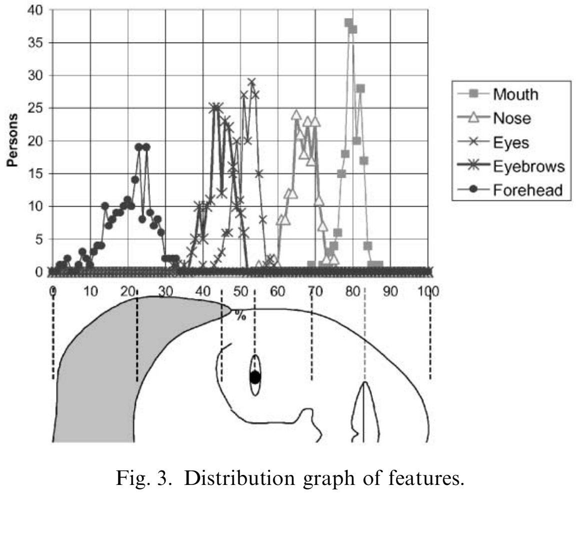


Fig. 3. Distribution graph of features.

a distribution value, and y_0 as a predefined average position of each feature. The Gaussian weight value of the position $W(y)$ is defined as the following equation:

$$W(y) = \alpha \exp\left(-\frac{(y-y_0)^2}{2\sigma^2}\right).$$

The equation is used as a weight value for the feature position estimation. We multiple this value to a self-correlation value of the extracted edge image of the face. Therefore, the face image is processed to make feature edges clear. We first use the Canny edge detector to extract the edges of the features, and use only the horizontal edges. This is because most features on the face, like eyes and mouth, are based on horizontal lines. Therefore, it is efficient to remove wrinkles on the cheeks because they do not have strong horizontal edges.

3.2. Individual feature location matching

First, all the regions are moved at the same time, and the best fitting positions are calculated considering the image matching and the distribution weight. Here we can define pixel value on the image as $P(x,y)$, and pixel value in the eye region as $P_e(x,y)$, also the weight function of the eye region as $W_e(y)$. The probability of the eye's position $E_e(Y)$ is calculated as follows:

$$E_e(Y) = W_e(Y)\sum_x \sum_y P_e(x,y)P(x, Y+y),$$

where Y is the fitting position. Now, considering the total balance of the regions, we use the probability of the eyes $E_e(Y)$, the mouth $E_m(Y)$, the eyebrows $E_b(Y)$ and the nose $E_n(Y)$ that are calculated in the same way. The result of the possibility $E(Y)$ is calculated by the following equation:

$$E(Y) = E_e(Y) + E_m(Y) + E_b(Y) + E_n(Y).$$

An approximated position is defined where the largest value of $E(Y)$ appears. Then these regions are separated into two parts for more detailed position extraction. Eye regions and eyebrow regions constitute one part while nose and mouth regions do the other. From database analysis, it is found that the distance between the eye and eyebrow is not very different from one person to another. The distance from the nose to the mouth is also in a narrow range. This is shown in Fig. 4. The dispersion ranges of the eye–eyebrow and the nose–mouth are narrower than the eye–nose. Thus, again the region fitting is done in the narrower region for each part. In this process, equations that are used for the calculation are the same as shown before, but the differences appear only at the distribution value σ^2. Finally, the parts are completely separated from each other, and the same position fitting calculation is applied.

Since the edge of a bottom line of nose is not always strong, another dependency method is used for the calculations in this region. The positions of the eyes and mouth are fixed before the nose position extraction. Then the distribution is calculated between the eye and mouth in place of face size. Fig. 5 shows the distribution of nose position. Using this distribution parameter, the bottom line of nose is estimated with the similar approach. This process while considering the face balance gives a robust fitting result. Fig. 2(B) shows that these regions fit to the proper position by this matching algorithm.

Using the method described above, 200 faces in our face database built during several public demonstrations [14] are experimented to check the accuracy of the global matching. The database contains a large range of human races, ages of both genders. Also there are people with facial hair. However, the database does not include

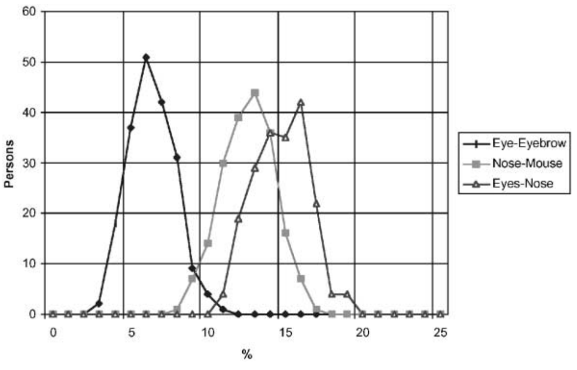


Fig. 4. Distribution graph of each feature's distance. Five percent of distance is the 1/20 length of the face.

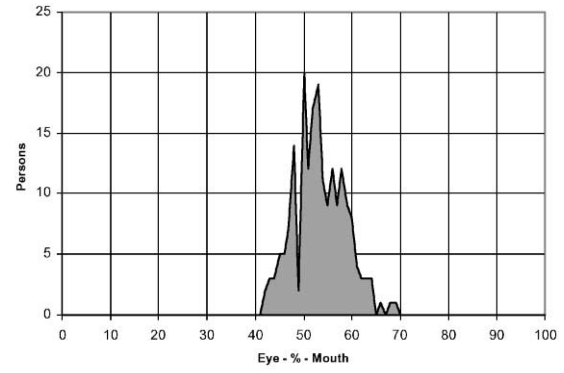


Fig. 5. Distribution graph of a nose tip. Zero percent is the eye position and 100% is the mouth position.

people with glasses or other similar things. The extraction rate in finding the proper position of all features was 201 in 203 cases. The two missed detections occur when either the photographs are very noisy, or the face is very far from the average.

The position of each feature is also applied to the side face. The difficulty for transferring the feature information from the front view to the side view comes from the non-orthogonal views and different illuminations between two photographs. We normally take one photograph of a target person from the front view. Then the target turns about 90°, and we take one more photograph for the side view. It means, the pair of photographs is not taken at the same time and the direction of the light can be changed. In addition the head angle is not always kept in same condition because human can move. If the photographs are taken at the same time from his/her front and side, the data is more accurate. Thus, we have to consider the error from the front and the side. The global position detected from the front view is used to the side view, but these are not considered as accurate positions. The more accurate position of each feature is calculated in detail in the extraction phase of the side view.

4. Detail detection

In the global matching, the position of facial features is obtained. In this section, we describe the method for recognizing the feature shape of each region. This detail matching contains basically the recognition of the size, shape and exact location of features. The features, i.e. regions, to be recognized are eyes, mouth, forehead, chin, nose and eyebrows. The automatic shape extraction of the ears is not always easy since hairs sometimes hide ears. At present time, ear position processing is not supported in our method. The shape extraction of each region is treated differently but the image processing methods we use are similar and the result is given as curves for each region as shown in Fig. 2(C) and (D).

4.1. Jaw curve detection

The definition of the jaw curve on front view is not trivial. Even though everybody has a jawbone, fat and facial hair can blur its edge and the border between the face region and the neck disappears in some cases. In these cases, the jaw curve definition often varies a lot. Thus, fully automatic location of this curve is not possible and therefore we use the following algorithm:

(1) Suppose the generic model approximately defines both sides of the jaw curve, and the bottom position, where it is called tip of the jaw.
(2) Find a curve that connects these three positions.
(3) Modify the curve connecting these points to fit better.

Here, to find the fitting curve is the main problem. It is divided into two parts. One is a preprocessing of the image, and the next one is a curve fitting.

To respect the conventions in order to deform the generic model later, we define a jaw curve as the curve that contains the tip of the jaw, which can be easily found from the side view as shown in Fig. 6.

4.1.1. Image preprocessing flow

There are many edge detector functions. It is common to use such functions before the curve segment fitting. Our purpose is to emphasize the weak jaw curve and force it to remove other edges. Even though the Canny edge detector is famous for the pure edge extraction and is robust to

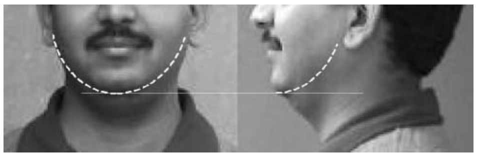


Fig. 6. Our definition of jaw curve.

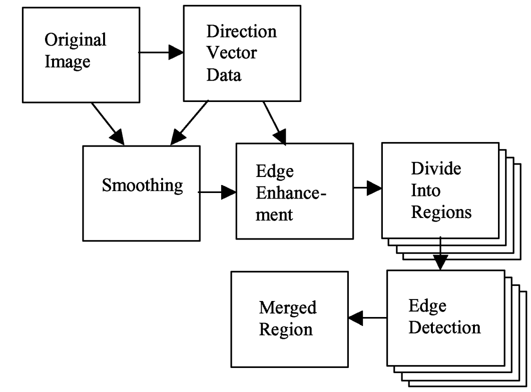


Fig. 7. Image processing flow of a jaw curve enhancement.

extract sensitive edges, it is not suitable for this kind of situations. Thus, we have developed an image processing flow to extract the best edge for our curve-fitting algorithm. Fig. 7 shows this image preprocessing flow for jaw curve location. First, the direction vectors of each pixel are calculated from the original image. This vector information is used to smooth the original image, and enhance the edge. After this edge enhancement, the image is divided into four regions as shown in Fig. 8 and edge detection is done for each region. Finally, the regions are merged into one image. In this flow, only the already-known directions of jaw curve are emphasized to make a curve-fitting algorithm described later work well.

Direction vector calculation: First we define pixel values on the image as $P(x, y)$, and assume that each pixel is made of red and green and blue pixels $P_r(x, y), P_g(x, y), P_b(x, y)$. The luminance of the image for each pixel $Y(x, y)$ is calculated by the following equation:

$$Y(x, y) = 0.30P_r(x, y) + 0.59P_g(x, y) + 0.11P_b(x, y).$$

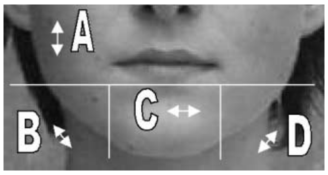


Fig. 8. Regions separated for gradient filter.

This equation is made from the YIQ color model, which is used in color TV broadcast. This luminance value is proportional to the amount of light perceived by the eye. Thus, this can be processed without affecting colors. After this conversion, slope vectors of each pixel to the neighbor directions are calculated. Suppose the z value of each pixel is the value of the brightness. A normalized perpendicular vector, to the curve made by $P(x, y)$ and $P(x + 1, y), \vec{n}_{(1,0)}$, is shown as follows:

$$l = \sqrt{(Y(x, y) - Y(x + 1, y))^2 + 0^2 + 1^2},$$

$$\vec{n}_{(1,0)} = ((-Y(x + 1, y) + Y(x, y))/l, 0, 1/l).$$

The sum of the neighbor vector $\vec{n}$, shown below, is the direction of the steepest slope, and the level of the slope:

$$\vec{n}_{(0,0)} = (0, 0, 0),$$

$$\vec{n} = \sum_{x=-1}^{1} \sum_{y=-1}^{1} \vec{n}_{(x,y)}.$$

Directional smoothing filter: A smoothing filter is sometimes used for blurring an image. This kind of filter makes an image smooth looking, and it also removes noise. In image processing, this kind of filter is used for removing undesired details of an image. Considering the computation overhead of the function, a 3×3 spatial filter, which all coefficients have a value of one, is commonly used.

When the effect from this filter is not enough, we may increase the size of the filter. The output of the larger matrix makes a smoother image. However, this filter will also delete the true edge, not only the noise. The jaw curve is sometimes very weak and this kind of side effect will cause to miss location of the curve. To avoid this side effect, we propose a method to use directional information calculated previously. The direction to do smoothing is perpendicular to the previous calculated direction. Thus, convert it to four directions that are 0°, 45°, 90°, 135°, and use four kinds of smoothing filter for the each direction.

Edge detection: Sometimes an edge of a jaw is not strong, but everybody has a jaw with similar shape. To obtain an effective result from the weak information, the image is separated into four parts and a gradient filter is used for each part. Let A, B, C and D on the image be a region in the upper half, lower left, lower middle, lower right of the jaw image as shown in Fig. 8. Suppose the jaw shape is represented as an arc of a half-ellipse, only vertical or near to vertical edges appear on the region A. Region C has horizontal edges, and the regions B and D have edges of diagonal direction. Therefore, four gradient-filter to each region A, B, C and D are prepared as follows.

$$\begin{vmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{vmatrix} \begin{vmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{vmatrix}$$

$$\begin{vmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{vmatrix} \begin{vmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{vmatrix}$$

These simple gradient filters are better than sensitive edge detectors like Canny or Deriche in its processing speed and the result is good enough to track the curve. The output from this flow makes an enhanced edge to help curve fitting later.

4.1.2. Jaw curve fitting

As an outline-tracking algorithm, 'Snake' method created by Kass et al. [11] is widely recognized. This method uses a controlled continuity spline function, and transforms the shape of the curve to make the energy function minimize from the initial state of the curve. However, the problem is that the calculation cost is expensive and therefore it is quite slow. Thus, we have to adapt this method to a jaw curve fitting for fast processing. We use a few control points, and use them effectively with pre-analyzed parameters. We use the symmetrical information and the curvature value, and make an inner energy function corresponding to mean jaw shape. Fig. 9 shows some samples of jaw shapes. This graph indicates that the shape varies for each individual, but there is symmetry in almost all the cases. First, the initial position of the curve is defined by the following equation:

$$\left(\frac{x - C_x}{r_1}\right)^2 + \left(\frac{y - C_y}{r_2}\right)^2 = 1,$$

where $y < C_y$ and (C_x, C_y) is the center of the arc, and r_1, r_2 are the radiuses as shown in Fig. 10. Let S_n be point on the defined curve, N be a number of points, and $\theta(S_n)$ be a curvature at S_n as shown in Fig. 11. Length of the curve l can be defined as follows:

$$l = \sum_{n=0}^{N-2} |S_n - S_{n+1}|.$$

The external energy E_{ext} that is line-fitting value on the image is shown by the following equation with P, a pixel value on the curve from S_n to S_{n-1}. Parameter γ is a constant value,

$$E_{\text{ext}} = -\gamma\left(\sum_{n=0}^{N-2} \int_{S_n}^{S_{n+1}} P\right)/l.$$

This value becomes larger when the curve fit to pixels that have strong preprocessed edge value. The energy to keep the curve to be a jaw shape E_{cur} is calculated with value $\theta(S_n)$, and a parameter of

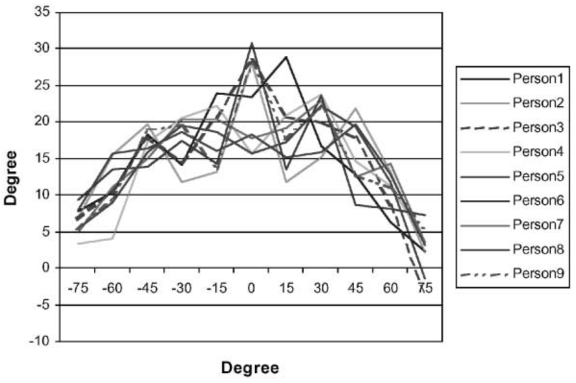


Fig. 9. Symmetry of a jaw shape. Set tip of jaw to 0°, and left and right positions to 90°.

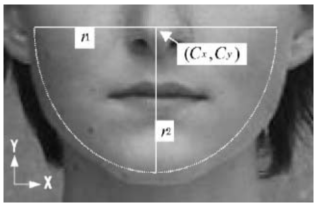


Fig. 10. Initial position of a deforming curve.

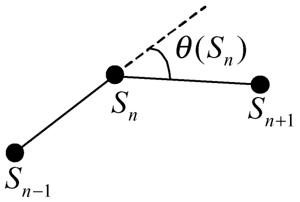


Fig. 11. Definition of an angle.

jaw shape α_n:

$$E_{\text{cur}} = \sum_{n=1}^{N-1} \alpha_n(\theta(S_n)).$$

Symmetrical energy E_{sym} is calculated as follows:

$$E_{\text{sym}} = \sum_{n=1}^{(N-1)/2} \beta_n(|\theta(S_n) - \theta(S_{N-n})|),$$

where β_n is a symmetrical range parameter. In addition, since the position of tip of jaw and approximated positions of both sides are predefined, position energy E_{pos} can be defined as follows:

$$\begin{aligned} E_{\text{pos}} &= |S_{N/2} - (C_x, C_y - r_2)|^2 \\ &\quad + |S_0 - (C_x - r_1, C_y)|^2 \\ &\quad + |S_N - (C_x + r_1, C_y)|^2. \end{aligned}$$

With these energies, the sum of the energy E is calculated as follows with pre-calculated energies:

$$E = E_{\text{ext}} + E_{\text{cur}} + E_{\text{sym}} + E_{\text{pos}}.$$

The minimum value shows the converged jaw curve. Fig. 12 shows the final fitting result of the jaw image. If the photo is not exactly from the front view, the fitting result cannot be accurate.

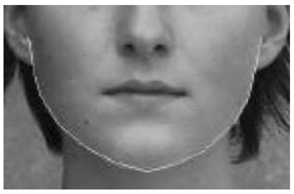


Fig. 12. Fitting result of a deforming curve.

4.1.3. Forehead curve detection

The forehead curve and the jaw curve have similar characteristics for processing the image. The similarities and the differences between the jaw and the forehead location are as follows:

(1) Both have weak curves. It is necessary to enhance the feature of the curve.
(2) Both are arc shapes. Same method can be used to fit curves. However, the fitting shape is not the same.
(3) Forehead curve cannot be defined as a physical model. The difficulty is that there are some people who do not have hairs while others have long front hairs to hide their forehead. In these cases the definition of the forehead is not clear.
(4) Forehead region has sometimes noise due to front hairs. Front hairs make strong edges compared with a border curve between skin and hairs. In addition, sometimes the symmetry is not kept because of front hairs.
(5) There is a freedom of hairstyle. Forehead shape is different to each individual. It cannot be defined as with the jaw shape.

Thus, the image preprocessing method to remove front hair noises and the parameters to fit the forehead curve are different from the jaw curve location method. The results of two forehead shapes are shown in Fig. 13.

4.2. Nose shape detection

The definition of the nose region is not clear and also the curves around the nose are not easily recognizable. The curve of the nose goes up very smoothly from the cheek to the tip of the nose, and it does not make a proper feature. The feature also does not exist at the tip of the nose. Lights in the environment also influence to the image. Thus, a

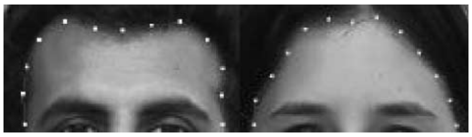


Fig. 13. Forehead fitting results.

similar processing method like for a jaw curve or a forehead curve cannot be adapted to the nose shape recognition. In addition, there are not many needs to recognize a nose shape, except a model construction, and few researches have been done for the extraction. However, nose wings can be useful for nose shape extraction, since an edge, though weak, can be seen. We solve the nose shape extraction problem by dividing the algorithm into several steps.

4.2.1. *Nose wing detection*

Fig. 14 shows an image of a nose and the two curves that will be used for fitting the nose wing. On this image I_0, we define the left points A, B as $(x_a, y_a), (x_b, y_b)$, and the right points A', B' as $(x'_a, y'_a), (x'_b, y'_b)$. $P(x, y)$ and $P'(x, y)$ show the points on the each curve. Because of their similarities, we will only discuss the left curve. Let θ be the angle of the left curve,

$$\theta = \tan^{-1}\left(\frac{y_b - y_a}{x_b - x_a}\right).$$

Let σ^2 be a distribution value and x and y be the distances from the center pixel. Normal Gaussian operator for two-dimensional case $G(x, y)$ is defined as follows:

$$G(x, y) = \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right).$$

This function is sometimes used as a smoothing operator in image processing. A constant value related to the distribution is ignored to simplify the equation. The graph of this equation is symmetric around its center, and the distribution parameter defining the region is smooth. When the distribution parameter is large, the accuracy of the position of extracted edge becomes worse. On the other hand, large distribution parameter helps in removing the noise. Hence, we change the circle

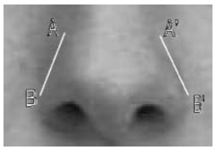


Fig. 14. Nose wing fitting.

shape to an ellipse to enhance the edge information in the proper direction. It is important to process the image, because the edge is sometimes very weak at the nose wing. Let X and Y be rotated angles of the former x and y and define the rotation angle as θ. Also, define the constant parameter α and β,

$$X = \left(\frac{x\cos\theta - y\sin\theta}{\alpha}\right), Y = \left(\frac{x\sin\theta - y\cos\theta}{\beta}\right).$$

The definition of Gaussian operator changes to the following:

$$G_\mathrm{d}(x, y, \theta) = \exp\left(-\frac{X^2 + Y^2}{2\sigma^2}\right).$$

Now, let $\nabla^2 G_\mathrm{d}(x, y, \theta)$ be Laplacian of Gaussian (LoG) of the $G_\mathrm{d}(x, y, \theta)$. New image $I_1(x, y, \theta)$ is calculated by the convolution with LoG as follows:

$$\nabla^2 G_\mathrm{d}(x, y, \theta) = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\exp\left(-\frac{X^2 + Y^2}{2\sigma^2}\right),$$

$$I_1(x, y, \theta) = I_0(x, y) * \nabla^2 G_\mathrm{d}(x, y, \theta).$$

We can define the length of the curve defined by (x_a, y_a) and (x_b, y_b) by following simple equation:

$$l = \sqrt{(x_b - x_b)^2 + (y_b - y_a)^2}.$$

Remember that θ is also defined by (x_a, y_a) and (x_b, y_b). Hence, the probability function E_nose can be defined by the following equation:

$$E_\mathrm{nose}(x_a, y_a, x_b, y_b) = \frac{\sum\sum I_1(x, y, \theta)P(x, y)}{l} + \frac{\sum\sum I'_1(x, y, \theta)P'(x, y)}{l'}.$$

The maximum value of E_nose can be considered as the best fitting curve to the nose wing. Thus, we find the four parameters (x_a, y_a) and (x_b, y_b) at the best matching position.

4.2.2. *Making of nose shape*

The detection of the nose wings position is not enough to make a complete shape of a nose. First, point A and A' in Fig. 14 are connected to the top of the nose, points C and C' , to make the nose shape clear. The points C and C' are virtual points, and these points have no definition from image processing. These are defined at one of the fourth position between the eyes. Then, to be able to deform the shape using the following merit function, the curve is divided into several points. Let $\theta(S_n)$ be an angle between two curves crossing at S_n as shown in Fig. 11. $W_n()$ is a weight function for each position of the nose. E_{nose} is a probability function calculated by the equation described for the nose wing extraction:

$$F = E_{\text{nose}} - \sum_n W_n(\theta(S_n)).$$

This function means that the value F is small when the curve is not similar to a nose shape, or the curve does not fit to the image. Thus, the curve of the nose shape is deformed to fit to the nose. After this processing, the left and right balance is considered and the detailed shape is created. Fig. 15 shows the fitting result of the curve on the nose. In this figure, it can be seen that the curve fits to a very weak edge and is in the proper position.

4.3. *Eye, eyebrow and mouth*

The eye's texture is different from the skin and has a higher reflection rate than other features. The edge around an eye appears strong, but the reflection in the pupil may disturb processing and may provoke strong noise. A wrinkle on the eyelid may also confuse the extraction process. Tistarelli et al. [19] use a deformable template-matching algorithm for extracting the eye shape. However, we found it difficult to apply this method to a low-quality image. The algorithms used to recognize a very weak edged shape in the nose shape section, and to recognize connecting curve in jaw and forehead section can also be applied to detect eyes, eyebrows and mouth. To detect an eye:

(1) The left and right eye positions are considered symmetrical. Then, the approximate pupil positions are detected by a convolution method.
(2) The circle shape of the pupil model is used to fit to the image and some position candidates are chosen. The combinations where symmetrical information is not valid are removed.
(3) The method used in the jaw shape location is used to fit the upper eyelid and lower eyelid.
(4) Other possibilities of the upper eyelid are also calculated to avoid confusing wrinkles with an eye curve, and these candidates are also checked.

The eye curve is thus extracted. Fig. 16 shows the fitting results. The curves are not influenced by the wrinkles, and the proper positions are detected.

The mouth is similarly recognized. However, the border between lip and skin is not always clear. Also a moustache and a beard sometimes influence the extraction and the borders are not constant. Our research shows that there is a relation between the horizontal nose width and the lip width. For example, a person who has a big nose has a big possibility to have a thick mouth. The initial lip thickness is defined from this relation, using previously detected nose width. Then, the same

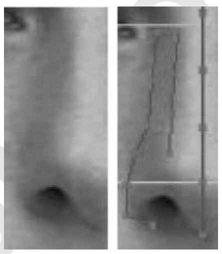


Fig. 15. Result of the nose fitting. Left side is the original image. Right side is the fitting result. The feature curves are matched to the weak edges.

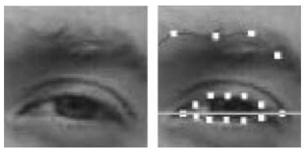


Fig. 16. Result of eye fitting.

method used during the nose wing detection is used to detect the lip shape.

4.4. Side face

Approximated eye positions, nose shape and mouth position are already recognized during global matching. These results can be used for the side face extraction. However, it is common that when the face is rotated, and the positions become incorrect. The side face extraction is done as follows:

(1) The same generic model for the front face is placed on the side face as shown in Fig. 2(A).
(2) The front part of the generic model is deformed to fit to the approximate position of the recognized features.
(3) Nose curve is extracted. The edge, which appears almost as a straight line, will be the base curve of the front face.
(4) Convex and concave information of the front face are used, and other parts are deformed. For example, a convex curve appears near the position of the eyebrow and a concave curve appears between the eye and eyebrow position. The positions of eye and eyebrow are located at the front view extraction.

Vertical length and horizontal length of skull are almost the same. The generic feature is then deformed to fit to the side face image as shown in Fig. 2(D).

5. Individualized 3D model creation from detected features

We detected 2D feature points on two photographs automatically in previous sections. The question is how to modify a generic model, which has more than a 1000 points to make an individualized smooth surface. First, we have to choose if we do the modification in 2D and then combine two sets of 2D to make 3D or we do the modification in 3D directly. This is a critical issue for the data handling for error tolerance. We have to take a method to decrease the dependency of perfectly orthogonal photograph input. Several kinds of distance-related functions in 2D have been employed by many researchers [1,10,12] to calculate the displacement of surface points related to the feature points detected. However, these methods cannot recover the proper shape when the detected features are not perfectly orthogonal.

As we described in the end of Section 3, the front and side view photographs are taken in an environment where the two views are not perfectly orthogonal and the illuminations are different. We apply an algorithm on detected 2D features to make 3D features aiming to decrease the dependency of orthogonality of photographs and then take modification in 3D. Since we know the feature point structure from photographs, there are possibilities to handle them in a proper way by filtering using the knowledge of feature points.

5.1. Asymmetric 3D features from two sets of 2D features

Our input is two views of a person. One frontal view and one side view are the only input. Since only one side view is used, one may misunderstand that the final virtual clone has a symmetric face. However, the frontal view has almost all information for asymmetric information and the way to produce 3D points from two 2D points saves asymmetric information of the face. Two sets of feature points on frames have structure, which means every feature point has its own name. In our case, we use about 160 feature points. Some points among them have position values, which means they are visible on images, on both the frontal and the side views, while others have values only on the frontal view or on the side, for example, back part of a head is not visible in the frontal view. The problem is how to make (x, y, z) from a given set of 2D points. Since the perfect orthogonal pair photographs are not easy to be realized, it may result in unexpected 3D shape if we take an average of y_s and y_f for y coordinate where subscripts s and f mean side and frontal view. These are our criteria to combine two sets of 2D features, say (x, y_f) and (z, y_s), to make 3D features. Fig. 19 shows the name convention we are using such as

- FV means the feature point has value (x, y_f) on the frontal view,

- SV means the feature point has value (z, y_s) on the side view,
- _R_ means the right side region indicated as _R_,
- _L_ means the left side region indicated as _L_,
- _C_ means neither _R_ nor _L_.

We use following algorithm to construct 3D features from two sets of 2D detected features on two views of a person:

(1) _R_, FV, !SV: We use a predefined relation from a typical face to get z, for instance the depth value z of inner corner point of eye is calculated from middle and outer corner points of the eye.
(2) FV, SV: Points have x, y_f, y_s, z, and we take y_f for y value. Note that we do not take the average of y_f and y_s.
(3) _R_, !FV, SV: We use a predefined relation from a typical face to get z, for instance the x value, for instance the x of neck point invisible is calculated with visible neck points of the frontal view.
(4) _L_, FV, !SV, _L_: We take z value from the corresponding point with _R_ that has the reflection counter part by vertical line, we take the depth value z from the reflection part., for example the left corner of the left side eye is corresponding to the right corner of the right side eye.
(5) _L_, !FV, !SV: We use a predefined relation from a typical face to get x and we take y and z values from the corresponding point with _R_.
(6) !FV, SV, _C_: We take x value from the line between two features with FV and _C_, for example we take the feature on top of the hair and middle chin feature or simply 0. Here we set x as 0, which will be used in the texture coordinate fitting method later.
(7) We take y_s for y value in 'ear' region, which is useful for 'ear' texture for texture mapping.

5.2. Modifying the generic model in 3D and texture mapping

We modify a generic model using 3D features. The Dirichlet Free Form Deformation [16] is used to change the shape of the generic model by taking 3D feature points as control points for deformation. The heavy calculation to get coefficients of vertex movement related to control points is done only once as a pre-process with the generic model and the saved coefficients for vertices are applied for each new shape modification. Then to increase photo-realistic looking of virtual objects, we utilize texture mapping by applying real images onto them. For virtual faces, the texture can add a grain to the skin, including the color details for the nose, lips and etc. Texture mapping needs both a texture image and texture coordinates. The input images are not appropriate to be used for texture mapping since we use two photographs taken with only one camera asking the target rotate, which changes the light condition and luminance; hence, the generation of texture images also needs to be processed. In this case just a simple combination of the frontal view and the side view does not provide properly smooth texture mapping, caused by both non-perfect orthogonal condition and non-coherent luminance.

There are two steps for this image generation:

(1) Geometrical deformation of the side view photographs to integrate to the front view photograph. This deformation is to resist non-perfect orthogonal photograph input. The detected features are used to make automatic deformation of the side view image and integration on to the front view.
(2) Smoothing of boundaries between different images to resist non-coherent luminance. The smoothing is obtained using multi-resolution techniques.

After generation of the one single image for texture mapping, we calculate texture coordinate for each vertex on the 3D facial model. To give a proper coordinate on a texture image for each point on a head, we first project an individualized 3D head onto three planes, the frontal (x, y), the left (y, z) and the right (y, z) planes. With the information of features used for geometrical deformation, we decide on which plane a 3D-head point is projected. More detailed description for the process is found in another literature [14]. Fig. 17 shows some examples of final individualized 3D heads.

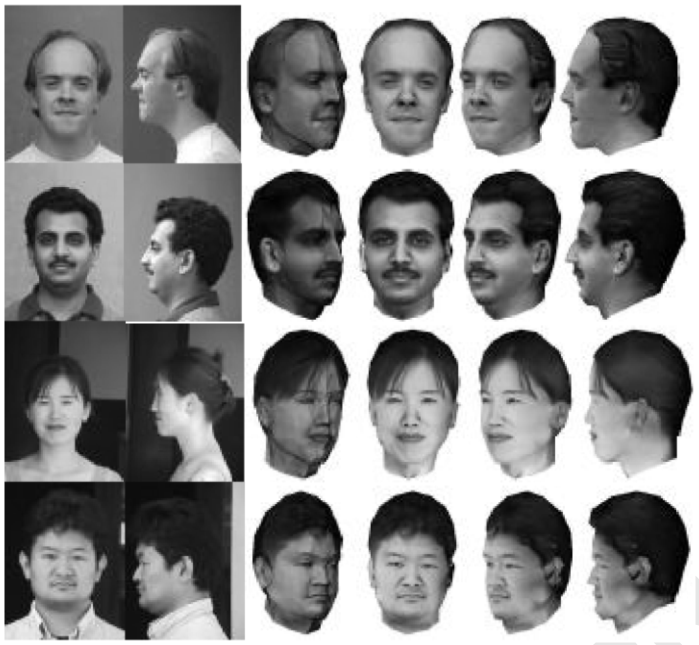
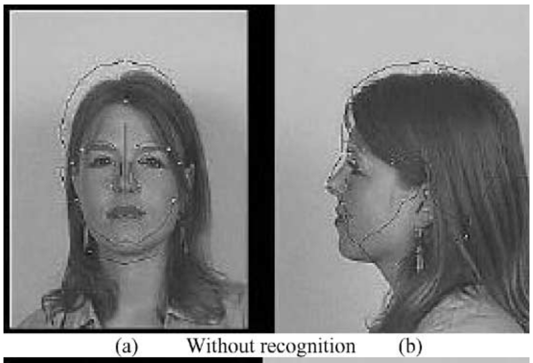


Fig. 17. 3D construction results.

6. Discussion for automatic feature detection

We discuss both a global matching part and a detailed shape extraction part. The global matching has 99.0% rate of success. The main reason of the high rate is that feature positions of most people are in Gaussian distribution. Thus, the image recognition with the feature position analysis makes the success rate high. The failure occurs when strong noise like a mustache appears and the distribution of facial features is far from the average.

The detailed shape extraction part is separated into forehead, jaw, eyes, nose, mouth and side view. The jaw shape extraction is successful when the symmetry is kept. If the front photo is not taken exactly from the front view, the accuracy of the curve fitting drops. However, the success rate is more than 90% with photographs that are taken from the exact angle. The forehead extraction is also in the same situation, but is influenced more by the front hair. The nose shape extraction is sometimes influenced by a strong edge of a nose hole. Some nose holes can be seen from the front view, but not everything can be, which makes the appearance not clear. Our nose shape extraction algorithm supposes to extract very weak edge curves, and sometimes fits to the nose holes. The success rate is about 70%. The eye and mouth shape extraction has about 80% success. The side view silhouette extraction depends on the background image. The success rate of every shape extraction is calculated by the multiple of every region extraction. Thus, we cannot say this is a full automatic shape extraction. However, the method helps to fit features on the face. An operator has to edit miss-recognized shapes after the extraction. In addition, feature extraction of silhouettes is currently not supported since the movement of the curve is influenced by the background intensity and long hair is difficult to deal with. An operator must also move these curves, but there are a few control points to make these curves. Thus, an auto-fitting of these curves does not help shortening the operation time. However as we describe how to create 3D features from two sets of 2D features in Section 5.1, the accurate feature detection on the side view is not as important as the front view. Most information used for 3D face creation comes from the front view and only missing information for depth comes from the side view. Fig. 18(C) and (D) show feature extraction result with the method explained in this paper. Fig. 18(A) and (B) shows the generic feature positions without adaptation to the photographs.



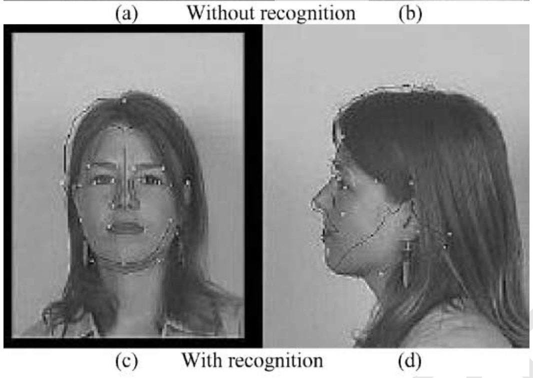


Fig. 18. Comparison of results.

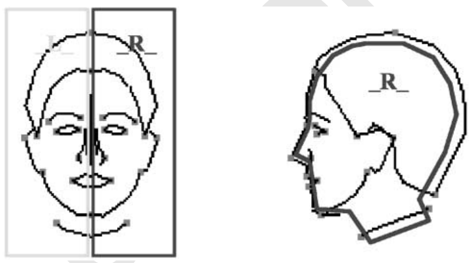


Fig. 19. Feature points names are organized in a special way to indicate if the feature belongs to _R_, _L_ or centerline to make the algorithm to create 3D points.

With the image-processing technique, the time to place feature markers on the face becomes less than 20 s with Pentium II, 333 MHz machine. This automatic feature detection method supports any unskilled operator to produce her/his 3D animatable virtual clone. The total process time including automatic feature detection, 3D surface deformation and texture generation and mapping is about 1 min.

7. Conclusion

In this paper, we address our novel approach for an automatic face feature detection to help the process of 3D model creation. The approach has two main steps: (i) global matching to look for feature positions, and (ii) detailed matching for feature shape detection. A statistical analysis of our face photographs database and feature character driven image analysis and processing are the important factors for our robust and practical feature detection methods. Well combined with successive processes to fulfill the complete 3D facial creation methodology such as 3D feature construction, 3D surface deformation, and texture generation and mapping, the success rate marks very high with various races, aged, gender group of people. With this automatic feature detection, we can now create an individual face 3D-model in about 1 min.

However, there are some parts that are not supported to extract the shape: (i) The parts that do not contain many points and easy to edit. Automatic fitting does not help shortening the operation time. (ii) The parts automatic fitting is difficult. These are the future work to complete automatic 3D model generation. In addition, we expect to add more processing works to increase the accuracy of extraction in future. The trade off between the manual job and the recognition speed should be considered to improve the performance.

Acknowledgements

This research is supported by European project CROSSES (N_IST-1999-10510).

References

[1] T. Akimoto, Y. Suenaga, S.W. Richard, Automatic creation of 3D facial models, IEEE Computer Graphics and Applications, IEEE Computer Society Press, Silver Spring, MD, 1993.

[2] R. Brunelli, T. Poggio, Face recognition through geometrical features, in: ECCV'92, S. Margherita Ligure, Springer, Berlin, 1992, pp. 792–800, http://hera.itc.it:3003/~brunelli/.

[3] T.F. Cootes, C.J. Taylor, Locating objects of varying shape using statistical feature detectors, in: European Conference on Computer Vision, Cambridge, England, 1996.

[4] P. Fua, Y.G. Leclerc, Taking advantage of image-based and geometry-based constraints to recover 3-D surfaces, Comput. Vision Image Understanding 64 (1) (July 1996) 111–127.

[5] P. Fua, Regularized bundle-adjustment to model heads from image sequences without calibration data, Int. J. Comput. Vision 28 (2) (July 2000).

[6] C. Garcia, G. Zikos, G. Tziritas, Wavelet packet analysis for face recognition, J. Image Vision Comput. 18 (4) (2000) 289–297, http://www.csd.uch.gr/~tziritas/publications.html.

[7] T. Goto, M. Escher, C. Zanardi, N. Magnenat-Thalman, MPEG-4 based animation with face feature tracking, in: CAS'99 (Eurographics Workshop), Milano, Italy, September 7–8, 1999, Springer, Wien, New York, pp. 89–98.

[8] T. Goto, M. Escher, C. Zanardi, N. Magnenat-Thalmann, Multimodal interaction in collaborative virtual environments, in: International Conference of Image Processing (ICIP'99), Vol. 3, Kobe Japan, October 25–28, 1999, pp. 1–5.

[9] N. Intrator, D. Reisfeld, Y. Yeshurun, Face recognition using a hybrid supervised/unsupervised neural network, Pattern Recognition Lett. 17 (1996) 67–76, http://www.math.tau.ac.il/~nin/.

[10] H.H.S. Ip, L. Yin, Constructing a 3D individualized head model from two orthogonal views, Visual Comput. 12 (1996) 254–266.

[11] M. Kass, A. Witkin, D. Terzopoulos, Snake: active contour models, Int. J. Comput. Vision (1988) 321–3.

[12] T. Kurihara, K. Arai, A transformation method for modeling and animation of the human face from photographs, in: Proceedings of Computer Animation'91, Springer, Tokyo, 1991, pp. 45–58.

[13] W. Lee, P. Kalra, N. Magnenat-Thalmann, Model based face reconstruction for animation, in: Proceedings of MMM'97, World Scientific Press, Singapore, 1997, pp. 323–338.

[14] W. Lee, N. Magnenat-Thalmann, Fast head modeling for animation, J. Image Vision Comput. 18 (4) (2000) 355–364.

[15] T.K. Leung, M.C. Burl, P. Perona, Finding faces in cluttered scenes using random labeled graph matching, Comput. Vision (1995).

[16] L. Moccozet, N. Magnenat-Thalmann, Dirichlet free-form deformations and their application to hand simulation. in: Proceedings of Computer Animation'97, IEEE Computer Society Press, 1997, pp. 93–102.

[17] M.J.T. Reinders, P.J.L. Beek, B. Sankur, J.C.A. Lubbe, Facial feature localization and adaptation of a generic face model for model-based coding, Signal Process. Image Commun. 7 (1995) 57–74.

[18] M.J.T. Reinders, R.W.C. Koch, J.J. Gerbrands, Locating facial features in image sequences using neural networks, Automat. Face Gesture Recognition (1996) 230–235.

[19] M. Tistarelli, E. Grosso, Active vision-based face authentication, Image Vision Comput. 18 (4) (2000) 299–314.