Original Article
International Journal of Fuzzy Logic and Intelligent Systems
Vol. 14, No. 2, June 2014, pp. 84-90
http://dx.doi.org/10.5391/IJFIS.2014.14.2.84

ISSN(Print) 1598-2645
ISSN(Online) 2093-744X

Three-Dimensional Face Point Cloud Smoothing Based on Modified Anisotropic Diffusion Method

Suryo Adhi Wibowo and Sungshin Kim
Department of Electrical Engineering, Pusan National University, Busan, Korea

ljfis

Abstract

This paper presents the results of three-dimensional face point cloud smoothing based on a modified anisotropic diffusion method. The focus of this research was to obtain a 3D face point cloud with a smooth texture and number of vertices equal to the number of vertices input during the smoothing process. Different from other methods, such as using a template D face model, modified anisotropic diffusion only uses basic concepts of convolution and filtering which do not require a complex process. In this research, we used 6D point cloud face data where the first 3D point cloud contained data pertaining to noisy x-, y-, and z-coordinate information, and the other 3D point cloud contained data regarding the red, green, and blue pixel layers as an input system. We used vertex selection to modify the original anisotropic diffusion. The results show that our method has improved performance relative to the original anisotropic diffusion method.

Keywords: Three-dimensional face point cloud, Smoothing, Modified anisotropic diffusion, Selecting vertices

1. Introduction

The current trend of face recognition research is toward greater use of three-dimensionl (3D). This trend is due to the rapid evolution of technologies that have enabled the development of image acquisition tools such as 3D scanners and the Microsoft Kinect sensor. In addition, the emphasis of research has moved towards 3D in order to improve performance accuracy of 2D face recognition under difficult conditions. To obtain highly accurate results, appropriate preprocessing steps are needed because their result will affect the results and processes of feature extraction and classification.

Mian et al.[1] have researched automatic 3D face detection, normalization, and recognition. They used data from the face recognition grand challenge (FRGC), which was taken from the Minolta Vivid 900/910 series sensor. The Cyberware^{TM} 3030PS laser scanner was used by Blanz and Vetter [2] to record 3D face data. It was also used as an interactive tool for filling holes and removing spikes. Unfortunately, the price of these data acquisition devices is prohibitive. Although the 3D face data has high resolution [2]; it requires a smoothing process in preprocessing. They used a median filter for the smoothing process. Li et al. [3, 4] used the Microsoft Kinect sensor, which is the cheapest hardware available for 3D face recognition. Resampling and symmetric filling was used for smoothing. Chen et al. [5]

Received: Jun. 2, 2014
Revised : Jun. 24, 2014
Accepted: Jun. 24, 2014

Correspondence to: Sungshin Kim
(sskim@pusan.ac.kr)

used geometric hashing and a sample training set of labeled images as their denoising method. Mean and median filtering are also used for mesh smoothing which has been applied to face data [6]; to get a smooth result, these methods require many iterations. However, to improve the results of the 3D point cloud texture in the preprocessing step, especially using RGB-D images, a researcher usually needs the assistance of a 3D face template model [2-4]. When a 3D face template model is used, a fitting method is absolutely necessary. Iterative closest point (ICP) is used to fit a 3D face point cloud to a 3D face template model [3], but this method requires a long computation time. In 3D face recognition [7, 8], an affine transform or homogeneous transform is used to replace the ICP method for fitting [4]. We emphasize that these methods are not very efficient.

Anisotropic diffusion is the method introduced by Perona and Malik [9]. This method is very useful because it is simpler than the others. Anisotropic diffusion can be implemented on 1D, 2D, and 3D data. Gerig et al. [10] used anisotropic diffusion for filtering MRI data where it has been applied to 2D and 3D image data with one or two channels [11, 12]. Linear anisotropic diffusion mesh filtering has also been investigated [13].

Anisotropic diffusion has also been used for geometric surface smoothing [14] and has applications in image processing [15].

In this paper, we propose a smoothing 3D face point cloud method based on modified anisotropic diffusion. We use input data from the Curtin Faces Database [4]. The data are derived from the Microsoft Kinect, which produced red, green, blue, and depth (RGB-D) data. These data have been processed into 6D data. Because the 3D point cloud data is noisy we used the relation between 2D data and 3D point cloud data to process it more easily. To denoise and smooth the 3D point cloud, we used the basic anisotropic diffusion method. However, we were required to modify the original anisotropic diffusion to increase performance because the original anisotropic diffusion could not compute perfectly in our case. Selected vertices have been used to detect the value from vertices.

The contributions of this research are as follows.

1) We have developed an easier way to process 3D face point cloud data that uses the relation between 2D images and 3D object point clouds.
2) We have developed a robust, modified smoothing method, where the output has a smooth texture and the same number of vertices as the input in the smoothing process. Our proposed method could be implemented with the Microsoft Kinect, which is a cheaply priced device for 3D object processing.

The paper is structured as follows: In Section 2, we present the method and approach. In Section 3, performance evaluation and results are described. In Section 4, we present our conclusions.

2. Method and Approach

Triangle mesh is one method that can represent a 3D object based upon vertices and face data. This method is simpler than the other; however, we must know about the relationship between vertices and faces. It also requires complex computation to process, if we only use vertices and faces as the information. We propose that the easiest way to process a 3D object point cloud is by using the relation between a 2D image and a 3D object point cloud. Figure 1 shows the result of each step of the cropping process for a 3D face point cloud.

2.1 Point Cloud Reconstruction

We used a data sample from the Curtin Faces Database [4], which has 307,200 rows and six columns of point cloud matrix data for each subject. For easy calculation, we should rearrange the point cloud matrix to 460 rows and 640 columns, where the first, second, and third columns contain x-, y-, and z-coordinate data, respectively. The fourth, fifth, and sixth columns contain data regarding the red, green, and blue layers, respectively. Eq. (1) illustrates the data in a 6D point cloud.

$$\begin{bmatrix} PC_{1.1} & \cdots & \cdots & PC_{1.307,200} \\ \vdots & & & \vdots \\ PC_{6.1} & \cdots & \cdots & PC_{6.307,200} \end{bmatrix}^T = \begin{bmatrix} A_1 \\ \vdots \\ A_6 \end{bmatrix}^T, \quad (1)$$

where $PC_{1.1} \cdots PC_{1.307,200}$, $PC_{6.1} \cdots PC_{6.307,200}$ are the first and last columns. It has length 307,200. $A_1 \cdots A_6$ pertains to the rearranged result of the point cloud to the matrix i, j where $i = 480$ and $j = 640$.

2.2 Skin Detection

For skin or face detection, we can use existing algorithms such as Lucas-Kanade, Viola-Jones, etc. However, in this experiment, we used a simple method to detect the face in 2D images. We used the YCbCr color format to separate the face from the background. We know that$A_k$is point cloud data, which has x-($k = 1$), y-($k = 2$), and z-coordinates($k = 3$),

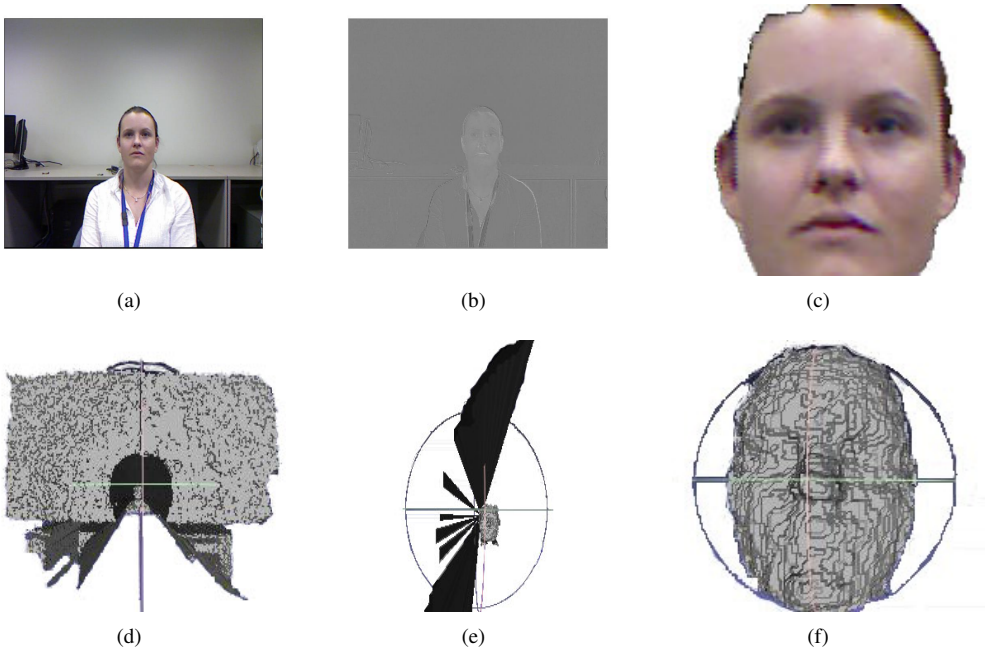

Figure 1. (a) RGB image, (b) chrominance red image, (c) cropped image, (d) 3D point cloud, (e) cropped 3D face point cloud before multiply with NaN and (f) cropped 3D face point cloud after multiply with NaN.

as well as red($k = 4$), green($k = 5$), and blue layer pixel information($k = 6$), where the sizes are 480 rows and 640 columns, respectively. Because it is equivalent to the red, green, and blue layer color format, we converted it to the YCbCr color format for skin detection. The chrominance red layer from the result of converting the color format has much more information than the other because skin colors are strongly influenced by red pixels.

2.3 3D Face Point Cloud Cropping

The chrominance red layer was converted to the monochrome format for easy cropping of 2D faces. Because the object and environmental conditions are always changing, an adaptive threshold should be used to obtain an optimal result for the black and white color format. To solve this problem, we used the Otsu method to adaptively calculate the threshold. Although the adaptive threshold was already used in this system, the result of this system has some noise. We used a median filter and selected an area to reduce its noise. We used a kernel size of 7×7 for the median filter. Although using the median filter caused closing and opening of pixels in the imageI_{mf}, the result still provides much better noise reduction. To remove undesired objects, we selected by area for each object which had the label lb before

$$I_{rn} = \begin{cases} Face, & \sum_{i,j} I_{mf}(lb) \geq th \\ 0, & \text{else} \end{cases} \tag{2}$$

where I_{rn} and I_{mf} are the image result from noise removal and the image result from the median filter, respectively. We used a threshold th of 5,000 to select the area. The $Face$ part could be determined if it met the requirements that have been defined before. The histogram was computed from I_{rn} and then cropping face part depended on information from the histogram. Eqs. (2), (3), and (4) describe cropping by using histogram information

$$P_1 = I_{rns1} \neq 0, \qquad I_{rns1} = \textstyle\sum_i I_{rn} \tag{3}$$

$$P_2 = I_{rns2} \neq 0, \qquad I_{rns2} = \textstyle\sum_j I_{rn} \tag{4}$$

$$P_{\max 1} = P_{\min 1} + \Delta_1, P_{\min 2} = P_2 + \Delta_2, P_{\max 3} = P_2 + \Delta_3 \tag{5}$$

wher eI_{rns1}, I_{rns2} are the sum of pixels by rows and columns in I_{rn}, respectively. After computation of the histogram, we must find the histogram's result that is not equal to zero. Maximum and minimum boundaries for cropping coordinates can be determined from these steps. $P_{\min 1}$, $P_{\min 2}$ is the minimum boundary, $P_{\max 1}$, $P_{\max 3}$ is the maximum boundary, and Δ_1,

Δ_2, Δ_3 are the thresholds for cropping. Δ_1, Δ_2, Δ_3 have the values are 125, -23, and 130, respectively. The variable I_{fc} represent the cropping face from 2D image that can be done after we solved Eq. (5).

Because $A_{4\cdots6}$ is related to $A_{1\cdots3}$, a 3D face point cloud could be extracted using Eq. (5) and we can call it Dc_k. An additional steps are required to capture the texture of a 3D face. The first step is multiplying it by the 2D image, which is the result from face cropping I_{fc},

$$D_{fcrop}\{k\} = \{Dc_k \cdot I_{fc}\}_{k=1}^{3} \tag{6}$$

Unfortunately, the result of the extraction process still has some noise. This happens because point cloud x-, y-, and z-coordinates are always sensitive to deformation. To solve this problem, we take a second step to calculate the absolute value that represents absolute face texture$\hat{d}$,

$$\hat{d} = \sqrt{D_{fcrop}\{1\}^2 + D_{fcrop}\{2\}^2 + D_{fcrop}\{3\}^2} \tag{7}$$

The result usually has range $0 \leq \hat{d} \leq 3,000$, depending on the distance when we acquire the data.

$$idx_{face} = \begin{cases} Face, & 100 \leq \hat{d} \leq 1,000 \\ 0, & \text{else} \end{cases} \tag{8}$$

Eq. (8) describes idx_{face}, the index location of 3D face point cloud. Now we have the rough 3D face point cloud $D_{roughface}$, because we have the index face information.

In $D_{roughface}$, there are some minor noises that can cause strange shapes on the 3D face point cloud. To remove this noise, we suggest following Eqs. (9) and (10).

$$D_c^k = \{\alpha \cdot O\}_{k=1}^{3} \tag{9}$$

$$D_c^k\left(idx_{face}\right) = \{D_{roughface}\left(idx_{face}\right)\}_{k=1}^{3} \tag{10}$$

where O is the matrix that has same size as the input system. α is the constant that should be added to remove noise. We used α equal to NaN. This is the key to our research. Because is NaN implemented in our system, it can eliminate the noise.

2.4 3D Face Point Cloud Smoothing

Because a 3D face point cloud has nonlinear characteristics, the anisotropic diffusion method has been used to smooth it. Anisotropic diffusion extends isotropic diffusion with a nonlinear term limiting diffusion across boundaries defined by a large gradient in image intensity. In order to enable the isotropic diffusion to preserve features, Perona and Malik [7] modified it.

$$AD = div(c(x,y,t)\cdot\nabla I) = c\left(x,y,t\right)\cdot\Delta I + \nabla_c \cdot \nabla I \tag{11}$$

Eq. (11) describe anisotropic diffusion, where div is the divergence operator with ∇, Δ is the gradient and Laplacian operator with respect to the space variables [7]. Eq. (11) also can be defined as

$$I_s^{t+1} = I_s^t + \lambda \cdot \sum_{p\in\eta_4(s)} g\left(I_p^t - I_s^t\right)\cdot\left(I_p^t - I_s^t\right) \tag{12}$$

(12) where I is the image, s and p are pixel positions, $\eta_4(s)$ is the 4-neighborhood of the pixel at s, λ is a scalar controlling the rate of diffusion, t is a discrete time step, and $g(x)$ is an edge stopping function. Perona and Malik [7] propose $g(x)$ that has two functions.

$$g_1(x) = \frac{1}{1+\frac{x^2}{\sigma^2}} \tag{13}$$

$$g_2(x) = e^{-\left(\frac{x^2}{\sigma^2}\right)} \tag{14}$$

where σ is a parameter controlling how large a change in pixel intensity is considered an edge [16].

In our system, we computed 3D face data like 2D images, so I in Eq. (12), can be replaced by D_c^k. Because the z-coordinate has a strong influence on the texture of a 3D face point cloud, we only processed it using anisotropic diffusion. Unfortunately, because of the cropping process, there are some point cloud values that could not be computed perfectly using anisotropic diffusion. To solve this challenge, we modified the anisotropic diffusion by

$$\begin{aligned} D_{c,s}^{3,t+1} &= D_{c,s}^{3,t} + xr, \\ xr &= \left[\lambda \cdot \sum_{p\in\eta_4(s)} g\left(D_{c,p}^{3,t} - D_{c,s}^{3,t}\right)\cdot\left(D_{c,p}^{3,t} - D_{c,s}^{3,t}\right)\right]\cdot VS \end{aligned} \tag{15}$$

$$VS(0) = \left\{\lambda \cdot \sum_{p\in\eta_4(s)} g\left(D_{c,p}^{3,t} - D_{c,s}^{3,t}\right)\cdot\left(D_{c,p}^{3,t} - D_{c,s}^{3,t}\right)\right\}_{p,s=NaN} \tag{16}$$

where VS is selected vertices and $VS(0)$ is the VS that has NaN value, and convert it equal to zero. For original anisotropic diffusion, we used public code on mathwork [17].

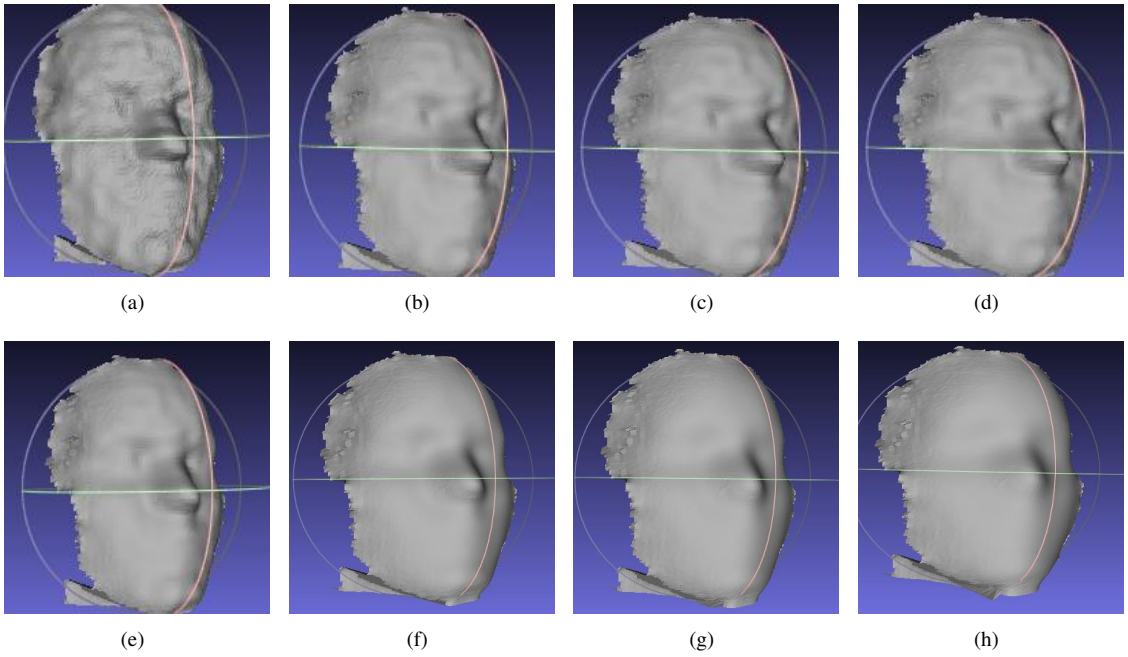

Figure 2. The 3D face point cloud smoothing result arranged by the number of iterations. (a) using 2 iterations, (b) using 4 iterations, (c) using 6 iterations, (d) using 8 iterations, (e) using 10 iterations, (f) using 20 iterations, (g) using 30 iterations, (h) using 40 iterations.

3. Performance Evaluation and Result

We tested the proposed method using data from the Curtin Faces database. In this system, we propose three scenarios to investigate the method. The first scenario is a performance comparison of smoothing using five different angles as an input. The second scenario is a performance comparison of smoothing using five different faces, but the same angle as an input. The last scenario is to investigate the performance of smoothing by iteration. The processes were carried out on a 2.4 GHz Intel core i3 processor with 2 GB memory.

3.1 Performance Comparison

In this experiment, we compared our proposed method and the original anisotropic diffusion method. In the first scenario, the subject was one female face posed at various angles. The angles that we used in this experiment were $0°$, $60°$, $-60°$, $90°$, and $-90°$. Table 1 shows the simulation result for $0°$, $60°$, $-60°$, $90°$, and $-90°$. From this table, modified anisotropic diffusion substantially improves the original anisotropic diffusion performance at all angles. Our proposed method can maintain the same number of vertices input in the smoothing process. The computation time of our modified anisotropic diffusion and the original anisotropic diffusion is almost the same; there is no significant difference between the methods.

In the second scenario, we used two females and one male face, all posed at the same angle. We used a $0°$ angle in this experiment. Table 2 shows that modified anisotropic diffusion also can maintain the same number of vertices input in the smoothing process. Both methods have almost the same computation time; there is no significant difference between our modified anisotropic diffusion and the original anisotropic diffusion.

3.2 Iteration Effect

We also investigate the influence of iteration on smoothing process. We used the follow number of iterations at each stage of our experiment: 2, 4, 6, 8, 10, 20, 30, and 40. Figure 2 shows the results from our experiment. From these results, we can see that when the number of iterations was less than 10, the result still left some noise in the texture of 3D face point cloud. A smooth result was only reached when the number of iterations exceeded 10. Unfortunately, these smooth results could cause the disappearance of some features in the 3D face point cloud.

The number of iterations that produced smooth texture in the 3D face point cloud while preserving its features is 10. In 10 iterations, we can still can see features from the 3D face point cloud such as the eye, nose, and mouth regions.

Table 1. Results of comparison methods with different angles

Data	Angles	Number of vertices			Computation Time (s)	
		Input	Result of each method		Result of each methods	
			MA	A	MA	A
Person1	0°	19,262	19,262	12,169	0.488	0.458
Person1	90°	20,061	20,061	11,506	0.486	0.444
Person1	60°	20,633	20,633	11,386	0.481	0.439
Person1	−60°	21,350	21,350	13,162	0.527	0.436
Person1	−90°	20,051	20,051	11,394	0.517	0.435

MA, modified anisotropic diffusion; A, anisotropic diffusion.

Table 2. Results of comparison methods with different faces

Data	Number of vertices			Computation Time (s)	
	Input	Result of each method		Result of each method	
		MA	A	MA	A
Person1	19,262	19,262	12,169	0.488	0.458
Person2	16,291	16,291	8,138	0.521	0.441
Person3	21,532	21,532	12,821	0.461	0.437

MA, modified anisotropic diffusion; A, anisotropic diffusion.

4. Conclusion

We have presented a modified anisotropic diffusion for smoothing a 3D face point cloud. The key idea of our research is how to smooth 3D face point clouds, the results of which must have the same shape as the original input smoothing and also have the same number of vertices. We have demonstrated that our modification method is more robust than the original anisotropic diffusion. Our method maintains the same shape and number of vertices as the original input in the smoothing process.

Our method is more robust than the original anisotropic diffusion because we modified it by selecting vertices which have values that could not be computed perfectly in anisotropic diffusion. Different faces and different angles in the 3D face point cloud did not influence our system. Our result still has the same shape and number of vertices. There are also no significant differences between the computation times of our modified anisotropic diffusion and the original anisotropic diffusion. The optimal number of iteration is 10 because it will produce a smooth result and preserve the appearance of 3D face point cloud features.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgments

This work was supported by BK21PLUS, Creative Human Resource Development Program for IT Convergence and supported by a 2-Year Research Grant of Pusan National University.

References

[1] A. Mian, M. Bennamoun, and R. Owens, "Automatic 3D face detection, normalization and recognition," in *Proceedings of the 3rd International Symposium on 3D Data Processing, Visualization, and Transmission*, Chapel Hill, NC, June 14-16, 2006, pp. 735-742. http://dx.doi.org/10.1109/3DPVT.2006.32

[2] V. Blanz and T. Vetter, "Face recognition based on fitting a 3D morphable model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, pp. 1063-1074, Sep. 2003. http://dx.doi.org/10.1109/TPAMI.2003.1227983

[3] B. Y. L. Li, A. S. Mian, W. Liu, and A. Krishna, "Using Kinect for face recognition under varying poses, expressions, illumination and disguise," in *IEEE Workshop on Applications of Computer Vision*, Tampa, FL, January 15-17, 2013, pp. 186-192. http://dx.doi.org/10.1109/WACV.2013.6475017

[4] B. Y. L. Li, W. Liu, S. An, and A. Krishna, "Tensor based robust color face recognition," in *Proceedings of the 21st International Conference on Pattern Recognition*, Tsukuba, Japan, November 11-15, 2012, pp. 1719-1722.

[5] Y. L. Chen, H. T. Wu, F. Shi, X. Tong, and J. Chai, "Accurate and robust 3D facial capture using a single RGBD camera," in *Proceedings of the IEEE International Conference on Computer Vision*, Sydney, Australia, December 1-8, 2013, pp. 3615-3622. http://dx.doi.org/10.1109/ICCV.2013.449

[6] H. Yagou, Y. Ohtake, and A. Belyaev, "Mesh smoothing via mean and median filtering applied to face normals," in *Proceedings of the Geometric Modeling and Processing*, Wako, Japan, July 10-12, 2002, pp. 124-131. http://dx.doi.org/10.1109/GMAP.2002.1027503

[7] C. M. Ma, S. H. Yoo, and S. K. Oh, "Design of face recognition algorithm based optimized pRBFNNs using three-dimensional scanner," *Journal of Korean Institute of Intelligent Systems*, vol. 22, no. 6, pp. 748-753, Dec. 2012. http://dx.doi.org/10.5391/JKIIS.2012.22.6.748

[8] S. H. Choi, S. Cho, and S. T. Chung, "Improvement of face recognition speed using pose estimation," *Journal of Korean Institute of Intelligent Systems*, vol. 20, no. 5, pp. 677-682, Oct. 2010. *http://dx.doi.org/10.5391/JKIIS.2010.20.5.677*

[9] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629-639, Jul. 1990. http://dx.doi.org/10.1109/34.56205

[10] G. Gerig, O. Kubler, R. Kikinis, and F. A. Jolesz, "Nonlinear anisotropic filtering of MRI data," *IEEE Transactions on Medical Imaging*, vol. 11, no. 2, pp. 221-232, Jun. 1992. http://dx.doi.org/10.1109/42.141646

[11] H. Jang, H. Ko, Y. Choi, Y. Han, and H. Hahn, "A new face tracking method using block difference image and Kalman filter in moving picture," *Journal of Korean Institute of Intelligent Systems*, vol. 15, no. 2, pp. 163-172, Apr. 2005

[12] S. K. Oh, S. H. Oh, and H. K. Kim, "Design of three-dimensional face recognition system using optimized PRBFNNs and PCA: comparative analysis of evolutionary algorithms," *Journal of Korean Institute of Intelligent Systems*, vol. 23, no. 6, pp. 539-544, Dec. 2013. http://dx.doi.org/10.5391/JKIIS.2013.23.6.539

[13] G. Taubin, "Linear anisotropic mesh filtering," IBM Research Report RC-22213. Available http://mesh.brown.edu/taubin/pdfs/Taubin-ibm22213.pdf

[14] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher, "Geometric surface smoothing via anisotropic diffusion of normals," in *Proceedings of the IEEE Visualization*, Boston, MA, November 1, 2002, pp. 125-132. http://dx.doi.org/10.1109/VISUAL.2002.1183766

[15] J. Weickert, *Anisotropic Diffusion in Image Processing*, Stuttgart, Germany: B.G. Teubner, 1998. Available http://www.lpi.tel.uva.es/muitic/pim/docus/anisotropic_diffusion.pdf

[16] T. R. Jones, "Feature preserving smoothing of 3D surface scans," M.S. thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[17] D. Lopez, "Anisotropic diffusion (Perona & Nalik)," Available http://www.mathworks.com/matlabcentral/fileexchange/14995-anisotropic-diffusion-perona-malik-

Suryo Adhi Wibowo received his B. Eng. Degree and M. Eng. degree in Electrical Engineering Majoring Telecommunication Engineering from Telkom Institute of Technology, Indonesia, in 2009 and 2012, respectively. He is currently a Ph.D. candidate at Department of Electrical and Computer Engineering, Pusan National University, Korea. His research interests include computer vision, intelligent system, pattern recognition and signal analysis.
E-mail: suryo@pusan.ac.kr.

Sungshin Kim received his B.S. and M.S. degrees in Electrical Engineering from Yonsei University, Korea, in 1984 and 1986, respectively, and his Ph.D. degree in Electrical Engineering from the Georgia Institute of Technology, USA, in 1996. He is currently a professor at the Electrical Engineering Department, Pusan National University. His research interests include fuzzy logic controls, neuro fuzzy systems, neural networks, robotics, signal analysis, and intelligent systems.
E-mail: sskim@pusan.ac.kr