碩士學位論文

단일 이미지로부터 3D 얼굴을 복원하기 위한 안드로이드 앱 설계에 대한 연구

A Study of an Android Application Design for 3D Face Reconstruction from a Single Image

國民大學校 大學院

電子工學科

Truong Huu Phuc

2012

碩士學位論文

단일 이미지로부터 3D 얼굴을 복원하기 위한 안드로이드 앱 설계에 대한 연구

A Study of an Android Application Design for 3D Face Reconstruction from a Single Image

指導敎授 鄭 求 珉

이 論文을 碩士學位 請求論文으로 提出함

2012年 12月

國民大學校 大學院

電子工學科

Truong Huu Phuc

2012

Truong Huu Phuc 의
碩士學位 請求論文을 認准함

2012年 12月

審査 委員長 정 경 훈 ㊞
審査 委員 鄭 求 珉 ㊞
審査 委員 文 燦 宇 ㊞

國民大學校 大學院

Abstract

In this thesis, a fast and effective method for 3D Face Reconstruction based on Canonical Correlation Analysis (CCA) Algorithm and its application in Smartphone are designed and implemented. The modeling step that creates correlation parameters is built based on 400 pairs of training images from database. These parameters are used as a criterion of 3D face reconstruction from frontal image on Smartphone.

Spherical harmonics are used as illumination basis to handle general light conditions. In order to deal with large dimensionality data, some prior works suggested principle component analysis (PCA) using the depth information in cylindrical coordinates, or Independent Component Analysis (ICA). This thesis exploited N-mode SVD based technique to handle high-order tensor due to its easy-to-use and convenient characteristic. N-mode SVD significantly reduced the computational burden in higher-order tensor's calculations.

The image got from Smartphone is processed to crop frontal face and identify eyes, mouth's position. Then, the alignment is applied to the frontal image to guarantee that the mark-points of the image are in the same position with ones in the model. The Canonical Correlation Analysis (CCA) algorithm is utilized to define a mapping between feature points of frontal face image and its corresponding 3D face images in modeling. The modeling is executed in Matlab with 2D images and its corresponding 3D ones from FRGC 2.0 database. The modeling result, the mapping, helps reconstructing 3D face from the single frontal image in Smartphone. Finally, the texture map are generated to insert to the 3D model result to give a better performance of reconstruction. Because the frontal image is any image got in real life not only from the database, we do not have a result 3D image

to compare. Therefore, we just prove and perform the reconstruction by visualizing the result as a 3D image in Smartphone. Besides, we also mention the experimental result using database in [14] to confirm the accuracy of this application.

The proposed method can effectively train a 3D face model and take a short time to reconstruct a 3D image. The computing strong point is performed clearly in the Android Smartphone application.

Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

Conversion from 2D image to corresponding 3D form is a significantly important researching field of computer vision and computer graphics. It is because all of objects in the real world, real life are 3-dimension ones, or in the other words, the world is composed by 3D objects.

3D model reconstruction of human face, a branch of 3D conversion also has a wide application ranging from 3D animation to video conferencing. 3D face reconstruction can be applied for animation by Model-based method [7] [8], or based on the Video method [6]. Moreover, it is also utilized to recognize human face by extracting 3D geometry information of a face and generating virtual samples by rotating the result 3D face model [9] [10] [11].

Figure 1.1: Reconstructed 3D shape for face recognition

It is a complicated challenge to reconstruct 3D shape of human face from 2D image because it involves many parameters to estimate and process, such as pose information, light condition, surface characteristics, etc. These parameters are combined by complex nonlinear equations to represent an image, thus it is difficult to find and estimate them to obtain an accurate 3D

reconstruction. Up to now, there are a number of techniques and algorithms proposed to solve this problem. However, the researches only handle one or some of the parameters with the results which still far from satisfactory. In order to estimate the pose parameter, Choi et al. [2] proposed a method of using Expectation Maximization (EM) Algorithm based on weak perspective projection by calculating the sum of the posterior probabilities of all the 3D feature points; a combination of EM Algorithm and a 3D face shape model was suggested in [3] [4] to deal with a more various pose case. To find surface characteristics, some methods were proposed such as geometric stereo [12], photometric stereo [13].

1.2 Previous works

Most previous approaches often realize the reconstruction from multiple images. Stereo vision method [12] [13] infers information on the 3D structure and the distance of a scene from two (or more) images taken from two (or more) camera from different viewpoints. Firstly, it is necessary to find the correspondences of each image pixel in different cameras, and then 3D structure is built from these correspondences. The correspondences include intrinsic and extrinsic parameters. The first one is the mapping of an image point from one camera to pixel coordinates in each other camera, and the second one describes the relative position and orientation of the images that built based on the rotation, translation and scale properties. An approaches for textured 3D face reconstruction based on two images taken at any defined angle was proposed in [16]. This technique uses 35 feature points that was marked manually of the images to create the analytical coordinates. By utilizing orthogonal projections with known angles and characteristics of cameras, the 3D coordinates of these

feature points was generated. Then, missing vertices are produced by using Kriging interpolation method. Before constructing the realistic 3D shape, the view independent or view dependent texture map was generated based on triangulation of one face' s side. These methods can easily reconstruct 3D coordinates of an human face; however, it requires a restricted condition of hardwares (cameras) and the result is still limited.

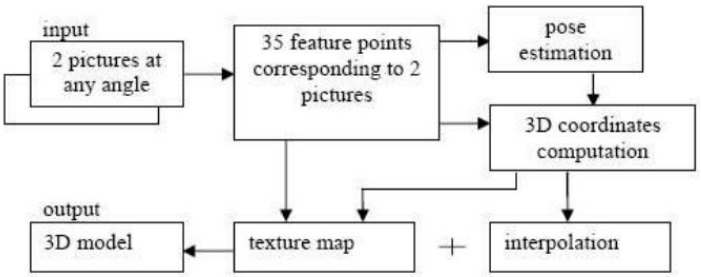


Figure 1.2: Overview of 3D face reconstruction using two pictures

Another approach is based on statistical models of objects[4] [8] [10] [11]. This model-based approach imposes model constraints to rule out the ambiguity and to guarantee a unique solution. This method can give a good result of 3D reconstruction based on the statistical model learned from training samples. Another advantage of this model-based technique is that only one single image is used to realize the reconstruction. That method is applied in many areas, such as face recognition [10] or image relighting. Another method described in [15] to reconstruct 3D face model from a single frontal face. Firstly, the alignment is utilized to locate a frontal face and 83 facial features points from the input 2D image. A 3D face model database which includes 3D laser scanned heads is needed to do the reconstruction. The geometry of a 3D face model is represented by a shape vector that consists of coordinates of its vertices. Because the facial feature points of different faces correspond to their position, PCA (Principle Component Analysis) Algorithm is employed to get shape representation of

the face by the primary components. Because the X, Y coordinates of feature points in the frontal image are different from ones in the 3D face model in database, to guarantee an accurate reconstruction result, they are forced to align to the coordinates of feature points in 3D face models. For the non-feature point's vertices, an interpolation is employed to complete the 3D face reconstructing.

A fast model based mapping technique is proposed in [23] is to find a mapping near infrared image tensor space to 3D tensor space by using statistical learning. In the learning step, a canonical correlation analysis (CCA) based multi-variate mapping from near infrared image to 3D image is trained from a given training set. This technique is simple, fast and easy to use but can give a reliable result, so we decided to employ its strong points. In our work, we create a CCA based mapping between tensor space of face surface, specifically we use spherical harmonics to represent for face surface's characteristic, and tensor space of depth.

1.3 Motivation

The goal of this thesis is to provide a practical method that can reconstruct 3D Face model from a single image taken by a camera in Android Smartphone. Herein, we just solve the frontal face image with unknown light condition. In this case, the face can be formulated as a bilinear object whose parameters are personal characteristics and light condition.

The proposed method takes a efficient short time to applied facial shape reconstruction in Smartphone. The main contribution of this thesis is that to design and implement a 3D application in Android Smartphone to be used in practice based on a fast and effective method to reconstruct 3D facial shape from an single frontal face image. The method using CCA to get the correlation between identity vector and depth map is suggested and testing with FRGC 2.0 database in Matlab language in [14]. Consider its effectivity, in our work, we created a corresponding method in C code, and then, designed and implemented an attractive application in Android OS based Smartphone.

1.4 Organization of Thesis

In this thesis, 3D reconstruction of a single image using canonical correlation analysis algorithm for an Android Smartphone was discussed. The remainder of the thesis is organized as follows.

In section 2, the prior works related to the proposed method are briefly summarized. Illumination features that used to reconstruct the 3D shape and tensor algebra are mentioned in this section. Based on the tensor algebra, an efficient method to reduce dimensionality, N-mode SVD, is introduced.

In section 3, the model-based mapping method, CCA algorithm, is explained. Besides, the overall procedure for 3D face reconstruction is also briefly discussed.

In section 4, the detail design and implementation of the proposed method in Android Smartphone are discussed.

In section 5, experimental results and performances are shown.

Section 6 contains some concluding remarks and suggestions for future works.

2. Related Works

2.1 Tensor Algebra fundamentals

The detailed explanation of tensor algebra can be found in [17], so we, herein, just give an general introduction of its definition, basic mathematical operations and fundamental formulas. Tensor, also known as a multidimensional array or n-way array, is a high-order multidimensional extension of vector (1-order tensor) and matrix (2-order tensor). Let $A \in R^{n_1 \times n_2 \times \ldots \times n_N}$ be a tensor, then the order of A is N the kth mode of A is n_k-dimensional, and an element of A is denoted as $A_{i_1 i_2 \ldots i_n \ldots i_N}$, where $1 \le n \le N$. The inner product of two tensors $A \in R^{m_1 \times m_2 \times \ldots \times m_N}$ and $B \in R^{m_1 \times m_2 \times \ldots \times m_N}$ with the same dimensions is defined as

$$\langle A, B \rangle = \sum_{i_1 = 1, \ldots, i_N = 1}^{i_1 = m_1, \ldots, i_N = m_N} A_{i_1, \ldots, i_N} B_{i_1, \ldots, i_N}.$$

The norm of a tensor A is $\| A \| = \sqrt{\langle A, A \rangle}$, and the distance between two tensors A and B is $\| A - B \|$.

The mode-k flattening of a tensor A is denoted as a matrix $A_{(k)} \in R^{n_k \times (\Pi_{j \neq k} n_j)}$, and $A_{i_1 i_2 \ldots i_N}$ is the same as the $(i_k, i')th$ element of $A_{(k)}$ where

$$\begin{aligned} i' &= i_1 + n_1 (i_2 - 1 + n_2 (i_3 - 1 + \ldots \\ &\quad + n_{k-1} (i_{k+1} - 1 + n_{k+1} (i_{k+2} - 1 + \ldots \\ &\quad + n_{N-2} (i_{N-1} - 1 + n_{N-1} (i_N - 1)))))) \ldots \\ &= 1 + \sum_{j \neq k} (i_j - 1) \prod_{l \neq k, l < j} n_l. \end{aligned}$$

The mode-k product, denoted by $\times_k$, of a tensor and a matrix is extended from the product of matrices. Let $M \in R^{n_k' \times n_k}$ be a matrix, and a tensor $B = A \times_k M \; (\in R^{n_1 \times n_2 \times \ldots \times n_{k'} \ldots \times n_N})$ is defined as

$$B_{i_1 \ldots i'_k \ldots i_N} = \sum_{i_k=1}^{n_k} A_{i_1 \ldots i_k \ldots i_N} \times M_{i'_k i_k}.$$

It can be also expressed in terms of flattened matrices, $B_{(k)} = MA_{(k)}$, where $A_{(k)}$ and $B_{(k)}$ are mode-k flattening of tensor A and B. The mode-k flattening and product are described in Fig. 2.4 and 2.5, respectively. The following relations are satisfied for the mode-k product:

$$\begin{aligned} A \times_j M_1 \times_k M_2 &= A \times_k M_2 \times_j M_1 \quad \text{for } j \neq k, \\ A \times_k M_1 \times_k M_2 &= A \times_k M_2 M_1, \\ A \times_k (M_1 + M_1) &= A \times_k M_1 + A \times_k M_2, \\ (A + B) \times_k M &= A \times_k M + B \times_k M \end{aligned}$$

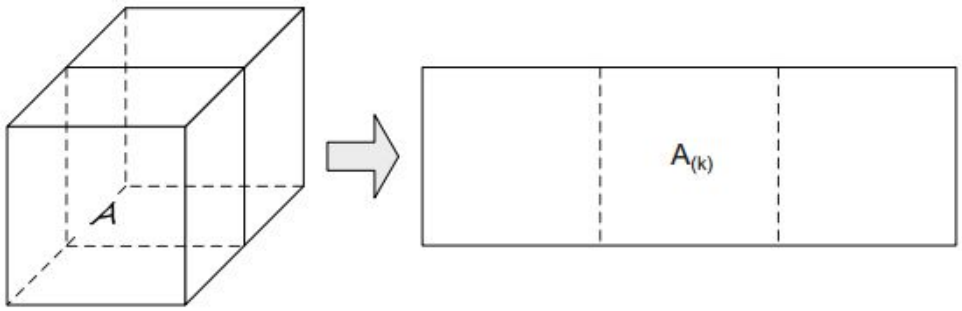


Figure 2.1: Mode-k Flatten a tensor

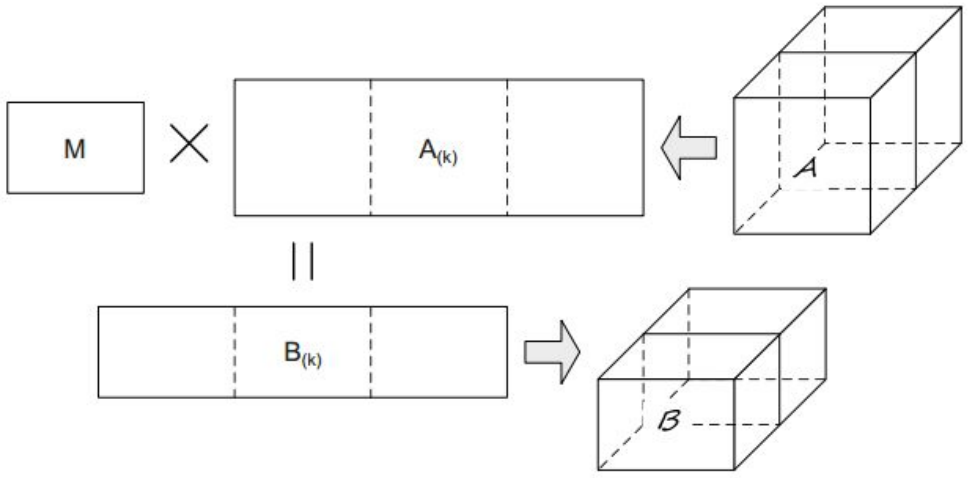


Figure 2.2: Mode-k Product of a matrix and a tensor

The mode-k concatenation is the tensor counterpart of the matrix concatenation. It is showed that

$$D_{(j)} = \begin{bmatrix} A_{(j)} & B_{(j)} \end{bmatrix} \text{ for } j \neq k, \tag{2.1}$$
$$D_{(k)} = \begin{bmatrix} A_{(k)} \\ B_{(k)} \end{bmatrix}.$$

Moreover, the following relations are satisfied:

$$\begin{bmatrix} A \times_j M & B \times_j M \end{bmatrix}_k = \begin{bmatrix} A & B \end{bmatrix}_k \times_j M \quad \text{for } j \neq k, \tag{2.2}$$
$$\begin{bmatrix} A \times_k M_1 & B \times_k M_2 \end{bmatrix}_k = \begin{bmatrix} A & B \end{bmatrix}_k \times_k \begin{bmatrix} M_1 & 0 \\ 0 & M_2 \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \end{bmatrix},$$
$$\begin{bmatrix} A \times_k M_1 & A \times_k M_2 \end{bmatrix}_k = A \times_k \begin{bmatrix} M_1 \\ M_2 \end{bmatrix},$$
$$\begin{bmatrix} A & B \end{bmatrix} \times_k \begin{bmatrix} M_1 & M_2 \end{bmatrix}_k = A \times_k M_1 + B \times_k M_2$$

The mode-k concatenation is described in Fig. 2.4.

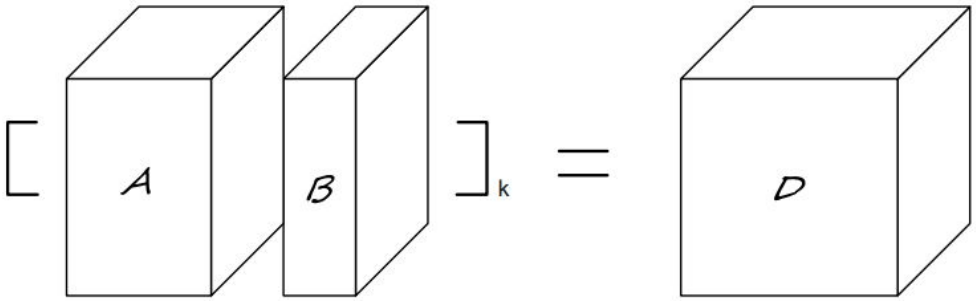


Figure 2.3: Mode-k concatenation

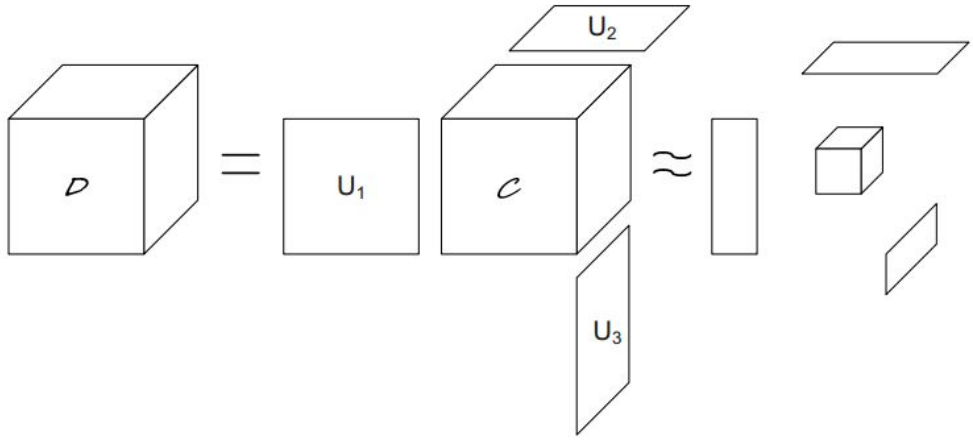


Figure 2.4: N-mode SVD to dimensional reduction

2.2 N−mode SVD (Singular Value Decomposition)

In order to separate and parsimoniously represent the constituent factors, tensor D should be decomposed. We subject D to a generalization of matrix SVD. N−mode SVD is a generalization of conventional matrix SVD, defined as

$$D = C \times_1 U_1 \times_2 U_2 \times_3 \cdots \times_N U_N ,$$

(2.3) where C is the core tensor and $U_k s$ are the orthogonal mode matrices derived from SVD of $D_{(k)} = U_{(k)} \sum^{k} V_k^T$, and C is defined as

$$C = D \times_1 U_1^T \times_2 U_2^T \times_3 \cdots \times_N U_N^T . \qquad (2.4)$$

The core tensor does not have a simple, diagonal structure; however, it is analogous to the diagonal singular value matrix $\sum$ in conventional matrix SVD. The core tensor governs the interaction between the mode matrices $U_1, U_2, \cdots, U_N$. Mode matrix U_n contains the orthogonal vectors spanning the column space of matrix $D_{(n)}$ resulting from the mode−n flattening of D (Fig. 2.1). The N−mode SVD algorithm for decomposing D is done as follows. Firstly, for $n = 1, 2, \ldots, N$, figuring matrix U_n in (5) by applying SVD to the flattened matrix $D_{(n)}$ and getting $U_{(n)}$ which is the left matrix of SVD. Finally, the core tensor is

computed as (6).

Because in the core tensor, the elements near the lower diagonal entries are usually much larger in magnitude than the others and any pair of two sub-tensors are orthogonal, N-mode SVD is usually utilized to truncate the unimportant dimensions of C and U_k that correspond to the entries of smaller magnitude, to reduce the dimension of the original tensor. This concept is clearly illustrated in Fig. 2.4.

The reshaping of a tensor is denoted as

$$B = R(A; n'_1, n'_2, ..., n'_N) ,$$

where $A \in R^{n_1 \times n_2 \times .... \times n_N}, B \in R^{n'_1 \times n'_2 \times .... \times n'_N}$, and $\prod_l^N n_l = \prod_l^{N'} n'_l$

and $B_{i'_1 ... i'_N} = A_{i_1 ... i_N}$ for

$$1 + \sum_{j=1}^{N'} (i'_j - 1) \prod_{l<j} n'_l = 1 + \sum_{j=1}^{N} (i_j - 1) \prod_{l<j} n_l .$$

Each i_j of $\{i_1, i_2, ..., i_N\}$ that corresponds to i'_j of $\{i'_1, i'_2, ..., i'_N\}$ can be computed as follows

$$i_j = 1 + \left(\left\lfloor \frac{\sum_{j'=1}^{N'} (i'_{j'} - 1) \prod_{l'<j'} n'_l}{\prod_{l<j} n_l} \right\rfloor \bmod n_j \right),$$

(2.5)

where is the floor operator. For a Nth-order tensor A, the mode-1 or mode-N flattening can be defined by the reshaping operator:

$$A_{(1)} = R\left(A; n_1, \prod_{l>1} n_l\right),$$
$$A_{(N)} = R\left(A; \prod_{l<N} n_l, n_N\right)^T .$$

The vectorization operator is defined as the following:

$$(A) = R\left(A; \prod_l n_l\right) .$$

The mode-k product of a tensor is expressed in terms of the reshaping

operator and the Kronecker product. Let $B = A \times_1 M_1 \times_2 M_2 \times_3 \cdots \times_N M_N$, where $A \in R^{n_1 \times n_2 \times \cdots \times n_1}$, $B \in R^{n'_1 \times n'_2 \times \cdots \times n'_1}$, and $M_k \in R^{n'_k \times n_k}$, then it can be shown after some manipulation that the following relation is satisfied:

$$R\left(B; \prod_{i \le \tilde{N}} n'_i, \prod_{j > \tilde{N}} n_i\right) = \left(M_{\tilde{N}} \otimes M_{\tilde{N}-1} \otimes \cdots \otimes M_1\right), \qquad (2.6)$$
$$R\left(A; \prod_{i \le \tilde{N}} n_i, \prod_{j > \tilde{N}} n'_i\right) = \left(M_N \otimes M_{N-1} \otimes \cdots \otimes M_{\tilde{N}+1}\right),$$

where $\tilde{N} < N$. A similar relation need to be satisfied for flattening operators as

$$B_{(k)} = M_k A_{(k)} \left(M_N \otimes \cdots \otimes M_{k+1} \otimes M_{k-1} \otimes \cdots \otimes M_1\right)^T. \qquad (2.7)$$

2.3 Lambertian Reflectance and Face model' s tensor

2.3.1 Lambertian Reflectance

Lambertian reflectance is the property that defines an ideal diffusely reflecting surface. The apparent brightness of such a surface to an observer is the same regardless of the observer's angle of view. On another words, a Lambertian surface is an ideal diffuse surface which reflects light source in all directions.

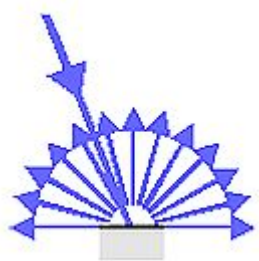


Figure 2.5: Ideal diffuse reflection

Under a single light source, the brightness of each pixel is satisfied the following Lambert' s cosine law:

$$I(x,y) = \rho(x,y) \lfloor b(x,y)^T s \rfloor . \qquad (2.8)$$

where

$$\lfloor x \rfloor = \begin{cases} x, & x > 0 \\ 0 & otherwise \end{cases}$$

$I(x,y), \rho(x,y)$, and $b(x,y)$ are the brightness, albedo, and the surface normal vector of the pixel (x,y), respectively, and s is the light source vector. Because of its simple formulation, the Lambertian assumption is widely applicable in many fields. Basri and Jacobs in [18] have proved that with arbitary light condition, the image of a convex Lambertian object can be efficiently approximated as

$$I(x,y) \approx f(x,y)^T s'$$

in 9D linear subspace, where f(x, y) is the 9 lowest spherical harmonic coefficients of the surface normal vector of the pixel (x, y), and s' is the shperical harmonic coefficients of the light condition. An effective example of the nine lowest spherical harmonic coefficients $f(x,y)$ of a pixel is

$$f(x,y) = \rho \left[1 \;\; b_x \;\; b_y \;\; b_z \;\; 3b_z^2 - 1 \;\; b_x b_y \;\; b_x b_z \;\; b_y b_z \;\; b_x^2 - b_y \right]^T .$$

According to the experimental results, images in a 9D space can captures at least 98 percent of the energy of all the model' s images. Frolova and et all. in [18] have shown that the accuracy of spherical harmonic approximation in 4D and 9D spaces is at least 87.2% and 99.2%, respectively.

2.3.2 Tensor formulation of Face model

In recent studies involved with image, the image is usually represented by a vector. However, this transformation may disturb the high-order statistical information and spatial structure of the image. In addition, the derived vector is usually high-dimensional, and hence brings about the curse of dimensionality problem. In this thesis, we have applied a tensor model to

formulate 3D image. It helps maitain the intrinsic image' s structure.

Assume that human face is a Lambertian model, we can approximate the image of human face as a linear equation:

$$I(x,y) \approx f(x,y)^T s \ , \qquad (2.9)$$

where $I(x,y)$ is the brightness of the pixel (x,y) and $s \in R^{n_l}$ is the light condition vector; $f(x,y)$ is a n_l−dimensional vector related to the surface characteristics, specifically, is scaled normal $(n_l = 3)$ or spherical harmonics $(n_l = 4 \text{ or } n_l = 9)$. In this work, we use spherical harmonics to deal with light condition handling, hence $n_l = 4$ or $n_l = 9$.

To represent face model as a tensor, we reformulate (2.9) as a tensor equation:

$$I \approx F \times_3 s^T$$

There $I \in R^{n_y \times n_x}$ is an image matrix, $I_{yz} = I(x,y)$, and $F \in R^{n_y \times n_x \times n_l}$ is a 3−order tensor, $F_{yxi} = f_i(x,y)$. Assume that the pose of face is fixed to the viewing direction and F is a function of personal identity vector $\phi \in R^{n_p}$,

$I(\phi, s) \approx F(\phi) \times_3 s^T$.

There exists a proper representation of ϕ, where $F(\phi)$ is a linear function of as the following:

$F(\phi) = \overline{F} + T \times_4 \phi^T$.

where $\overline{F} \in R^{n_y \times n_x \times n_l}$ is a 3−order tensor and $T \in R^{n_y \times n_x \times n_l \times n'_p}$ is a 4−order tensor. In this work, $\overline{F}$ and T are computed from the training step by utilizing N−mode SVD technique due to its ability to handle high−order tensor and dimensional reduction. $\overline{F}$ is calculated by getting

the mean value of each spherical harmonic component. Specifically, $\overline{F_{yxi}}$ is the mean value of $i-th$ spherical harmonic component of the $(x,y)-th$ pixel in the training samples. Meanwhile, T is obtained by applying N-mode SVD to the 4-order tensor Q with Q_{yxi} constructed from spherical harmonic component of the $(x,y)-th$ pixel of $j-th$ training sample. Before using N-mode SVD to truncate unimportant dimension parts, we align Q as

$$\widetilde{Q_{yxi}} = Q_{yxi} - \overline{F_{yxi}}$$

then,

$$\begin{aligned} \widetilde{Q} &\approx C \times_1 U_y \times_2 U_x \times_3 U_l \times_4 U_p \\ &\approx T \times_4 U_p \end{aligned} \tag{2.10}$$

There $U_y \in R^{n_y \times n'_y}$, $U_x \in R^{n_x \times n'_x}$, $U_l \in R^{n_l \times n'_l}$, $U_p \in R^{n_p \times n'_p}$, are mode matrices of columns, rows, positions, and people in the training set, respectively; and core tensor $C \in R^{n'_y \times n'_x \times n'_l \times n'_p}$, where $n'_y < n_y, n'_x < n_x, n'_l < n_l, n'_p < n_p$. Hence is a dimension reduced tensor of in mode-4.

2.4 Alternating Least Square Optimization

For each new image I' of each person, we need to find a pair of ϕ and s that satisfy:

$$I' = \left(\overline{F} + T \times_4 \phi^T\right) \times_3 s^T. \tag{2.11}$$

This work can be considered as finding ϕ and s in a nonlinear least-square optimization by reformulating (2.11) as follows:

$$\text{minimize } \| J \|^2 = \| I' - \left(\overline{F} + T \times_4 \phi^T\right) \times_3 s^T \|^2. \tag{2.12}$$

or

$$\text{minimize } \| J \|^2 = \| I' - \left(\overline{F} + S \times_1 U_y \times_2 U_x \times_4 \phi^T\right) \times_3 s^T \|^2. \tag{2.13}$$

According to the demonstration in [14], (2.13) can be simpled by as a new problem:

minimize $\| J \|^2 = \| L - \left(R + S \times_4 \phi^T\right) \times_3 s^T \|^2$.

(2.14)

where $L = U_y^T I' U_x$, $R = \overline{F} \times_1 U_y^T \times_2 U_x^T$

To find an optimal solution ϕ and s, we apply the ALS method. We first optimize (2.14) with respect to ϕ for a fixed s and then with respect to s for a fixed ϕ, and this process is repeated until convergence. Optimizing with respect to either ϕ or s gives a linear least squares problem and the solution can be found easily in a closed form. First, we set $\phi \leftarrow 0$ (mean face), and then calculate following equations iteratively,

$$
\begin{aligned}
V &\leftarrow R + S \times_4 \phi^T, \\
s &\leftarrow V_{(3)}^{+\,T} l, \\
W &\leftarrow S \times_3 s^T, \\
\phi &\leftarrow W_{(4)}^{+\,T} (l - R_{(3)}^T s).
\end{aligned}
\qquad (2.15)
$$

Here , $(.)^+$ denotes the Moore-Penrose pseudo-inverse and l is the vectorized version of L. This iteration is executed until the norm of the change of s is less than a predefined threshold ϵ .

3. Depth Map Reconstruction

This chapter gives an explanation for related algorithm (CCA) and the mapping of the method suggested in [14]. CCA is a simple algorithm to use but effective and powerful by its characteristics. We also describe the way to apply CCA to face reconstruction. In summary, this method is a simple-to-use and fast as well as effective one for 3D Reconstruction field. We have ported this method to Android by designing a powerful application to perform its quality with real image, not only images from database. This work is clearly described in Chapter 4.

3.1 Canonical Correlation Analysis (CCA) Algorithm

It is known that not all the component variables in the parameter vector have the same contribution to the mapping task and there exists redundancy and noise among them which may even have negative effects for the mapping. Therefore, we first apply CCA approach on two spaces to find the most correlative and complementary factors and then build the mapping based on them.

Canonical Correlation Analysis (CCA) is a very powerful tool for finding the linear relationship between two sets of multi-variate measurements in their leading factor subspaces. Similar to principle components analysis (PCA), CCA also reduces the dimension of original sets, because just some pair data are required to estimate the relationship between two sets. Nevertheless, CCA can deal well with two multidimensional spaces, so it is much better for regression task than PCA.

Consider the linear combinations $x = x^T \widehat{w_x}$ and $y = y^T \widehat{w_y}$ of the two variables respectively. Finding the canonical correlation between these sets is maximizing their correlation parameter. This means that the

function to be maximized is

$$\rho = \frac{E[xy]}{\sqrt{E[x^2]E[y^2]}} = \frac{E[xy]}{\sqrt{E[\widehat{w_x^T} x x^T \widehat{w_x}]E[\widehat{w_y^T} y y^T \widehat{w_y}]}}$$
$$= \frac{w_x^T C_{xy} w_y}{\sqrt{w_x^T C_{xx} w_x w_y^T C_{yy} w_y}}$$

whereby, $C_{xx} \in R^{p \times p}$ and $C_{yy} \in R^{q \times q}$ are the within-set covariance matrices of x and y, respectively, while $C_{xx} \in R^{p \times p}$ denotes their between-set covariance matrix.

The maximum of ρ with respect to w_x and is the maximum canonical correlation. In the context of CCA, the projections x and y are also called canonical variates.

A number of at most $k = \min(p,q)$ factor pairs $\langle w_x^i, w_y^i \rangle$, $i = 1,...,k$, can be obtained by successively solving

$$w^i = \left(w_x^{iT}, w_y^{iT}\right)^T = \arg max\left(w_x^i, w_y^i\right)\rho$$
$$\text{subject to } \rho\left(w_x^j, w_y^i\right) = \rho\left(w_x^i, w_y^j\right) = 0$$
$$\text{for } j = 1, \ldots, i-1$$

The factor pairs w^i can be obtained as solutions (i.e. eigenvectors) of a generalized eigenproblem. The extremum value $\rho(w^i)$, which are referred to as canonical correlations, are obtained as the corresponding eigenvalues.

By employing CCA, we perform regression on only a small number (compared to the original dimensionality of the data) of linear features, I.e. derived linear combinations of original response variables y. Thus, CCA can be used to compute the reduced rank-n regression parameter matrix by using only n < k factor pairs. Thereby, in contrast to standard multivariate regression CCA takes advantage of the correlations between the response variables to improve predictive accuracy.

3.2 Features mapping

A model-based mapping is used to calculate the depth information. In this work, CCA is employed to finding the mapping between depth and face surface. M. Reiter in [19] suggested predicting depth maps from RGB face image by utilizing CCA for a set of pairs of RGB image' s data vector and corresponding depth maps. That method can be easy to use, but the accuracy of the result is weaken because it is difficult to finding a accurate and direct map between image data and corresponding depth.

In this thesis, we estimated the depth map by utilizing CCA for finding a correlation, a linear relationship between personal identity vector of a face image ϕ and its corresponding depth d in modeling step. The procedure of this work is as follow

Step 1: Estimate the leading factor pairs from samples of N pairs of normal tensor and depth tensor.

Step 2: Compute the regression parameter matrix

Step 3: Identify the linear mapping between identity vector and depth map

It is simple to use and its computation is efficient to apply in a Smartphone. The personal identity vector of each face image ϕ is calculated from optimizing the nonlinear equation

minimize $\| J \|^2 = \| L - \left(R + S \times_4 \phi^T\right) \times_3 s^T \|^2$.

using Alternating Least Square (ALS) technique. The correlation parameters is built in modeling step by applying CCA for personal identity vector ϕ of each image and its corresponding depth map d in training samples. In reconstructing step, identity vector of the testing image is used to reconstruct the depth information based on the correlation parameters trained in modeling step by CCA algorithm.

3.3 Procedure

The overall procedure of the designed scheme can be summarized as follows.

[Modeling]

1. Apply affine transformation to the spherical harmonic image of each training sample to align all eyes, mouth ‘s position of training data into unanimously defined points.
2. Calculate the mean tensor and the deviation tensor using the transformed harmonic images.
3. Apply N-mode SVD to the deviation tensor.
4. Calculate the personal identity vector of each training sample

[Reconstruction]

1. Detect and crop face from test image
2. Find position of eyes and mouth of the cropped face
3. Apply the affine transform to the test image to get the aligned face.
4. Apply ALS algorithm to estimate personal identity vector and light condition
5. Compute the depth map using the mapping built by CCA and apply the inverse affine transform to get the final result.

4. Implementation in Smartphone

4.1 Overview of Design

In this work, a 3D reconstruction application is created for Android OS based Smartphone. The application run by loading an sample image in any storage in the phone, such as: internal memory or external card; then, the detecting face algorithm, that applied Haar-like wavelet, is called to identify and crop the frontal human face in the image.

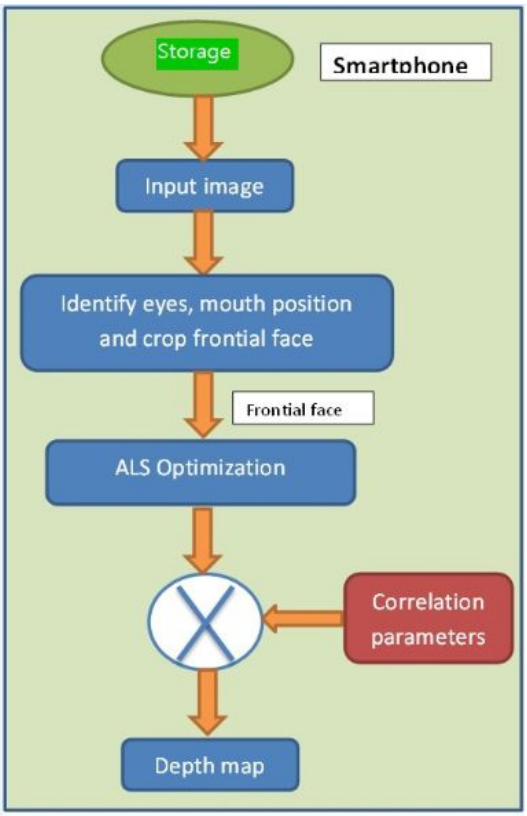


Figure 4.1: Processing Flowchart on Smartphone

Color cropped face is used to extract texture features that adding to the depth map result to recover a full 3D face. Meanwhile, the gray cropped face is the main object to reconstruct depth of each corresponding face pixel. The figure 4.3 described in detail the design of our application.

4.2 Face Detection and Alignment

Detect_Crop_Align_Face module keeps a key factor in the application. It is a verifying gate to confirm if an image is satisfy the condition of applying the reconstruction algorithm, the frontal face image, and is also a modifier that standardizes input image into suitable form. Specifically, this module applied Haar-like cascades technique to detect the frontal face existing in the image. If the module can find the face, it means the captured image fulfills the input requirement; unless, the work should stop here. To get fast results of face detection in mobile devise, it is decided to utilize OpenCV library which contains many programming functions mainly aimed at real-time computer vision. OpenCV provides a set of open source programming libraries for computer vision applications, and especially, it includes an implementation of face detection algorithm [15]. Therefore, we utilize OpenCV as a tool to implement face detection in this module. Face detection with OpenCV support is done by calling Haar-classifier that loaded from xml-format file trained to detect frontal face in images. Figure 3 illustrates detecting and cropping face of the application.

Because there are morphological differences between different faces due to human characteristics, such as length of face or distance of two eyes…, it is necessary to geographically normalize all images. In order to solve this geographical problem, we apply a landmarks based method. The positions of eyes and mouth are used as landmarks to represent facial morphography. We designated a positions set of landmarks and then, used an affine transformation ($\mathbf{A},\mathbf{t}$) to convert the images and spherical harmonic images

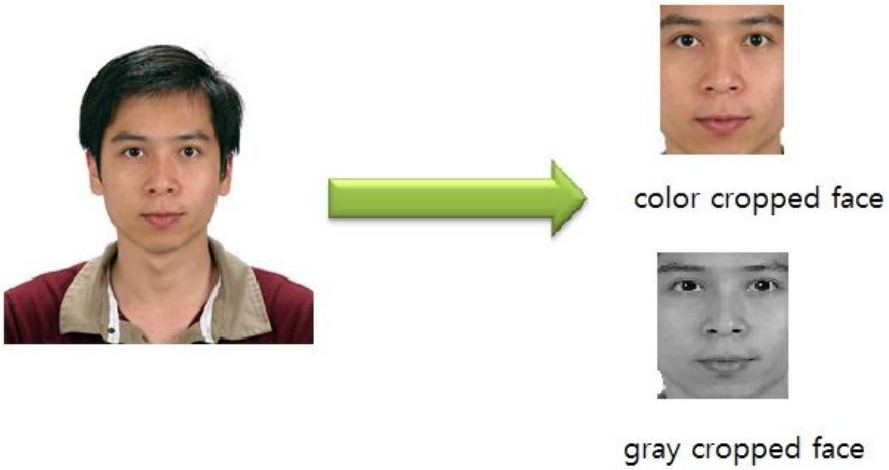


Figure 4.2: Cropped frontal face image

in training samples into a standard form in modeling step, so the images cropped in *Detect_Crop_Align_Face* module I_t also needs to be normalized in this way. After cropping and aligning, the image is resize to 120x100 in order to normalize image and find the corresponding depth information in the reconstruction step.

4.3 Three dimension Reconstruction

Figure 4.3: 3D Reconstruction model

Meanwhile, 3DFace_Reconstruction module is the core of the application. It can be said that this module is the engine of reconstructing because it includes most of important works related to the convert-to-3D techniques.

FIrstly, the image is normalized to improve its effect in finding identity vector.

$$I_{norm}(x,y) = \frac{I(x,y)}{\sqrt{\sum_{x,y} I(x,y)^2}}$$

This normalized image is basic input for applying ALS algorithm to find identity vector and light condition of the image. However, in order to improve the speed of optimizing and reduce computational cost, we utilized the result of N-mode SVD in modeling step to reduce dimension of the image. The detail of work can be found in [14]. It should be noted that dimension space reduction leads to reduce the number of iterative loop of the optimization algorithm. We denote the reduced image is $L = U_y^T I_{norm} U_x$, and its vectorized version is l. Then, personal identity vector and light condition of the image are calculated by minimizing the value $\| J \|^2 = \| L - \left(R + S \times_4 \phi^T\right) \times_3 s^T \|^2$. The optimization is done by alternatingly keep one value of (ϕ, s) be fixed and the other as a variable to minimize $\| J \|^2$.

In modeling step, a linear mapping from identity vector to depth information has already been calculated. Thus, in mobile device' s application, after estimating the identity vector of the face image

$$d = M\phi + D$$

There, d is the depth map which needs to be estimated; M is the linear mapping parameter matrix which calculated in modeling step. D is the average value of depth images of modeling faces. D needs to added to the formula of reconstruction depth information because CCA is applied for two non-zero sets.

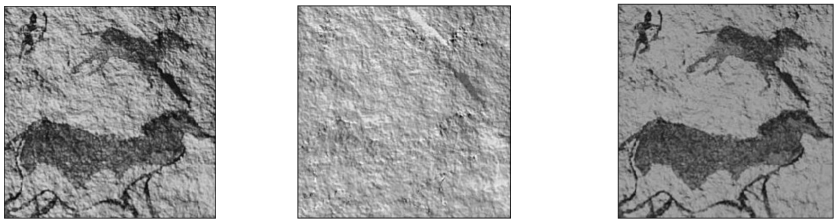
4.4 Texture Visualization

Visualization of 3D fields is investigated and used in various scientific and

engineering disciplines in many years. The most popular approach for this work is to employ a dense representation in the form of a texture-based visualization. Recently, the increasing power and functionality of GPUs (graphics processing units) has been exploited to promote the effect of this approach.

4.4.1 Texture

Texture can be seen in many images from multi-spectral remote sensed data to microscopic photography. Currently, there is still not an unique and precise definition of "texture" term, because it depends on the purpose in which image texture is used and the features extracted from the image. However, all researcher agreed that texture is an important surface characteristic to identify and recognize objects. It can be said that the texture of an image is something which describes the characteristic of the intensity surface of the image. Intensity can be measured at resolution of a single pixel, whereas texture can only be perceived from an image region which is large enough. The term *image texture*, or simply *texture*, usually refers to the image of a textured surface. Image texture can originate not only from surface albedo variations (2D) but also from surface depth variation (3D). Figure 4.2 is an illustration of extracting surface relief and albedo from a 3D surface. Albedo represents the reflectance ability of a surface (2D), meanwhile, surface relief represents surface properties in 3D.



a. 3D surface b. Surface relief c. Albedo

Figure 4.4: Extracting surface relief and albedo from 3D surface

4.4.2 Visualization

Visualization mentioned in this thesis is to perform the depth map with the texture features from the original 2D image in smartphone. This texture is created from color image and describes sufficiently information of each pixel in the face, so the result is fairly impressive. In this work, we exploited the support of OpenGL library for graphic development. In Android platform, we used the supported OpenGL ES 2.0 which is a subset of OpenGL API. It is designed to support Embedded System such as mobile phones, PDAs, game video consoles. OpenGL handles texture by look-up table method, it means that storing data of texture, "color" of each pixel, as one or more arrays of data with all of the arrays having some dimensionality. In our work, we used two arrays of data to store information of depth and color of each pixel of the reconstructed 3D face image. This process can be decribed as follow:

Firstly, it is necessary to declare usage of Vertex Array and Color Array in OpenGL. This enabling allows application give 3D points and their corresponding color values to buffer of OpenGL.

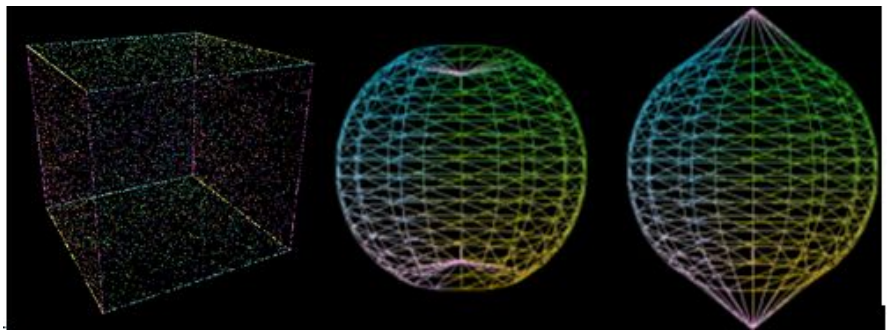


Figure 4.5: Example of using Vertex Array

Then, we push the depth and color information by using supported functions glVertexPointer(int size, int type, int stride, Buffer pointer) and glColorPointer(int size, int type, int stride, Buffer pointer), respectively.

These information is used to perform the 3D image based a vertice system which created in advance by using the function glDrawElements(int mode, int count, int type, Buffer indices). Figure 4.5 illustrates an example of pushing depth data of an object to the buffer supported in OpenGL and then, accessing and visualizing this information by the shader.

Figure 4.6 shows an object of utilizing color buffer in OpenGL. It is a box with different color information on its sides. The purpose of the graphic effect “color” is to make the result more impressive and make the object more recognizable.

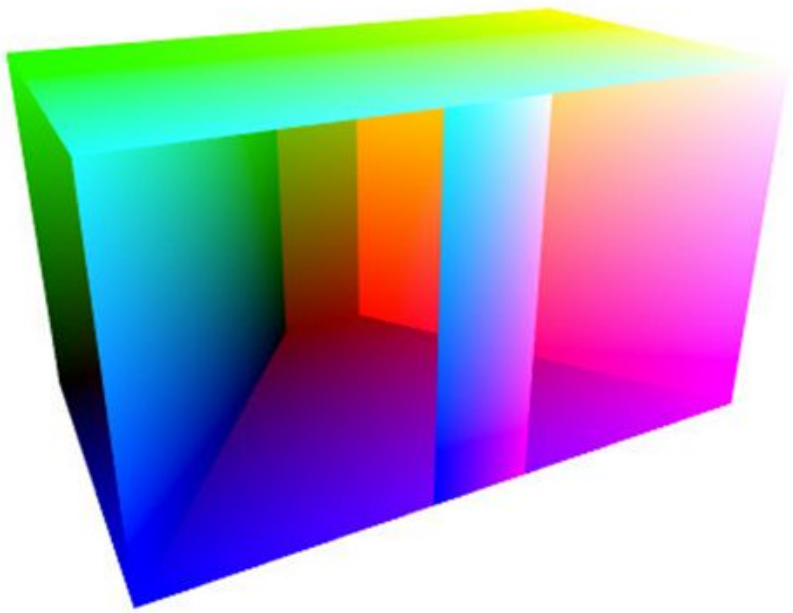


Figure 4.6: Example of using Color Array

5. Experimental Results

5.1 FRGC 2.0 Database

The Face Recognition Grand Challenge (FRGC) 2.0 was used to model the 3D human face. We used 400 sample images of FRGC to train the image model and 100 sample images for testing the trained model in our experiment. All the images, depths, and illumination bases were affine-transformed based on the center of each eye and the center of the mouth and were cropped to 100 × 120 pixels. We denote the illumination bases as "Harmonics" and "Images" for spherical harmonics and the rendered images, respectively.

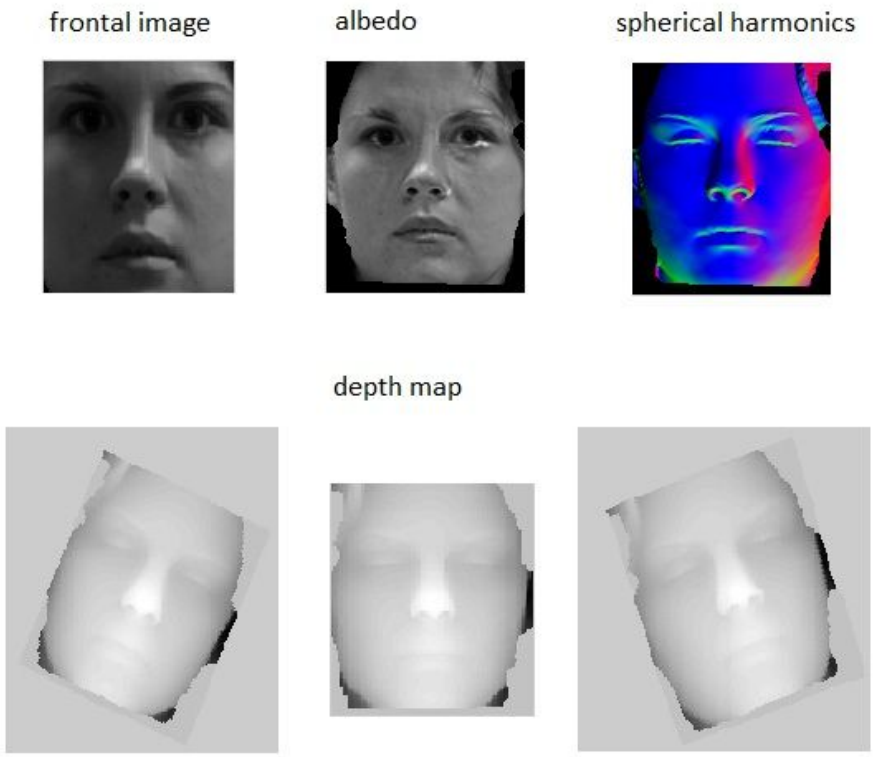


Figure 5.1: FRGC 2.0 database

For estimating the accuracy and quality of the used algorithm, we have tested it to FRGC 2.0 database. The result in database is good enough to guarantee that the result of reconstruction with the real frontal face image have a acceptable error.

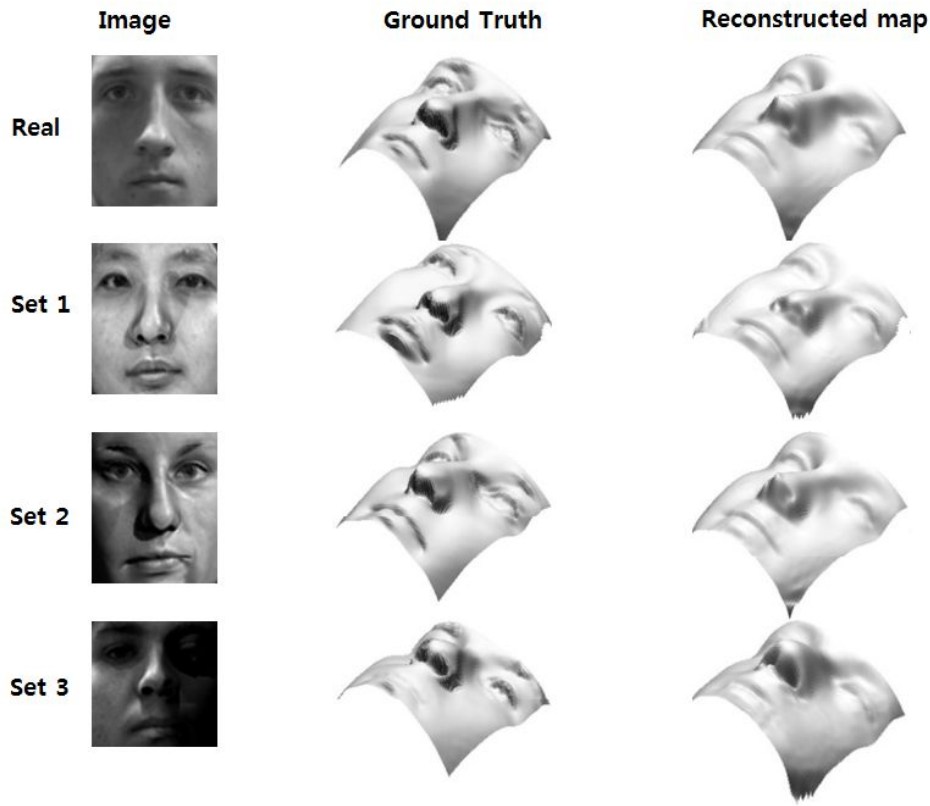


Figure 5.2: Reconstruction result in FRGC 2.0 database

Table 5.1

Mean absolute angle and depth error.

	angle error (degree)	depth error (voxel)
Real image set	14.26	2.47
Set 1	15.39	2.94
Set 2	16.30	3.29
Set 3	17.72	3.62

5.2 Results from sample image

Figure 5.3: Sample image

The sample image in Figure 5.3 is used as an input image to verify the quality of the application. The frontal face, then is detected, extracted and resized from the image to obtain a standard input for reconstruction step. The figure 5.4 illustrates the reconstructed result by creating a mesh grid to visualize depth information. This figure is done in Matlab with its supported image processing tool. The figure 5.5 is visualization of depth map in Smartphone by showing its 3D points in 3D space. It is the rough result of the application. From this rough map, texture features are added to form the final 3D image that describe reconstruction the most intuitively.

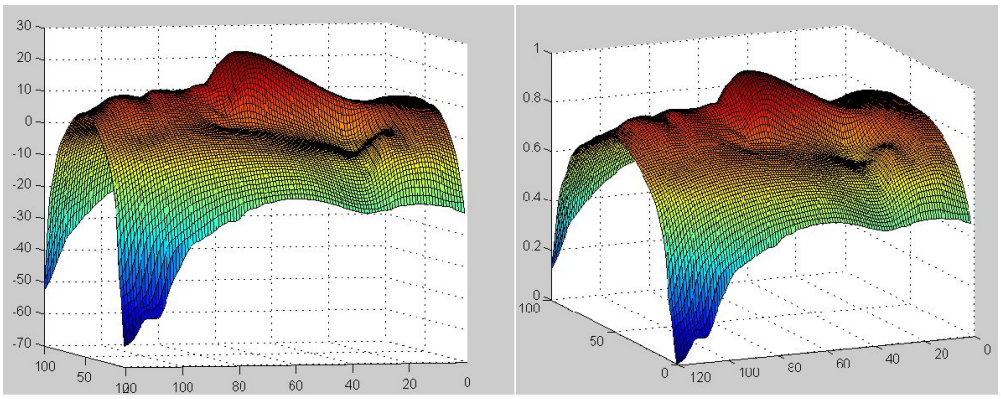


Figure 5.4: Mesh creation of reconstructed depth

Figure 5.5: Depth map

Figure 5.6: 3D face under different view

The processing time of each part of reconstruction application which listed in table 2, 3, 4, 5 shows clearly its practical applicability. In our experiments with other Android based devices, it takes averagely 5.6 ms, 13.5 ms, 2.855 ms and 4.471 ms for reconstructing part in Nexus One, Galaxy S, Galaxy S2 and Galaxy Note1, respectively. Totally, the time period for all processes in the devices is still acceptable for an Android application. Specially, it is 2.931 ms and 4.611 ms for Galaxy S2 and Galaxy Note 1. Note that it is faster to process in Galaxy S2 and Galaxy Note 1 than Nexus One and Galaxy S because they are older-released with better specifications and firmwares. The CPU speed of Galaxy S and one of Nexus One are both 1 GHz, but according to the experiment result, the performance in Nexus One is much better than one in Galaxy S. Meanwhile, Galaxy S2 and Galaxy Note 1 with CPU Duo-core 1.2 GHz and Dual-core 1.4 GHz, certainly give more effective computing costs.

Table 5.2

Processing time for each step in Nexus One.

Time	1	2	3	4	5	average
Detecting Face	1.635	0.71	0.713	0.711	0.729	0.900
Reconstructing	7.413	5.021	5.216	5.174	5.174	5.600
Visualizing	0.156	0.99	0.1	0.1	0.103	0.290
Sum	7.569	6.001	5.316	5.274	5.277	5.889

Table 5.3

Processing time for each step in Galaxy S.

Time	1	2	3	4	5	average
Detecting Face	2.321	0.885	0.99	0.969	0.83	1.199
Reconstructing	14.02	12.924	13.928	13.799	12.913	13.529
Visualizing	0.116	0.11	0.112	0.089	0.091	0.104
Sum	14.136	13.034	14.04	13.888	13.064	13.632

Table 5.4

Processing time for each step in Galaxy S2.

Time	1	2	3	4	5	average
Detecting Face	0.629	0.531	0.554	0.554	0.614	0.573
Reconstructing	2.841	2.812	2.905	2.983	2.732	2.855
Visualizing	0.09	0.065	0.076	0.085	0.065	0.076
Sum	2.931	2.877	2.981	3.068	2.797	2.931

Table 5.5

Processing time for each step in Galaxy Note 1.

Time	1	2	3	4	5	average
Detecting Face	1.775	0.951	1.24	0.956	0.706	1.126
Reconstructing	6.33	4.559	4.131	3.861	3.475	4.474
Visualizing	0.261	0.121	0.098	0.124	0.097	0.140
Sum	6.591	4.68	4.229	3.985	3.572	4.611

Unit: second (s)

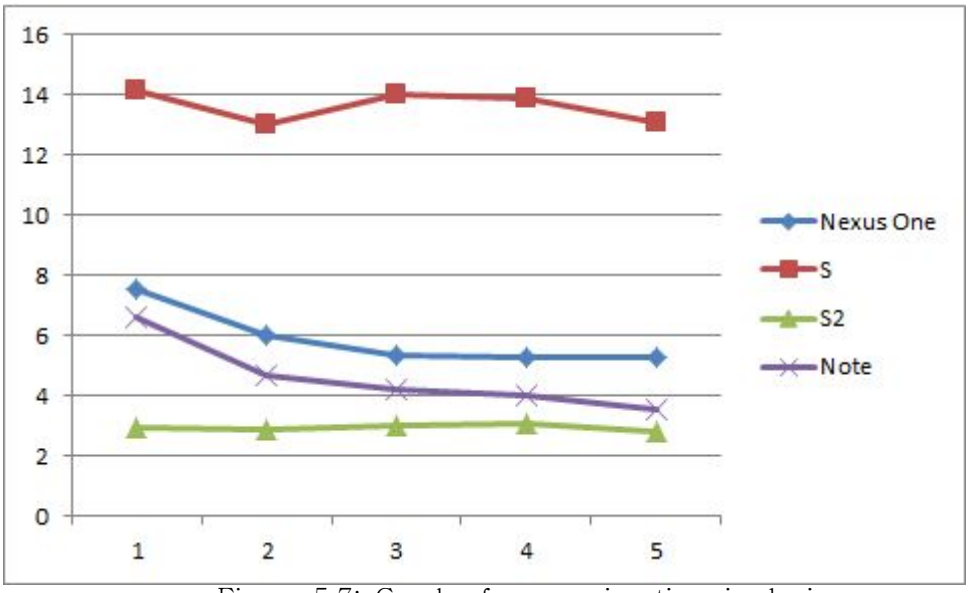


Figure 5.7: Graph of processing time in devices

6. Conclusions

6.1 Concluding remarks

In this thesis, a 3D Face Reconstruction scheme has been designed for an Android OS Smartphone. The mathematical analysis of mapping between the surface characteristics and the depth, and the performance of reconstruction algorithm have been provided.

In the prior works, multiple images of one object in different angles or multiple cameras in different angles was required to realize the 3D reconstruction. In contrast, this thesis suggest a method to reconstruct 3D Face Image from only one single image and introduce about the practical implementation of the algorithm in Smartphone. The processing time of the designed scheme is adequate for the purpose of creating an application in Smartphone.

6.2 Future works

Some suggestions for future works are addressed. Firstly, the application should process directly with data from camera. Camera is used to captured an image that contains a human face, then the alignment progress is employed to extract the standard frontal face image. That image is then preprocessed for illumination to adapt with the condition in database. That image now is applicable to the mentioned 3D Face reconstruction.

Secondly, developing the application to an photo-based-3D-face-speak one. An application stimulate human facial expression in 3D face model in accordance with the recorded voice. The frontal human face from the image is applied the designed 3D reconstruction scheme to build 3D face that is the source to perform the movement of mouth and facial in speaking. A

several efficient graphical transformations will be employed to stimulate the mouth' s opening, holding and closing, as well as eye blinking. The most important challenge in this work is to find an effective interpolation method for surrounding pixels of mouth and eyes when they are performing a movement or an action.

Thirdly, we will improve its performance by adding other parts of a human head to the 3D reconstructing, such as ears, head, hairs,... A 3D model of a head will be pre-built based on necessary information from database, reconstruction result, then will be inserted into this model with some predefined marking points.

Finally, the designed scheme needs to be improved in accuracy and processing time. The processing time is currently acceptable for an application in Smartphone, however, it is still a problem for a real-time application.

Bibliography

[1] O.Faugeras, “Three dimensional computer vision: a geometric viewpoint”, MIT Press, 1993.

[2] K. N. Choi and M. C. Mozer, “Recovering facial pose with the EM algorithm”, pp. 2073-2093, 2002.

[3] Y. Zhou, L. Gu, and H. Zhang, “Bayesian tangent shape model: Estimating shape and pose parameters via bayesian inference”, IEEE Conference on Computer Vision and Pattern Recognition, volume 1, pp. 109-116, 2003.

[4] Sung Won Park, Jingu Heo, Savvides M., “3D face reconstruction from a single 2D face image”, IEEE Computer Society Conference, pp. 1-8, 2008.

[5] H. Li, S.C. Yan, L.Z. Peng, “Robust multi view face alignment with edge based texture”, submitted to Journal of Computer Science and Technology, 2004.

[6] Sannier G, Magnenat Thalmann N, “A flexible texture fitting model for virtual clones”, Proceedings of Computer Graphics International, IEEE Computer Society, pp. 66-99.

[7] Yuencheng Lee, Demeteri Terzopoulos, Keith Waters, “Realistic modeling for facial animation”, Computer Graphics Proc. SIGGRAPH, pp. 55-56, 1996.

[8] W. Lee, P. Kalra, N. Magnenat Thalmann, “Model based face reconstruction for animation”, Proceedings of the MMM’ 9, World Scientific Press, Singapore, 99, pp. 333-338, 1999.

[9] R. Zhang, P. Tai, J. E. Cryer, M. Sha, “Shape From Shading: A Survey”, PAMI, 21(8), pp. 690-706, 1999.

[10] Y. Hu, D. Jiang, Shuicheng Yan, L. Zhang, H. Zhang, “Automatic 3D Reconstruction for Face Recognition”, Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference, pp. 843-848, 2004.

[11] Yuankui Hu, Ying Zheng, Zengfu Wang, “ Reconstruction of 3D face from a single 2D image for face recognition”, Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop, pp. 217-222, 2005.

[12] R. Hartley and A. Zisserman, “Multiple View Geometry in Computer Vision”, 2nd ed. Cambridge University Press, 2004.

[13] F.Verbiest and L. V. Gool, “Photometric stereo with coherent outlier handling and confidence estimation”, Proc. IEEE Conf. Computer Vision and Pattern Recognition, June 2008.

[14] M. Lee, C.-H. Choi, “Fast facial shape recovery from a single image with general, unknown lighting by using tensor representation”, Pattern Recognition (2011), doi: 10.1016/j.patcog.2010.12.018.

[15] Suen C.Y. Langaroudi, A.Z. Chunghua Feng, Yuxing Mao, “A survey of techniques for face reconstruction”, Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conf., 2007.

[16] Yuxing Mao, Chunhua Feng, C. Y. Sue., “Textured 3D face reconstruction on two images taken from different angle”, Int. Workshop on Advanced Image Tech., Bangkok, Thailand.

[17] M. A. O. Vasilescu and D. Terzopoulos, “Multilinear Subspace Analysis of Image Ensembles: Tensorfaces,” in Proc. European

Conf. Computer Vision, May 2002.

[18] Ronen Basri, David Jacobs, "Lambertian Reflectance and Linear Subspaces," iccv, vol. 2, pp.383, Eighth International Conference on Computer Vision (ICCV'01) - Volume 2, 2001.

[19] M. Reiter, R. Donner, G. Langs, H. Bischof, “3D and infrared face reconstructionfrom RGB data using canonical correlation analysis”, International Conference on Pattern Recognition, 2006.

[20] Min-Sik Lee, “Facial shape recovery from a single image with general, unknown lighting”, Doctoral Dissertation, Seoul National University, 2012.

[21] Jiahua Wu, B.Eng, “Rotation Invariant Classification of 3D Surface Texture Using Photometric Stereo”, Doctoral Dissertation, Heriot-Watt University, 2003.

[22] Daniel Weiskopf , Thomas Ertl, “GPU-Based 3D Texture Advection for the Visualization of Unsteady Flow Fields”, WSCG Conference Proceedings, 2004.

[23] Z.Lei, Q.Bai, R.He, S.Z.Li, “Face shape recovery from a single image using CCA Mapping between tensor spaces”, IEEE Int. Conf. on Computer Vision and Pattern Recognition, 2008.