An ellipsoidal model for generating realistic 3D facial textures

YoungJae Park

Department of Computing,
Soongsil University,
Sangdo 1-dong, DongJak-gu,
Seoul, Republic of Korea
E-mail: p1y1j1@ssu.ac.kr

SeokWoo Jang

Department of Digital Media Engineering,
Anyang University,
Anyang 5-dong, Manan-gu, Anyang-shi,
Kyunggi-do, Republic of Korea
E-mail: swjang@anyang.ac.kr

JoongJae Lee

Seocho R&D Campus,
221, Yangjae-dong, Seocho-gu,
Seoul, Republic of Korea
E-mail: arbitlee@naver.com

YangWeon Lee

Department of Computer Information Engineering,
Kunsan University,
1170, Daehangno, Gunsan,
Jeonrabuk-do, Republic of Korea
E-mail: ywrhee@kunsan.ac.kr

GyeYoung Kim*

Department of Computing,
Soongsil University,
Sangdo 1-dong, DongJak-gu,
Seoul, Republic of Korea
E-mail: gykim11@ssu.ac.kr
*Corresponding author

Abstract: In this paper, we suggest an image registration algorithm based on an ellipsoidal model with size-variable blocks that is similar in shape to a human face. While blocks are matched, the existing cylindrical model only considers that left-right curvature can accomplish a correct alignment on left and right images. However, registration errors are produced from up and down images as the cylindrical model does not reflect human head and jaw shape characteristics. The proposed algorithm exploits a block matching algorithm that uses size-variable blocks. The left-right and up-down curvature of the ellipsoidal face model is considered and input images are correctly aligned using their correlation. Next, we employ an image mosaic technique to generate a realistic facial texture from the aligned images. We stitch the images by assigning linear weights according to the overlapped regions and remove ghost effects to make the facial texture more natural. The experimental results show that the proposed algorithm can realistically generate 3D facial textures compared to other conventional methods.

Keywords: facial texture; ellipsoidal model; curvature; image mosaic; registration algorithm; correlation; block matching; weighting factor; stitching; colour.

Reference to this paper should be made as follows: Park, Y., Jang, S., Lee, J., Lee, Y. and Kim, G. (2013) 'An ellipsoidal model for generating realistic 3D facial textures', *Int. J. Computer Applications in Technology*, Vol. 46, No. 1, pp.36–44.

Biographical notes: YoungJae Park is a student at the Computing Department of the Soongsil University, Seoul, Korea. He received his MS in Computer Science from the Soongsil University of Korea, in 2008. His research interests are in computer vision, face detection, adult image identification, digital image processing and pattern recognition.

SeokWoo Jang is a Full-time Lecturer in the Department of Digital Media at Anyang University, Anyang, Korea. He received his PhD in Computer Science from the Soongsil University, Seoul, Korea, in 2000. His research interests are in robot vision fuzzy systems, AR, video indexing and retrieval, intelligence gram, biometrics and pattern recognition.

JoongJae Lee is a Postdoctoral Fellow in Korea Institute of Science and Technology (KIST), Seoul, Korea in 2006-2010. He received his PhD in Computer Science from Soongsil University of Korea, in 2005. His research interests include computer/robot vision, advanced driver assistant system (ADAS), augmented reality, and human robot interaction.

YangWeon Lee is a Professor in the Department of Computer Information Engineering at National Kunsan University, Kunsan, Korea. He obtained his PhD in Computer Science from the Soongsil University, Seoul, Korea, in 1994. His research work has centred on computer vision systems.

GyeYoung Kim is a Professor at the Computing Department of the Soongsil University, Seoul, Korea. He received his PhD in Computer Science from Soongsil University of Korea, in 1996. His research interests are in computer vision, multimedia database, augmented reality, human-computer interface, and biometrics.

1 Introduction

Modelling a 3D human face has been a significant research topic in computer graphics and computer vision, because it can provide users with more friendly and realistic interfaces. This technique has widely been used in many real applications, including virtual reality, computer games, video teleconferencing, surgical facial planning and entertainment. Though many researchers in face modelling focus on realistic facial texture generation, real-time processing, automation, and implementing adaptive algorithms, accurately generating complete and realistic 3D face models with computers remains a difficult task to achieve (Lee et al., 2011).

We can find many approaches to realistic 3D face modelling in related literature. A method using hardware equipment, such as 3D laser scanners and motion capturing devices obtains detailed geometry and realistic textures, but it requires a large volume of data and great expense (Wang et al., 2005). The frontal and profile 2D face image-based method first extracts feature points from 2D images and deforms the general 3D model by matching the extracted points against the 3D model feature points (Lee and Magnenat-Thalmann, 2000; Ivanov et al., 2003). Next, this approach generates the final face model through texture mapping. However, this technique has unnatural results in directions other then the frontal and profile views. The cylindrical model-based method using 2D images estimates the camera parameters by utilising five cameras setup in different directions and the coordinates of image feature points (Pighin et al., 2002; Chang and Chen, 2002). This method then reconstructs the geometric structure of a 3D model by extracting the 3D coordinates corresponding to the image feature points. This approach subsequently generates the cylindrical texture map using the reconstructed 3D model and 2D images. The method has some limitations, as it should accurately extract the geometric information of the deformed 3D model. The four or eight-view face image-based method projects a 3D face model into 2D images and deforms the model and generates a 3D model similar to a head by reconstructing it back into 3D (Pighin et al., 1998). The method should capture the images in correct directions to generate natural textures, because it maps into the deformed 3D model using image stitching techniques only. To resolve this problem, Lee and Choi (2001) proposed image registration using size-variable blocks of the cylindrical model that considers the facial curvature. The method can correctly align the left and right images because it reflects the left and right face curvature. However, the cylindrical model-based method does not consider the curvature of face in the top and bottom directions. This approach thus has errors when aligning top and bottom images. To solve this problem, we propose a new realistic facial texture generation method using an ellipsoidal model that is most similar in shape to the head.

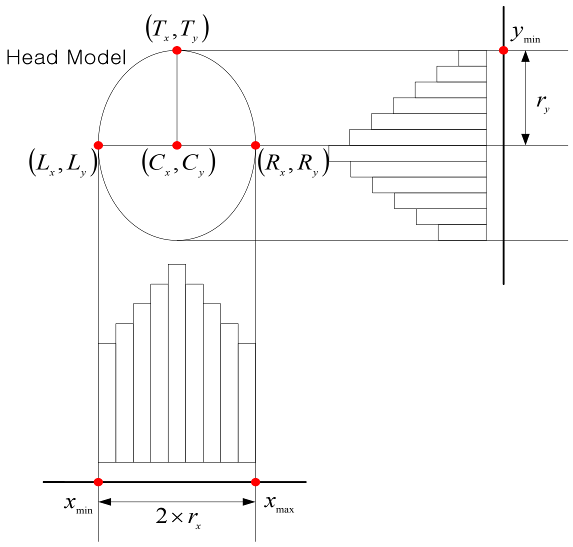
The suggested realistic 3D facial texture generation system contains two main steps: the image registration and image mosaicing steps. The image registration step aligns input images based on the size-variable blocks of the ellipsoidal model, which assumes that the shape of a head is ellipsoidal and aligns the 2D images captured in different directions using a block matching algorithm. When matching blocks, we use size-variable blocks according to the curvature of the facial model. As a block matching metric, colour and texture features are integrated to look for the best matching blocks. The image mosaicing step adopts the image stitching technique to generate the facial texture image. In this step, realistic facial textures are produced by eliminating ghost effects after images are stitched.

The rest of this paper is organised as follows. Section 2 describes the method to extract the parameters of an ellipsoidal model. Section 3 explains the image registration algorithm based on the size-variable block matching. Section 4 describes the image mosaicing technique to generate realistic facial textures. In Section 5, we present some experimental results to show that the suggested ellipsoidal model-based method can effectively generate facial textures compared to other conventional methods, and Section 6 gives conclusions and future work.

2 Estimating ellipse parameters

To align images using the ellipsoidal model, the ellipse parameters r_x and r_y must be determined. In this figure, r_x denotes the minor axis and r_y the major axis. Figure 1 shows how to estimate the ellipse parameters using image projection. In Figure 1, the ellipse represents a binarised head image. The minor axis r_x is computed by horizontally projecting the binarised image, and the major axis r_y is determined utilising vertical projection and the centre coordinates of the ellipse as the boundaries of the neck and jaw areas are not clear.

Figure 1 Estimation of ellipse parameters (see online version for colours)



In (1), $I(x, y)$ represents the binarised head image, $P_h(x)$ denotes the number of black pixels at the x position, and $P_h(y)$ is the number of black pixels at the y position. $x_{\min}$ is the x coordinate at which the first non-zero minimum value of $P_h(x)$ exists and becomes the left bound of the ellipse, while $x_{\max}$ becomes the right bound. Similarly, $y_{\min}$ becomes the upper bound of ellipse, but $y_{\max}$ is not considered a parameter because it contains the neck area. The centre-position of the ellipse (C_x, C_y) can be calculated using the x and y coordinates of $x_{\min}$ and $x_{\max}$, and r_y is determined by the difference between C_y and $y_{\min}$.

$$r_x = \frac{|x_{\min} - x_{\max}|}{2},\ r_y = \frac{|y_{\min} - C_y|}{2} \tag{1}$$

$$p_h(x) = \sum_y I(x, y),\ p_h(y) = \sum_x I(x, y)$$

$$\text{where} \begin{cases} x_{\min} = \min\limits_x \{p_h(x) > 0\},\ x_{\max} = \max\limits_x \{p_h(x) > 0\} \\ y_{\min} = \min\limits_y \{p_h(y) > 0\},\ y_{\max} = \max\limits_y \{p_h(y) > 0\} \\ C_x = \dfrac{x_{\min} + x_{\max}}{2},\ C_y = \dfrac{L_y + R_y}{2} \end{cases} \tag{2}$$

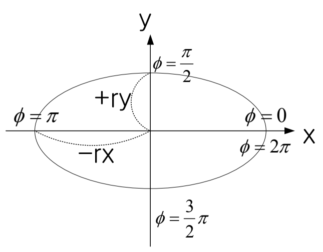
3 Image registration using the ellipsoidal model

To generate natural and realistic facial textures, input images must be correctly aligned. In this paper, we define the 3D face model as an ellipsoidal one that is most similar in shape to the head such that we can resolve the texture generation errors produced when aligning face images captured at inaccurate angles. To reflect the geometric curvature feature of the face model, size-variable block matching is performed when 2D face images are aligned. The suggested ellipsoidal model-based registration is more accurate than the existing cylindrical model-based method because the former considers upper and lower curvatures, as well as left and right ones, while the latter considers only left and right ones.

3.1 Size-variable blocks

In this paper, the boundary verification function of the super-ellipse model is used to obtain a location $p(x, y)$ on the ellipse that is apart from some *division* angle θ_{Div} (Motani, 2001). In general, the shape of the super-ellipse model is determined by its parameters. In Figure 2, r_x represents the minor axis, r_y denotes the major axis, and θ is an angle between 0 and 2π.

Figure 2 Super-ellipse model



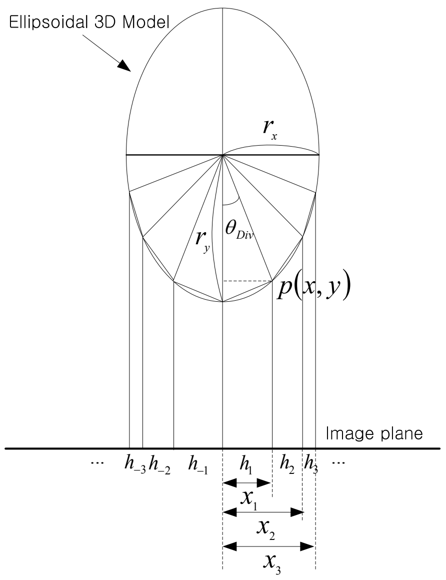
Equation (2) represents the super-ellipse model. In (2), $(x(\phi), y(\phi))$ means the position of the model and expresses the distance from the centre of the model using the sine and cosine functions. The shape of the super-ellipse model is determined as an ellipse when the value of n is 1.

$$S_{n,r_x,r_r} = \left(x(\phi),\, y(\phi)\right) \tag{2}$$

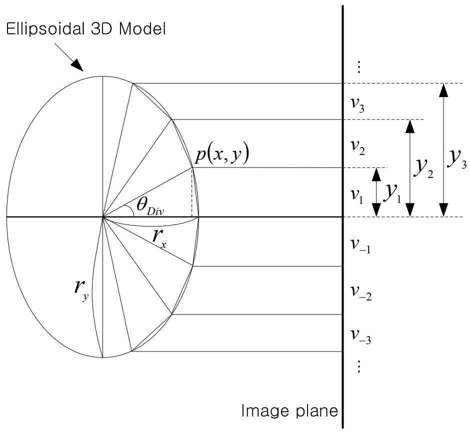
$$x(\phi) = r_x \cos^n(\phi),\ y(\phi) = r_y \sin^n(\phi)$$

where $0 < n < 1\,(\textit{rectangle}),\ n = 1\,(\textit{ellipse}),\ n > 1\,(\textit{diamond})$

Figure 3 Size-variable blocks, (a) width of block (b) height of block



(a)



(b)

The block sizes of 2D images corresponding to the 3D ellipsoidal model are defined in (3) and (4). Block size can be calculated by projecting the extracted point $p(x, y)$ on the ellipse into the 2D image plane. In (3), *BW* denotes the sizes of blocks that consider the left and right curvatures of the face model. In (4), *BH* denotes the sizes of blocks that reflect the top and bottom curvatures of the face model. θ_{Div} represents the division angle that equally partitions the 3D model, and n is the index of row and column of the blocks in the 2D image. h_n and v_n denote the size of each block projected into the 2D image. x_n and y_n represent the sum of sizes of the whole blocks projected.

$$\begin{cases} BW = h_n \\ h_1 = x_1 \\ h_n = x_n - x_{n-1}\ (n \ge 2) \\ x_n = r_x \cos\left[\dfrac{3\pi}{2} \times \left(n \times \theta_{Div}\right)\right] \end{cases} \tag{3}$$

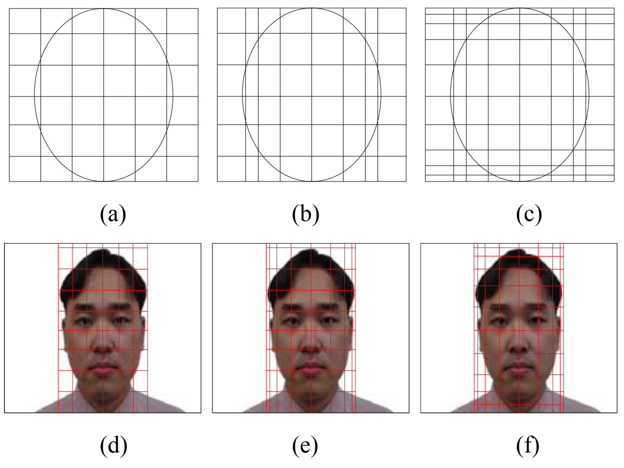
$$\begin{cases} BH = v_n \\ v_1 = y_1 \\ v_n = y_n - y_{n-1}\ (n \ge 2) \\ y_n = r_y \sin\left[n \times \theta_{Div}\right] \end{cases} \tag{4}$$

where $0 < \theta_{Div} < \dfrac{\pi}{4},\ 1 \le |n| \le \dfrac{\pi}{2\theta_{Div}}$

Figure 3 illustrates the size-variable blocks generated when the ellipsoidal model is divided at the same angle θ_{Div}. In Figure 3, when index n denoting the row and column of blocks has a value less than 0, it represents blocks that are positioned left of or lower than the centre of the 2D image.

Figure 4 illustrates how to divide images using the existing block matching and proposed methods. In Figure 4, (a) and (d) present results of partitioning images with fixed-sized blocks, (b) and (e) show results with size-variable blocks of the cylindrical model, and (c) and (f) show results with size-variable blocks of the ellipsoidal model.

Figure 4 Comparison of block sizes, (a) fixed-sized block (b) block of cylindrical model (c) block of ellipsoidal model (d) fixed-sized block (e) block of cylindrical model (f) block of ellipsoidal model (see online version for colours)



3.2 *Block similarity function*

In this paper, colour and texture features are used to match blocks because they effectively represent the correlation between images. These features are computed block-wise and utilised to define the block similarity with weighting factors that show the degree of contribution of each feature. In general, RGB colour space has widely been used to represent a colour image but is not efficient in image processing because it uses three colour components and is sensitive to illumination (Qazi et al., 2011). We thus adopt the YIQ colour model that is robust to brightness change. The equation transforming RGB colour space into YIQ colour space is defined in (5).

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.30 & 0.59 & 0.11 \\ 0.60 & -0.27 & -0.32 \\ 0.21 & -0.52 & 0.31 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{5}$$

In (6), $M_{colour}(B_i)$ represents the colour similarity for the i^{th} block and has a value between 0 and 1. As its value decreases, corresponding block similarity increases. $C_{B_i}^{\theta_1}$ denotes the colour feature for the i^{th} block of the image at an angle of θ_1. $C_{\max}$ is the maximum value of colour feature C. M and N are the width and height of a block, respectively.

$$M_{colour}(B_i) = \left| \frac{C_{B_i}^{\theta_1} - C_{B_j}^{\theta_2}}{C_{\max}} \right| \tag{6}$$

$$C_{B_i}^{\theta_1} = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I(x+i, y+j)$$

$$\text{where } 0 \le M_{colour}(B_i) \le 1$$

Texture features are used to resolve matching errors produced in homogeneous colour areas. For the texture feature, we use Gabor wavelet coefficients (Manjunath and Ma, 1996). The Gabor coefficients are acquired by applying the convolution operation of a Gabor wavelet kernel into an image, and they are characteristics that effectively reflect the texture of the image. In (7), $M_{texture}(B_i)$ denotes the similarity of texture for the i^{th} block, and $\Psi_{m\theta}(x, y)$ is an index vector that represents the magnitude and direction of the kernel. m denotes the index of magnitude and determines the frequency of the sine curve. θ denotes the direction index and determines the direction of the sine curve.

$$M_{texture}(B_i) = \left| \frac{T_{B_i}^{\theta_1} - T_{B_j}^{\theta_2}}{T_{\max}} \right| \tag{7}$$

$$T_{B_i}^{\theta_1} = \sum_{x=i}^{M-1} \sum_{y=j}^{N-1} I(x+i, y+j) \cdot \Psi_{m\theta}(x, y)$$

$$\Psi_{m\theta}(x, y) = Real\ part + i \cdot Imaginary\ part$$

$$= \frac{(k_m \cos k_\theta)^2 + (k_m \sin k_\theta)^2}{\sigma^2}$$

$$\cdot \exp\left\{ \frac{\left[(k_m \cos k_\theta)^2 + (k_m \sin k_\theta)^2 \right] \cdot \left[x^2 + y^2 \right]}{2\sigma^2} \right\}$$

$$\cdot \left[\cos(x k_m \cos k_\theta + y k_m \sin k_\theta) \right.$$

$$\left. + i \cdot \sin(x k_m \cos k_\theta + y k_m \sin k_\theta) - \exp\left\{ -\frac{\sigma^2}{2} \right\} \right]$$

$$\text{where } 0 \le M_{texture}(B_i) \le 1$$

The block similarity function, which is used as the matching metric between blocks, is defined in (8). In (8), $SF(i, j; u, v)$ means the matching similarity that reflects both colour and texture features and has a value between 0 and 1. α and β denote the degree of importance of each texture, and they have values of 0.4 and 0.6, respectively. (i, j) represents the position of the basis block, and (u, v) denotes the disparity between the basis block and a candidate block. The disparity vector of the best matched blocks is denoted by (u^*, v^*), where u^* is the disparity in the x axis and v^* the disparity in the y axis.

$$d(i, j) = (u^*, v^*) \tag{8}$$

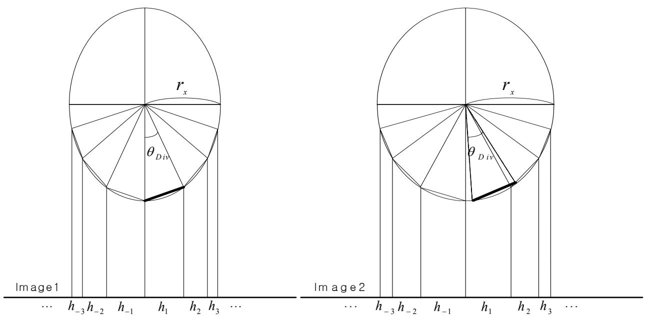
where $SF(i, j; u, v)$ is maximised

$$SF(i, j; u, v) = 1 - \{\alpha \cdot M_{colour}(i, j; u, v) + \beta \cdot M_{texutre}(i, j; u, v)\}$$

$$\text{where } 0 \le \alpha, \beta \le 1, \alpha + \beta = 1, 0 \le SF(i, j; u, v) \le 1$$

Figure 5 shows the process of generating the width of candidate block for the width h_1 of the first basis block. Figure 5 shows how the width of candidate blocks is computed by projecting into the 2D image points that are θ_{div} intervals away in the ellipsoidal model. The height of candidate blocks is calculated likewise.

Figure 5 Generating candidate blocks



r_x
θ_{Div}
Image1
$\cdots$ h_{-3} h_{-2} h_{-1} h_1 h_2 h_3 $\cdots$
r_x
θ_{Div}
Image2
$\cdots$ h_{-3} h_{-2} h_{-1} h_1 h_2 h_3 $\cdots$

When the displaced block similarity is computed, the number of candidate blocks made for the basis block is obtained as in (9) and the size of candidate blocks is calculated by (3) and (4).

$$NCB(B_i) = NCBW(h_n) \times NCBH(v_n) \tag{9}$$

$$\text{where} \begin{cases} NCBW(h_n) = r_x \left[1 - n \times \left(\dfrac{2\theta_{Div}}{\pi} \right) \right] \\ NCBH(v_n) = r_y \left[1 - n \times \left(\dfrac{2\theta_{Div}}{\pi} \right) \right] \end{cases}$$

In (9), $NCB(B_i)$ denotes the number of candidate blocks generated for the i^{th} basis block with width and height that are h_n and v_n. $NCB(h_n)$ and $NCB(v_n)$ denote the number of width and height of candidate blocks, respectively. As observed in (9), the number of generated candidate blocks decreases as the basis block gets away from the centre.

4 Image mosaicing

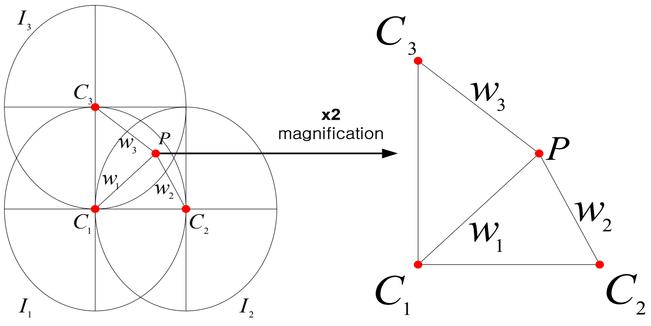
In this paper, image stitching is used to generate a facial texture image, and the cross-dissolve technique is adopted to fuse images naturally (Whitaker, 2000; Lensch et al., 2001). The final texture can be generated by blending colour values of pixels positioned at overlapping areas according to the corresponding weighting factor. To generate more natural textures, we use the weight that reflects the distance between the overlapped area and the input image.

4.1 Image stitching

Equation (10) defines the facial texture image that is generated through image mosaicing. In (10), T denotes the texture image, P is a pixel that exists in an overlapped area, and $w_i(P)$ means the weighting factor given to the i^{th} image for P, respectively. For non-overlapped areas, P values in the corresponding I_i are used. For overlapped areas, as in Figure 6, image stitching is performed by applying the weighting factor $w_i(P)$ in (11) to image I_i.

$$\begin{aligned}&IF\ (P\ unique\ in\ I_i)\\&\quad T = I_i(P)\\&ELSE\\&\quad T=\sum_{i=1}^{n} w_i(P)\cdot I_i(P)\end{aligned} \tag{10}$$

Figure 6 Weighting factor assigned to overlapped area (see online version for colours)



4.2 Weighting factors

When images are stitched, weighting factors are assigned considering overlapped areas. Figure 6 shows that the value of some pixel P is determined by applying a weighting factor that is inversely proportional to the distance of the centre position C_i. In this paper, the weighting factor is defined as in (11), which is based on the inverse distance weighting (IDW) used in data interpolation (Gordon and Wixom, 1978). In (11), r is a factor that affects the determination of the weighting factor according to the distance and is here set to 2.

$$w_i(p)=\frac{d_i^{-r}}{\sum_{k=1}^{n} d_k^{-r}},\ d_i=\overline{C_iP}=\sqrt{\left(x_{c_i}-x\right)^2+\left(y_{c_i}-y\right)^2} \tag{11}$$

where $0 \le w_i \le 1,\ \sum_{k=1}^{n} w_k = 1$

4.3 Eliminating ghost effects

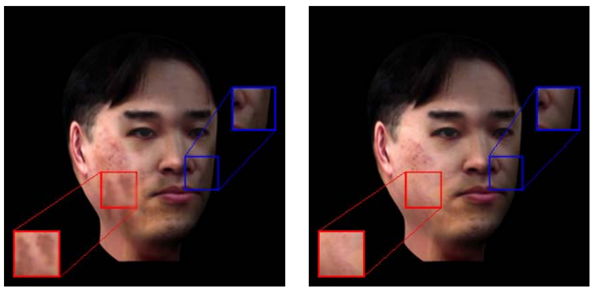
Although the suggested stitching method generates realistic facial textures, ghost effects exist in the boundaries of certain overlapped areas. To eliminate these ghost effects, we make the boundaries of images smoother and more realistic through the 3D convolution interpolation (Rocchini et al., 1999; Uyttendaele et al., 2001). The 3D interpolation in 2D space is defined as in (12). In (12), $C_{(i+k,j+1)}$ denotes the coefficients determined by sample data, and (h_x, h_y) is the sampling interval. int() makes the number in parentheses an integer.

$$I(x,y)=\sum_{k=-n}^{n}\sum_{l=-m}^{m} C_{(i+k,j+l)}\left(\frac{x-x_i}{h_x},\frac{y-y_i}{h_y}\right), \tag{12}$$

where $i=\operatorname{int}\left(\frac{x}{h_x}+0.5\right),\ j=\operatorname{int}\left(\frac{y}{h_y}+0.5\right)$

Figure 7(a) shows an image mapped into the 3D face model without eliminating ghost effects of generated facial texture, and Figure 7(b) is an image mapped into the 3D model after removing ghost effects. As Figure 7 clearly shows, eliminating ghost effects makes the 3D model more realistic.

Figure 7 Comparing boundaries of overlapped areas, (a) image with ghost effects (b) image without ghost effects (see online version for colours)



5 Experimental results

We captured test images with the Nikon Coolpix 2500 digital camera. The size of test images is normalised to 420 × 315, and the background areas are removed through preprocessing. We implemented the proposed algorithm using Microsoft Visual C++ compiler in Windows 2000. Figure 8 shows a 3D face model used in this paper, which includes 1,504 vertices and 846 meshes.

Figure 9 shows the face images captured at 45-degree intervals in the upper and lower sides, as well as the left and right sides. Figure 10 illustrates the generated facial texture images. Figure 10(a) shows the resultant facial texture made using the existing registration method, and Figure 10(b) shows the facial texture generated utilising the suggested ellipsoidal model-based registration method. As Figure 10 shows, the facial texture of the existing method contains blurred boundaries of overlapped areas and ghost effects.

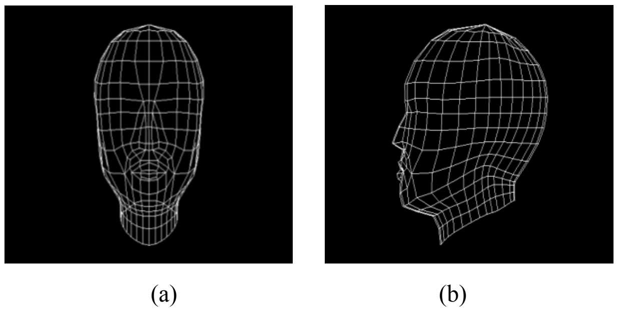


Figure 8 3D face model, (a) front view (b) profile view (see online version for colours)

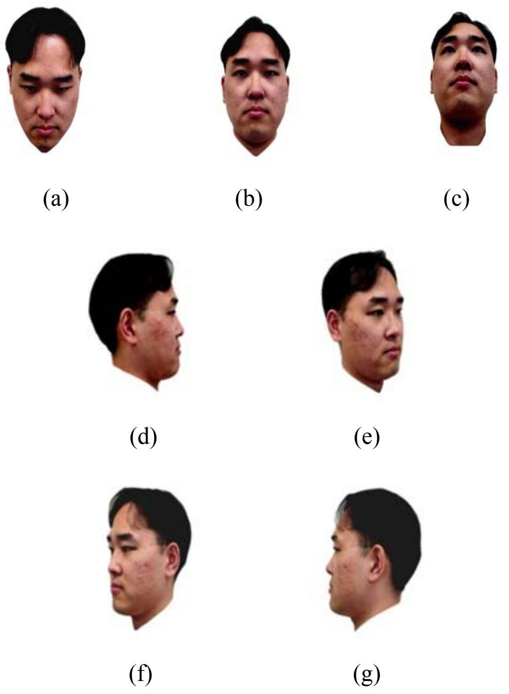
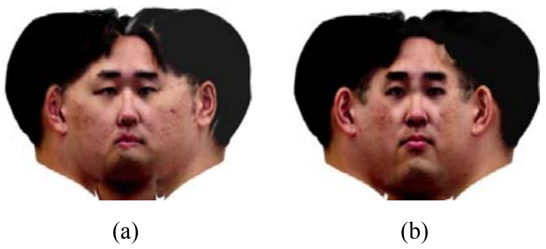


Figure 9 Test images, (a) upper side (45 degree) (b) front side (c) lower side (45 degree) (d) right side (90 degree) (e) right side (45 degree) (f) left side (45 degree) (g) left side (90 degree) (see online version for colours)

Figure 10 Facial texture generation, (a) facial texture (existing) (b) facial texture (proposed) (see online version for colours)



Tables 1, 2, and 3 show the number and magnitude of blocks generated when an image is divided at angles of 10, 15, and 30, respectively. These tables also contain the width and height of the candidate blocks considered in matching blocks.

Table 4 shows the mean squared errors (MSE) and computational complexity in matching blocks, which compares the performance of block matching according to divided angles. As Table 4 indicates, when the divided angle increases, the MSE value decreases, but the complexity increases. The matching accuracy gets low as the divided angle gets small because the discrimination of colour and texture features decreases. Considering the accuracy and computational complexity, we use a division angle of 15°.

Figure 11 illustrates the registration errors of resultant images obtained using the existing method (Lee and Choi, 2001) and the suggested algorithm. The image registration error is computed by comparing pixel values between the textures of original and registration images. In Figure 11, we can see that the registration error of the existing methods rapidly increases from the sixth image, while the registration error variance in the suggested method is relatively small, as it reflects the upper and lower curvatures of the image.

Table 1 The number of blocks and their magnitude ($r_x = 62$, $r_y = 112$, $\theta_{\text{Div}} = 30°$)

r_x	*62*	*Index*	x_n	h_n	y_n	v_n	$NCBW(h_n)$	$NCBH(v_n)$
r_y	112	1	31	31	56	56	41	75
θ_{Div} (degree)	30	2	53.69	22.69	96.99	40.99	21	37
θ_{Div} (radian)	0.5235	3	63	8.31	112	15.01	0	0

Table 2 The number of blocks and their magnitude ($r_x = 62$, $r_y = 112$, $\theta_{\mathrm{Div}} = 15°$)

r_x	*62*	*Index*	x_n	h_n	y_n	v_n	$NCBW(h_n)$	$NCBH(v_n)$
r_y	112	1	16.05	16.05	28.99	28.99	52	93
θ_{Div} (degree)	15	2	31	14.95	56	27.01	41	75
θ_{Div} (radian)	0.2617	3	43.84	12.84	79.20	23.20	31	56
		4	53.69	9.85	96.99	17.80	21	37
		5	59.89	6.19	108.18	11.19	10	19
		6	62	2.11	112	3.816	0	0

Table 3 The number of blocks and their magnitude ($r_x = 62$, $r_y = 112$, $\theta_{\mathrm{Div}} = 10°$)

r_x	*62*	*Index*	x_n	h_n	y_n	v_n	$NCBW(h_n)$	$NCBH(v_n)$
r_y	112	1	10.77	10.77	19.45	19.45	55	100
θ_{Div} (degree)	10	2	21.21	10.44	38.31	18.86	48	87
θ_{Div} (radian)	0.2617	3	31	9.79	56	17.69	41	75
		4	39.85	8.85	71.99	15.99	34	62
		5	47.49	7.64	85.80	13.80	28	50
		6	53.69	6.20	96.99	11.20	21	37
		7	58.26	4.57	105.25	8.25	14	25
		8	61.06	2.80	110.30	5.05	7	12
		9	62	0.94	112	1.70	0	0

To effectively compare the textures of Figure 10(a) and Figure 10(b), Figures 12 and 13 show the results of mapping the textures into the 3D facial model.

Table 4 Performance comparison according to division angles

θ_{Div}	*MSE*	*Complexity*
10°	9.02	17,470
15°	6.85	10,614
30°	5.26	3,852

Figure 11 Comparing the image registration error (see online version for colours)

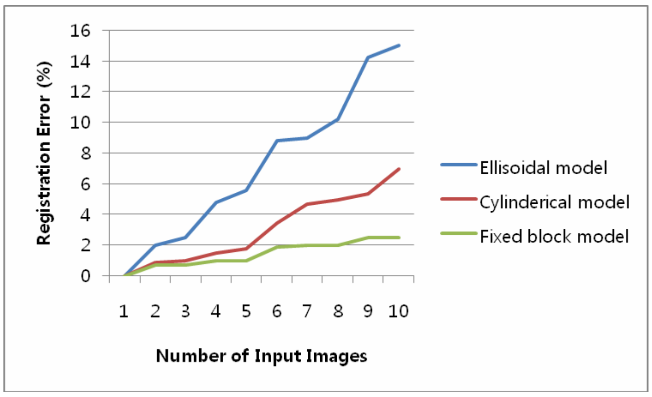


Figure 12 shows the texture mapping result of the existing fixed block-based method, and Figure 13 shows the result of the suggested ellipsoidal model-based method. The 3D facial textures generated by the existing method contain ghost effects near the boundaries of the overlapped areas; the overall structure of the facial contour is thus not natural. However, the 3D textures made using the suggested method do not include ghost effects and are natural and realistic. If a more detailed 3D face model is used, then more realistic facial textures could be obtained.

Figure 12 Texture mapping results of the existing fixed block-based method, (a) upper side (45°) front side lower side (45°) (b) right side (135°) right side (90°) right side (45°) (c) left side (45°) left side (90°) left side (135°) (see online version for colours)

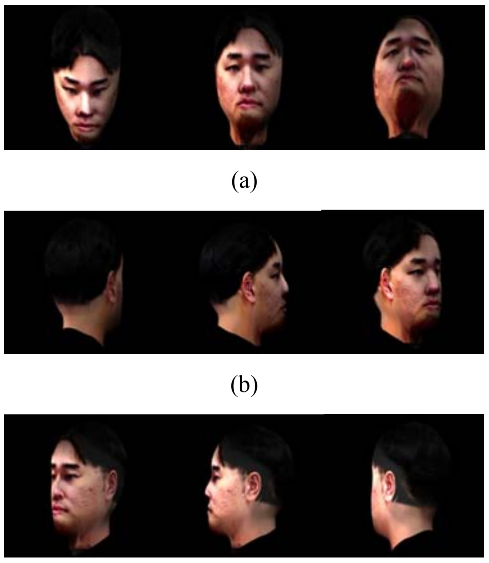
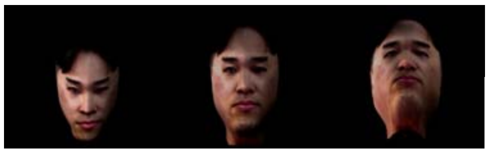
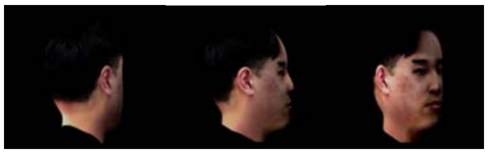


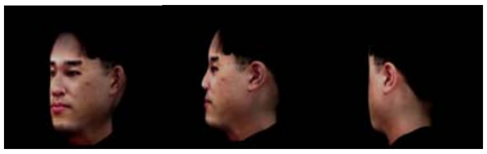
Figure 13 Texture mapping results of the suggested method, (a) upper side (45°) front side lower side (45°) (b) right side (135°) right side (90°) right side (45°) (c) left side (45°) left side (90°) left side (135°) (see online version for colours)



(a)



(b)



(c)

6 Conclusions

In this paper, we proposed a method to generate realistic and natural facial texture images using the registration algorithm based on the ellipsoidal model with size-variable blocks. To utilise the ellipsoidal model, the minor and major axes of an ellipse are first computed, and the images are subsequently matched block-wise while considering the curvature of the ellipsoidal model. When images are matched, matching errors between areas are resolved because image correlation and curvature are reflected. For image features, colour and texture are used and thus the intrinsic characteristics of the image are effectively considered. Generating a texture image from registered images uses the mosaicing technique. When stitching images, we adopt the cross-dissolve method to fuse them and use weighting factors inversely proportional to the distance from the centre of the image for overlapped areas. We also eliminate ghost effects existing in the boundaries of overlapped area such that we can generate realistic and natural textures.

We expect that the suggested method will be widely used in areas including face modelling and virtual reality, as it can generate natural 3D facial textures. For future work, we will apply the suggested method to various real applications to verify its practicability.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MEST) (No. 2011-0027715).

References

Chang, Y.J. and Chen, Y.C. (2002) 'Facial model adaptation from a monocular image sequence using a textured polygonal model', *Signal Processing: Image Communication*, Vol. 17, No. 5, pp.373–392.

Gordon, W.J. and Wixom, J.A. (1978) 'Shepard's method of metric interpolation to vivariate and multivariate interpolation', *Mathematics of Computation*, Vol. 32, No. 141, pp.253–264.

Ivanov, D., Lempitsky, V., Shokurov, A., Khropovand, A. and Kuzmin, Y. (2003) 'Creating personalized head models from image series', in *Proceedings of the International Conference on Computer Graphics and Vision*.

Lee, J-J. and Choi, H-I. (2001) 'An image registration algorithm based on cylindrical prototype model', in *Proceedings of the International Conference on Computational Science*, pp.37–43.

Lee, S.J., Park, K.R. and Kima, J. (2011) 'A SfM-based 3D face reconstruction method robust to self-occlusion by using a shape conversion matrix', *Pattern Recognition*, Vol. 44, No. 7, pp.1470–1486.

Lee, W-S. and Magnenat-Thalmann, N. (2000) 'Fast head modeling for animation', *Image and Vision Computing*, Vol. 18, No. 4, pp.355–364.

Lensch, H.P.A., Heidrich, W. and Seidel, H-P. (2001) 'A silhouette-based algorithm for texture registration and stitching', *Graphical Models*, Vol. 63, No. 4, pp.245–262.

Manjunath, B.S. and Ma, W.Y. (1996) 'Texture features for browsing and retrieval of image data', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 8, pp.837–842.

Motani, R. (2001) 'Estimating body mass from silhouette: testing the assumption of elliptical body cross-sections', *Paleobiology*, Vol. 27, No. 4, pp.735–750.

Pighin, F., Hecker, J., Lischinski, D., Szeliski, R. and Salesin, D. (1998) 'Synthesizing realistic facial expressions from photographs', in *Proceedings of the International Conference and Exhibition on Computer Graphics and Interactive Techniques*, pp.75–84.

Pighin, F.H., Szeliski, R. and Salesin, D. (2002) 'Modeling and animating realistic faces from images', *International Journal of Computer Vision*, Vol. 50, No. 2, pp.143–169.

Qazi, I-U-H., Alata, O., Burie, J-C., Moussa, A. and Fernandez-Maloigne, C. (2011) 'Choice of a pertinent color space for color texture characterization using parametric spectral analysis', *Pattern Recognition*, Vol. 44, No. 1, pp.16–31.

Rocchini, C., Cignoni, P., Montani, C. and Scopigno, R. (1999) 'Multiple textures stitching and blending on 3D objects', in *Proceedings of the Euro-graphics Workshop on Rendering*, pp.127–138.

Uyttendaele, M., Eden, A. and Szeliski, R. (2001) 'Eliminating ghosting and exposure artifacts in image mosaics', in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp.519–516.

Wang, C., Yan, S., Zhang, H. and Ma, W. (2005) 'Realistic 3D face modeling by fusing multiple 2D images', in *Proceedings of the International Multimedia Modeling Conference*, pp.139–146.

Whitaker, R.T. (2000) 'A level-set approach to image blending', *IEEE Transactions on Image Processing*, Vol. 9, No. 11, pp.1849–1861.