Image Motion Estimation for 3D Model Based Video Conferencing

CHEUNG Man-kin

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Philosophy

in

ELECTRONIC ENGINEERING

August 2000

Acknowledgement

It is a pleasure to have the opportunity to thank many people who gave me help during my study for the Master degree. I would like to thank my supervisor, Prof. Hung Tat Tsui, for his guidance, helpful advice and discussions, in particular, for his patience in correcting my thesis.

I would like to thank my colleagues in the Computer Vision and Image Processing Laboratory, who provided supportive assistance during my research period. Last but not least, I am deeply indebted to my parents who always encourage me in my studies.

Abstract

It is convenient and desirable to have video conferencing capability through a mobile. However, the minimum bandwidth for conventional video conferencing is 64 Kbps (for 176×144 image). This is well beyond the real capacity of existing mobile phone systems, which is less than 10 Kbps. In this thesis, a novel 3D model based approach is proposed to tackle this problem. It is capable of doing video conferencing using higher resolution images (e.g. 352×288) through the existing mobile phone system. In this approach, a generic 3D facial model is used to fit the intensity image of a human head. This fitted model is then sent to the receiving end during the set up stage. When the video conferencing begins, we only detect the parameters of rigid motions of the head and the non-rigid motions corresponding to the changes in facial expressions. These motion parameters are sent to the receiver side and used to change the pose and expression of facial model. From the changed facial model, the same image will then be regenerated in the sending end. Since only about thirty parameters are to be sent, it is well within the capacity of the mobile phone system.

There are several parts in this approach: feature extraction, 3D points estimation, wireframe fitting and motion estimation. The focus of this thesis will be on the last two parts. In the process of wireframe fitting, we use an elastic surface approach. Only twenty-six control points are used to fit a 3D wireframe model with 300 nodes. The motion estimation is divided into two steps: rigid and non-rigid motion. A Two Stage Algorithm is proposed to tackle the problem of rigid motion of the head, the

algorithm minimizes the difference between the synthesized and the original images. A Muscle-Based Approach is proposed to handle the non-rigid motions of the face by using the actions of connected muscles. The techniques for coding and synthesis of facial image are then developed. Good results have been obtained for real and simulated experiments.

簡介

大家都希望能夠於手提電話去進行既方便又直接的視像會議，但視像會議的最低傳送空間仍需要 64 Kbps（176×144 像素）．不過可惜現今無線電網絡只能提供不超過 10 Kbps 的速度．有見及此，我們將會建議使用一個新的系統，名爲“模型爲本的視像會議系統”，它能提供更高解象度的圖像（352×288 像素）．其概念是以一個人頭模型去模擬正在會議中的對象，方法是首先製造一個屬於自己的人頭模型，在每次視像會議進行之前，在初始階段將自己的模型傳送給對方及接收對方的模型．從而在會議進行時，只要將自己頭部的剛性運動及面部表情計算出來，再將這兩個動作資料傳送給對方，對方就能根據所收到的有關資料去改變早前所收到那個模型，並能以三十多個數據去重新製造一幅新圖像．以此兩個模型及相對應的動作去模擬兩方的會談，就能比壓縮影像更能節省運送空間．

整個系統分爲特徵抽取，三維重建，模型重整以及動作計算，而本論文只會集中討論後者兩個部份．於模型重整的步驟中，能單單用二十六個點去改變一個有彈性的表面，從而就能重新製造一個差不多有三百個點的三維模型．而動作計算方面則分爲剛性及非剛性運動，我們以二步法（Two Stage Algorithm）去找出頭部的剛性運動，概念是盡量減少人造圖像及真實圖像的差別．另外我們利用肌肉爲本的方法，就能憑著肌肉運動去計算及改變那些非剛性動作的面部表情．而這兩部份亦已得到相當不錯的結果．

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, video is playing an increasingly important role in the world of telecommunication. There is a common saying, that a "picture is worth more than a thousand words". Consumers are not contented with just voice and data communications. They also want information in the form of images, e.g. video data. However, transmission of sound and video demands much larger bandwidth than voice and data. Due to limited bandwidth in wireless communication, it is difficult to carry out video conferencing through an existing mobile phone system.

Even though the video is coded by *MPEG-1* or *MPEG-2*, the bandwidth requirement is still more than 2 Mbps. This is a hard requirement for wireless communication. Therefore, more efficient video conferencing techniques are required to deliver the video information over the existing telecommunication network. Traditional approaches of video compression, which are predictive-transform coding techniques, adopt the video coding standards, like H.261 and H.263 [4,6,10] for videoconferencing. The lowest bandwidth requirement is about 64 Kbps for 176×144 color images

In this thesis, a novel approach called 3D model-based coding is proposed to tackle the problem of video conferencing on an existing mobile phone system. The

block diagram for the proposed system is shown in Figure 1.1.

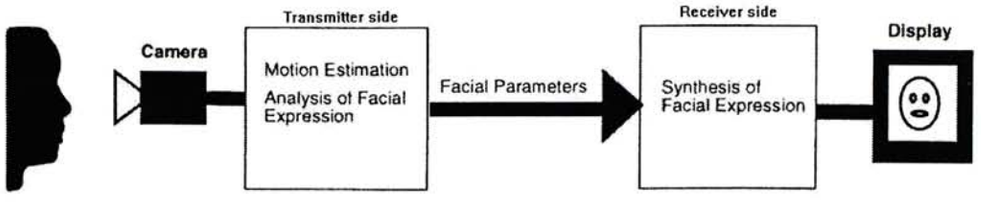


Figure 1.1: The block diagram of model-based 3D videoconferencing

In this approach, the human head is modeled by a 3D wireframe model (WFM). When the head changes the pose and expression, the corresponding motion parameters can be extracted on the transmitter side and be sent to the receiving end instead of the whole image through a low capacity channel. These parameters are used to change the pose and expression of texture mapped 3D WFM (*Facial Model*), such that the receiver can reproduce the image by using these motion parameters. So the corresponding image can be synthesized with an acceptable picture quality for the video conferencing by only about thirty motion parameters in real time process. Therefore, even there is no compression on these motion parameters for transmission, the required bandwidth is less than 6 Kbps for all image size. If the motion parameters are compressed, the bandwidth requirement will be much lower.

1.1. Building of the 3D WFM and FM

In the proposed 3D model-based videoconferencing, it is necessary to get the 3D WFM and FM of a person at the sending end before video conferencing. The block diagram of this initial process is shown in Figure 1.2,

This process consists of the following steps,

1) ***Stereo vision***: 3D information can be described from two stereo images, so the 3D

feature points can be used to the wireframe fitting process.

2) ***Wireframe fitting***: if the 3D information of the person *i* is available, his WFM can be generated by deforming the generic WFM to fit his face.

3) ***Texture extraction***: the texture of each patch can be extracted by using the corresponding between 3D WFM and image.

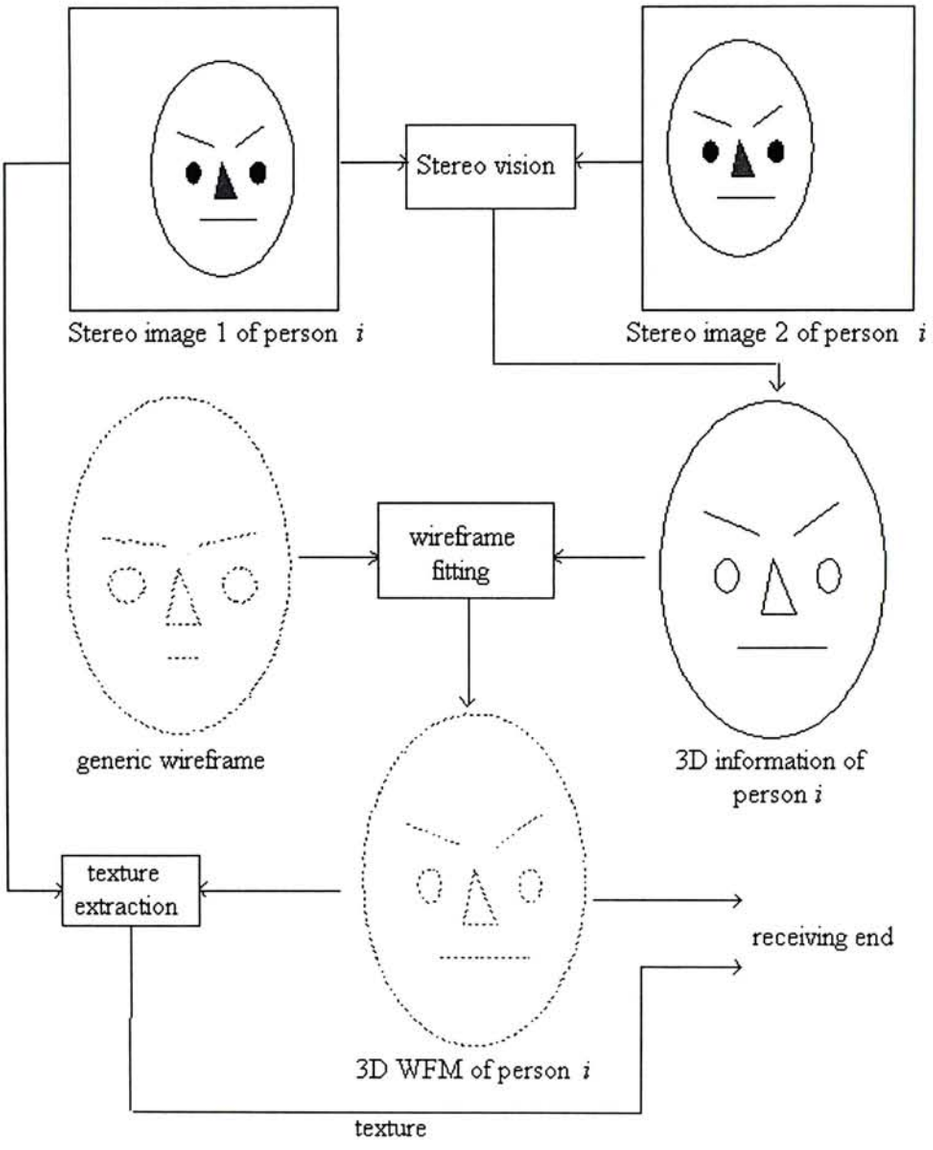


Figure 1.2: The block diagram for the building of 3D WFM at the sending end

After the 3D WFM is set up, it and its texture will then be sent to the receiving end. This 3D FM will represent the target head in the video conferencing.

1.2. Description of 3D Model Based Video conferencing

The block diagram of the 3D model-based videoconferencing is shown in Figure 1.3. By using this method, only eleven rigid and twenty non-rigid motion parameters, instead of the whole image, are needed to be transmitted. The procedures are shown as the following,

1) ***Feature extraction*** [37], to detect all control feature points, including rigid and non-rigid point. The procedure is that the head will be segmented by using the skin color. And then the feature points will be tracked by using the mask, such that it can speed up the extraction process. It is because closer initial points are given for detecting the points. However, feature extraction process will not be included in this thesis.
2) ***Rigid motion estimation***, to use the information of rigid control point to estimate the rigid transformation $\mathscr{D}$.
3) ***Facial motion estimation***, all nodes of WFM are undergoing the inverse transform, which is estimated in step 2. So the object is transformed to the frontal view. By comparing the difference between the transformed and incoming non-rigid control point, the facial motions are estimated.
4) ***Synthesis of facial motion***, facial (non-rigid) parameters are transmitted to the receiver and use these parameters to deform the WFM.
5) ***Rigid motion recovery***, rigid parameters are transmitted to the receiver and use these parameters to translate and rotate the WFM for recovering the motion of target face.
6) ***Texture mapping,*** the received textures of each patch are obtained in the initial stage and they have been mapped back into the corresponding patches, such that the 3D FM can be generated.

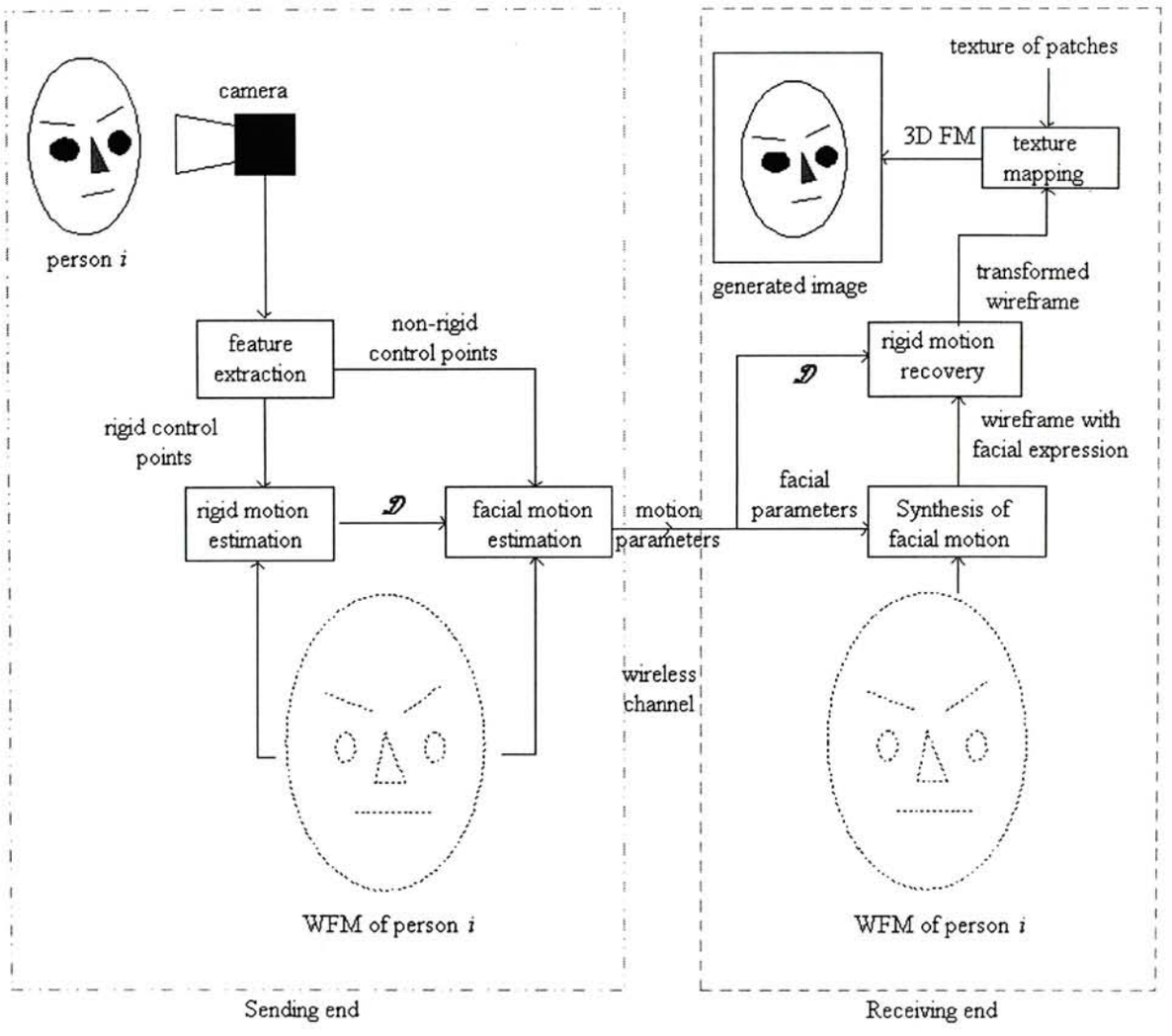


Figure 1.3: The block diagram of 3D model-based videoconferencing in real-time process

The Model-based novel motion estimation techniques has four main stages,

1. Feature extraction
2. Wireframe Model Fitting or Conformation
3. Pose Estimation
4. Facial Motion Estimation and Synthesis

In this thesis, the feature extraction is assumed known. In fact, the research of this part is the M.Phil project of another student.

1.3. Wireframe Model Fitting or Conformation

In 3D video conferencing, the human head is modeled by a highly detailed generic 3D WFM model consisting of a *triangulated mesh* (*patches*) of a WFM. However, the generic WFM may not be suitable for all users. The model should have slight adjustments to fit a particular user. Otherwise, that unchanged WFM cannot represent the human head in rigid and non-rigid motion correctly. As 3D model-based video coding relies on the modeling a generic 3D WFM to produce the facial image. The quality of the coding is highly dependent on the fitting of WFM onto the initial facial image.

If the users do not get their own 3D WFM, a set up process for.3D measurements will be made at short intervals. After the set up process, that user can be modeled by his WFM. The obtained rigid 3D feature points can allow us to estimate the 3D pose of the head and the non-rigid points will be used to detect the motion of facial features.

There are two ways to set up a WFM for matching an incoming human face: accurate measurement of all nodes or fitting a small set of nodes to the important feature points by using elastic surface. The latter is more practical and more convenient than the former one, it is very time-consuming to detect and estimate 3D coordinates of all the points from the surface. Also the measurement devices should be high precision [9,31], such as laser scanning and three-dimensional digitizers which are not easily available. Moreover, with the elastic surface approach, it is not necessary to calculate the information of all the nodes. We only need to get the data of the most essential nodes and then use them to interpolate the neighboring nodes. The mechanism is just like covering an elastic surface over the

given 3D points.

To fit an individual face by using interpolation, it requires less measurement on feature points but with lower accuracy on the WFM. The WFM is scaled and adjusted to correctly fit to a selected set of 3D points of that face. 3D points corresponding to facial features on eyes, mouth and nose are extracted as fixed points and the 3D WFM will be accurate fitted onto them. The original facial image is then texture mapped to the adjusted WFM. After the fitting of the WFM to the facial image is done during the initial set up stage, a 3D facial model is created which can be rotated and moved in any direction or deformed by facial parameters (non-rigid motion parameters).

By a study of the anatomy of human head, the generic WFM can be divided into several parts for better fitting to the corresponding facial regions. Before this, the size, width and length of WFM should be adjusted to match those of incoming face. Therefore, the method has two main steps, namely global and local deformations of WFM. The global deformation is used to deform the whole WFM, e.g. size and position of face while the local deformation is used to change the particular regions of WFM, e.g. size and position of right eye. In such way, if the 3D information of 26 selected feature points are available, they can be used to deform the WFM for fitting an incoming face.

1.4. Pose Estimation

In order to get the motion parameters, two types of motion have to be considered: the rigid motion (*pose of head*) and non-rigid motion (*facial motion*). Rigid motion does not deform the WFM while a non-rigid motion does. The non-rigid motions are also changed according to the rigid transformation of the head. Therefore, rigid motions should be known before the estimation of non-rigid motions. Rigid motion estimation of the head can be done by detecting at least four 3D points by stereo measurement. Hereafter, non-rigid motion corresponds to the motion of facial features at the eyes, mouth and nose.

In Pose estimation, it only considers the rigid motion of an object. There are four main approaches of 2D rigid motion estimation: feature based, optical flow based, block based and object matching. Most of the motion analysis algorithms estimate motion using the displacement vector field obtained by block matching techniques [4,6,26,34,39] and Kalman filter estimation [35]. This approach assumes that the motion is linear and translational. For 3D rigid motion estimation, Essential Matrix [25,38] can be used to find the relationship between two camera coordinate systems. By decomposing the Essential matrix, the rotation and translation between two camera systems can be obtained. However, this method is not robust and its error leads to ambiguous solution.

In this thesis, we proposed a novel Two Stage Algorithm [8], which is based on minimizing errors between the real image and the backprojected image. As such, it can estimate the affine transformation matrix, which gives an optimal estimation of a transformation between the two poses. This matrix is not constrained to the Euclidean translation and rotation of the head. Therefore, the rigid motion of

human head can be recovered and the image can be regenerated by using the obtained matrix.

1.5. Facial Motion Estimation and Synthesis

The function of the decoder in the 3D model-based coding system is to synthesize the output images of human face for different expressions. They are produced by deforming the 3D WFM with the received facial motion parameters. There are a number of ways to synthesize the facial movements on the 3D WFM. The two common ones are the "Clip and Paste"[1,26] and facial structure deformation methods. The Clip and Paste method is performed by extracting specific region (including texture) of the original image, these partial images include eyes and mouth. They will be compressed and transmitted to the receiver side. And then the transmitted parts will be decompressed and placed on the corresponding region of the synthesized image. The regions are identified on the translated head. However, the required bandwidth is still high because the transmitted partial images are quite large.

Recently, the most popular approach is the facial structure deformation method, which deforms the structure of the 3D facial model to simulate the facial expressions according to the facial action. The facial actions are described by using the Facial Action Coding System (*FACS*) [13,31], which was originally used in psychological studies and was a main approach on facial animation. *FACS* defines a set of minimal basic actions called *Action Units* (*AU*) performable on a human face such as inner brow raise, outer brow raise, etc. Each facial action can be decomposed as a combination of Action Units. There are 46 defined Action Units in the *FACS*. The deformation of the WFM is done through the movement of nodes according to the

deformation rules defined by the *AU*s. The deformation rules are defined by using physical and anatomical knowledge. After the deformation of WFM using these rules, the texture mapped WFM can regenerate the output image with the same expression. The *AU* corresponding to the changes in facial motion are estimated at the transmitter and the results are sent to the receiving end for synthesis of the new facial image

Another approach of facial structure deformation method is a Muscle-based method which is very similar to the *FACS*. The difference is that it only considers the direction of each muscle. The movement of nodes are limited by the directions of their connected muscles. If all muscles are defined and the non-rigid points are back-transformed to the frontal view for comparison, there is no need to recognize the facial action before the synthesis of facial image. Because all the connected muscles of these feature points are known, the motion of points will be constrained by the characteristic of *MU*s. So the movement of muscles can be estimated by the 2D motion of facial image points.

1.6. Thesis Outline

The WFM fitting is given in Chapter 2. Chapter 3 gives the theoretical background of epipolar geometry and proposed to use Essential Matrix to estimate the rigid motions between two poses of a head. Chapter 4 proposes two approaches to estimate the motion of target object. Chapter 5 is facial motion estimation and synthesis. Chapter 6 is the conclusion of this 3D model-based coding system and some proposed future improvements.

Chapter 2

Wireframe model fitting

In 3D video conferencing, the human head is modeled by a highly detailed generic 3D WFM model consisting of a *triangulated mesh* (*patches*) of a WFM, which is shown in Figure 2.1. However, the generic model may not be suitable for all users. Hence, the generic model should have slight adjustments to fit a particular user. Otherwise, that unchanged WFM cannot represent the human head in rigid and non-rigid motion correctly. Also, an elastic WFM is needed to fit the incoming human face.

There are two ways to obtain a 3D WFM: accurate measurement of all nodes and interpolate the WFM by some important feature points [27]. The latter is more practical and more convenient than high precision measurement devices [9,31], such as laser scanning and three-dimensional digitizers which are not easily available. Moreover, with the elastic surface approach, it is not necessary to calculate the 3D information of all the nodes. We only need to get the data of the most essential nodes and then use them to interpolate the neighboring nodes. The mechanism is just like covering an elastic surface over a given set of 3D points.

Undoubtedly, it is not necessary to use all the points or nodes to identify someone. Only the most important feature points are detected and needed in

undergoing recognition. Therefore, each human face is dominated by some common essential feature points and they are selected to be control feature points or fudicial points. If the 3D information of these control points are correct and the neighboring nodes are changed according to that of the control points, the reconstruction of WFM will become much simpler because approximately 26 points are used to interpolate over 300 nodes of 3D WFM.

2.1. Algorithm of WFM Fitting

In this thesis, the original generic WFM is obtained from University of Tokyo. It is shown in Figure 2.1. Twenty-six feature points are selected to deform the WFM and they are numbered in that figure. They are distributed on some important parts of the face, such as eyes, nose and mouth. Thus, the characteristics of the incoming face can be preserved in the WFM. The following table shows that five sets of points have been grouped to handle different regions of face.

Control feature points number	Target region
1, 2, 3, 4, 5, 6	outline of face
11, 12, 13, 14	organ 1 (right eye)
15, 16, 17, 18	organ 2 (left eye)
19, 20, 21, 22	organ 3 (nose)
23, 24, 25, 26	organ 4 (mouth)

Table 2.1: Definition of the correspondences between the human face and control point of the WFM

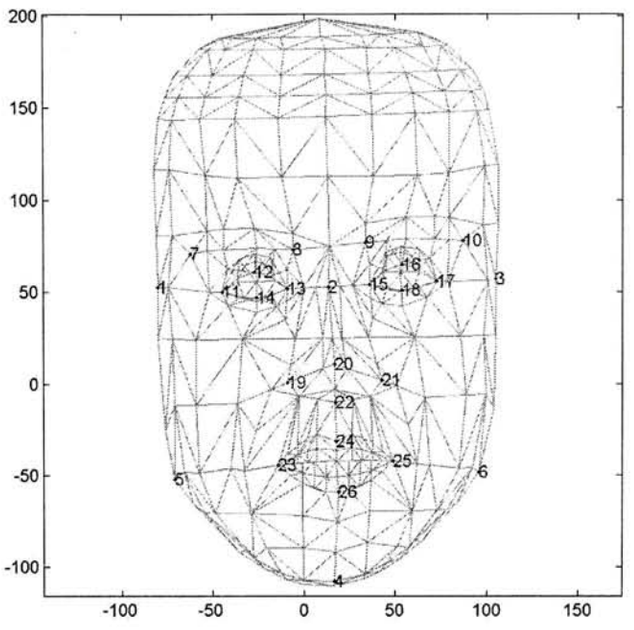


Figure 2.1: The generic WFM and selected control points

When doing WFM fitting, the 3D information of the control points should be known. Instead of using laser scanning, the 3D measurement can be carried out by the computer vision approach. There are several mutual methods to perform the 3D shape reconstruction [16,21,22,23,33]. In this way, the WFM is adjusted to fit those control points. The deformation process is divided into global and local deformation. The global deformation is used to deform the whole WFM, e.g. size and position of face. While the local deformation is used to change the particular regions of WFM, e.g. size and position of right eye. If the 3D information of 26 selected feature points are available, they can be used to deform the WFM for fitting an incoming face.

Table 2.2 shows the function of the twenty-six control feature points in the deformation process. For example, control pt. 1 and 3 are used to estimate the width of WFM.

Control feature points number	Function
1, 3	S_x: width of WFM
2, 4	S_y: length of WFM
1, 2	S_z: depth of WFM
2	offset point of WFM
mean of 11, 12, 13 and 14	$\boldsymbol{p}_{1cent}$: centroid of organ 1
11, 13	S_{1x}: width of organ 1
12, 14	S_{1y}: length of organ 1
11, 12	S_{1z}: depth of organ 1
mean of 15, 16, 17 and 18	$\boldsymbol{p}_{2cent}$: centroid of organ 2
15, 17	S_{2x}: width of organ 2
16, 18	S_{2y}: length of organ 2
16, 17	S_{2z}: depth of organ 2
mean of 19, 20, 21 and 22	$\boldsymbol{p}_{3cent}$: centroid of organ 3
19, 21	S_{3x}: width of organ 3
20, 22	S_{3y}: length of organ 3
19, 20	S_{3z}: depth of organ 3
mean of 23, 24, 25 and 26	$\boldsymbol{p}_{4cent}$: centroid of organ 4
23, 25	S_{4x}: width of organ 4
24, 26	S_{4y}: length of organ 4
23, 24	S_{4z}: depth of organ 4

Table 2.2: The function of the node in the WFM for fitting process

The elastic surface (*fine updating*) does not cover the feature points directly because the elastic surface cannot keep the global and local structure in the process. The WFM should be slightly modified such that the WFM can keep the structure of generic human face and fit it to match the incoming face.

2.1.1. Global Deformation

The global features, e.g. size and position of face, should be fitted before modifying the local features of WFM, so the general outline of face will be obtained.

2.1.1a. Scaling

The size of the face is different for different images and different people. So it

is necessary to resize the WFM to fit the incoming face.

$$ {}_1\boldsymbol{p}_i = \boldsymbol{S}_g \; {}_0\boldsymbol{p}_i \qquad \textit{eq.2.1} $$

$$ i = 1, 2, 3 \ldots . m. $$

where $\boldsymbol{S}_g$ is the scale matrix, $\boldsymbol{S}_g = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{pmatrix}$,

${}_0\boldsymbol{p}_i = ({}_0x_i, \; {}_0y_i, \; {}_0z_i)$ is feature point i of the WFM,

${}_1\boldsymbol{p}_i = ({}_1x_i, \; {}_1y_i, \; {}_1z_i)$ is feature point i of the scaled WFM,

m is number of node in WFM,

left subscript of $\boldsymbol{p}$ is used to indicate the step number.

2.1.1b. Shifting

As the human face is captured from different positions, the offset point should be defined. Then the incoming human face can be compared to the generic WFM in the same position to detect the difference between them. This offset point is selected to be control point no.2.

$$ {}_2\boldsymbol{p}_i = {}_1\boldsymbol{p}_i - \boldsymbol{p}_2' \qquad \textit{eq.2.2} $$

where $\boldsymbol{p}_2'$ is the control point no.2 of the incoming image, which is the offset point of the WFM.

2.1.2. Local Deformation

After global scaling and shifting, the positions of organs are also different between the WFM and an incoming face. Therefore, local deformation is needed to adjust the WFM to match each similar region to the incoming face image.

To have a concept of effective region, different parts of organ with different definition and updating equation should be worked out. There are four organs: the left eye, the right eye, the nose, and the mouth. Therefore, the feature points of WFM will be divided into 4 sub-groups (organs), one node can be belong to more than one subgroup.

$$\boldsymbol{p}_{ij} \subset \boldsymbol{p}_i \qquad \text{where } i = 1, 2, 3, \ldots, m$$

$$j = 1, 2, 3, 4$$

Each of these organs will have different effective regions which is defined by the window size. The point which is outside 2 times of window size will not be adjusted in the local deformation process.

$$\Delta \boldsymbol{D}_{ij} = \left| \boldsymbol{p}_{ij} - \boldsymbol{p}_{jcent} \right| \qquad \textbf{eq.2.3}$$

2.1.2a. Shifting

a) if $\Delta \boldsymbol{D}_{ij} \leq$ window size

$$_3\boldsymbol{p}_{ij} = {}_2\boldsymbol{p}_{ij} + \left(\boldsymbol{p}_{jcent}' - {}_2\boldsymbol{p}_{jcent} \right) \qquad \textbf{eq.2.4}$$

$$i = 1, 2, 3, \ldots..m.$$

$$j = 1, 2, 3, 4.$$

b) if window size $\leq \Delta \boldsymbol{D}_{ij} \leq$ 2×window size

$$_3\boldsymbol{p}_{ij} = {}_2\boldsymbol{p}_{ij} + {}_tg_j \cdot 2^{\frac{-\Delta \boldsymbol{D}ij}{\tau_j}} \left(\boldsymbol{p}_{jcent}' - {}_2\boldsymbol{p}_{jcent} \right) \qquad \textbf{eq.2.5}$$

$$i = 1, 2, 3, \ldots..m.$$

$$j = 1, 2, 3, 4.$$

where ${}_tg_j$ is the scale factor of organ j,

${}_2\boldsymbol{p}_{jcent}$ is the centroid of organ j after step 2 and $\boldsymbol{p}_{jcent}'$ is the centroid of input organ j.

2.1.2b. Scaling

a) if $\Delta \boldsymbol{D}_{ij} \leq$ window size

$$ {}_4\boldsymbol{p}_{ij} = {}_3\boldsymbol{p}_{jcent} + \boldsymbol{S}_j \left({}_3\boldsymbol{p}_{ij} - {}_3\boldsymbol{p}_{jcent} \right) \qquad \textit{eq.2.6} $$

$i = 1, 2, 3, \ldots..m.$

$j = 1, 2, 3, 4.$

b) if window size $\leq \Delta \boldsymbol{D}_{ij} \leq 2\times$window size

$$ {}_4\boldsymbol{p}_{ij} = {}_3\boldsymbol{p}_{jcent} + \boldsymbol{S}_j \cdot {}_s g_j \cdot 2^{\frac{-\Delta \boldsymbol{D}ij}{\tau_j}} \left({}_3\boldsymbol{p}_{ij} - {}_3\boldsymbol{p}_{jcent} \right) \qquad \textit{eq.2.7} $$

$i = 1, 2, 3, \ldots..m.$

$j = 1, 2, 3, 4.$

where $\boldsymbol{S}_j$ is the scale matrix of organ j, $\boldsymbol{S}_j = \begin{pmatrix} S_{jx} & 0 & 0 \\ 0 & S_{jy} & 0 \\ 0 & 0 & S_{jz} \end{pmatrix}$,

${}_s g_j$ is the scale factor of organ j.

2.1.3. Fine updating

Because the face is not exactly symmetric and the centroid of organ cannot represent the position well enough, fine updating is necessary for getting more accurate locations of nodes. An elastic surface with the following characteristics, is used for fine adjustment.

$$ {}_5\boldsymbol{p}_i = {}_4\boldsymbol{p}_i + {}_f g_j \cdot 2^{\frac{-\Delta \boldsymbol{d}ij}{\tau_j}} \left(\boldsymbol{p}_{jcon}{}' - \boldsymbol{p}_{jcon} \right) \qquad \textit{eq.2.8} $$

$i = 1, 2, 3, \ldots. m.$

$j = 1, 2, 3, \ldots. n.$

where $\Delta \boldsymbol{d}_{ij} = \left| \boldsymbol{p}_i - \boldsymbol{p}_j \right|$,

n is the number of control points,

${}_f g_j$ is the scale factor of organ j.

As such, the movements of nodes will follow the displacement of control points and the distance between them. So the displacement will be larger if the distance is smaller.

2.2. Steps of fitting:

1) The size of WFM was changed by using ***eq.2.1***.
2) The WFM was translated to fit the position of incoming face by using ***eq.2.2***.
3) The facial organs of WFM were translated to match the location of input data by using ***eq.2.4*** and ***eq.2.5***.
4) The facial organs were resized by using ***eq.2.6*** and ***eq.2.7***.
5) The coordinates of nodes were further adjusted by using ***eq.2.8***.

2.3. Function of Different Deformations:

Scaling and Shifting:

1) With local scale and position for particular region

The size of the face and position of organs are not the same for every person. So scaling and shifting process is necessary to modify the size and position of WFM for fitting the incoming face and its facial organs.

2) Avoid wrong order of changed node for large change of control points

Undoubtedly, if the size and position are too different from those of the WFM, the displacement or modification will be larger. Since the WFM is only an elastic surface, it cannot keep the relative shape if the changes are too large. Therefore, the whole local structures must move to target position before covering the elastic surface (fine updating).

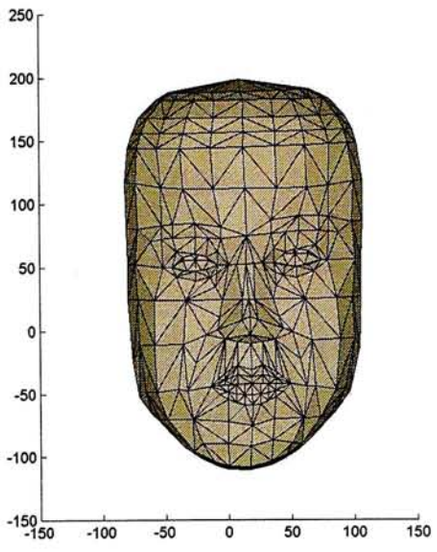
Fine Updating:

Getting local features from incoming face

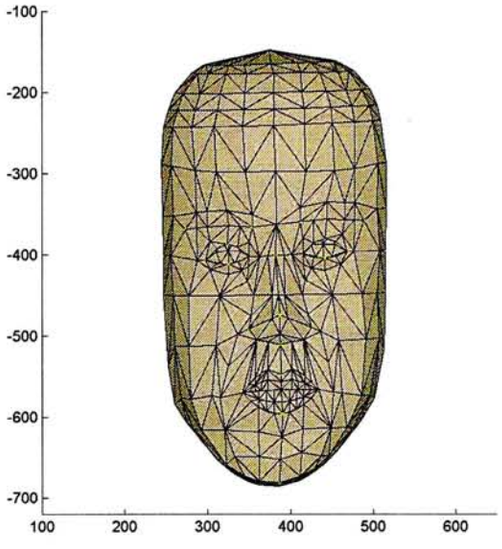
After scaling and shifting process, the nodes are almost located on the target positions. However, if the shape of the incoming face and their organs are not the same as the WFM, for example the incoming upper lip is thicker than the lower lip but the lips of the wireframe are of same thickness. These distinguishable features will be lost because only the centroids of face and facial organs will be used in the global and local deformation processes. Thus, the wireframe will not get these important features from the incoming face without the Fine Updating process.

2.4. Experimental Results:

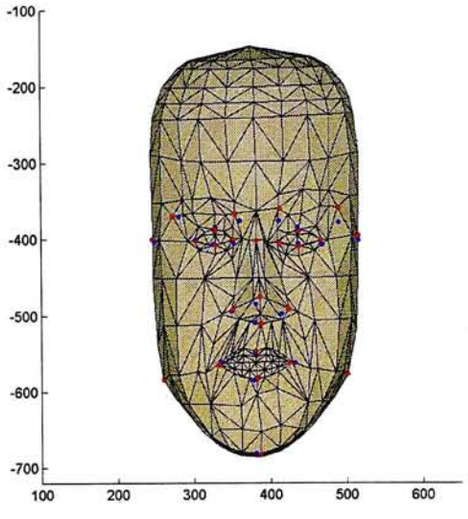
2.4.1. Output WFM in each step:



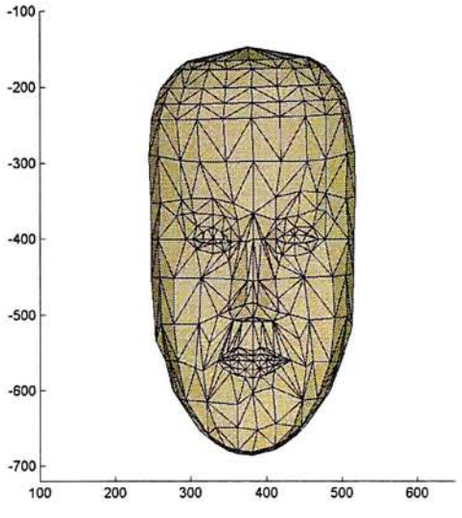
a) Original WFM



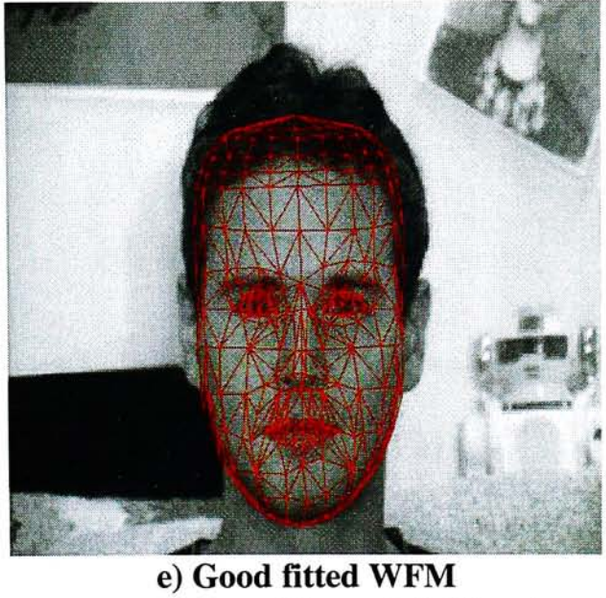
b) After 2.1.1



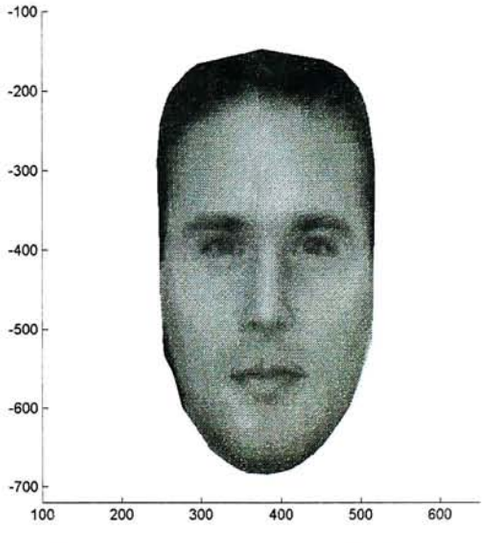
c) After 2.1.2



d) After 2.1.3



e) Good fitted WFM

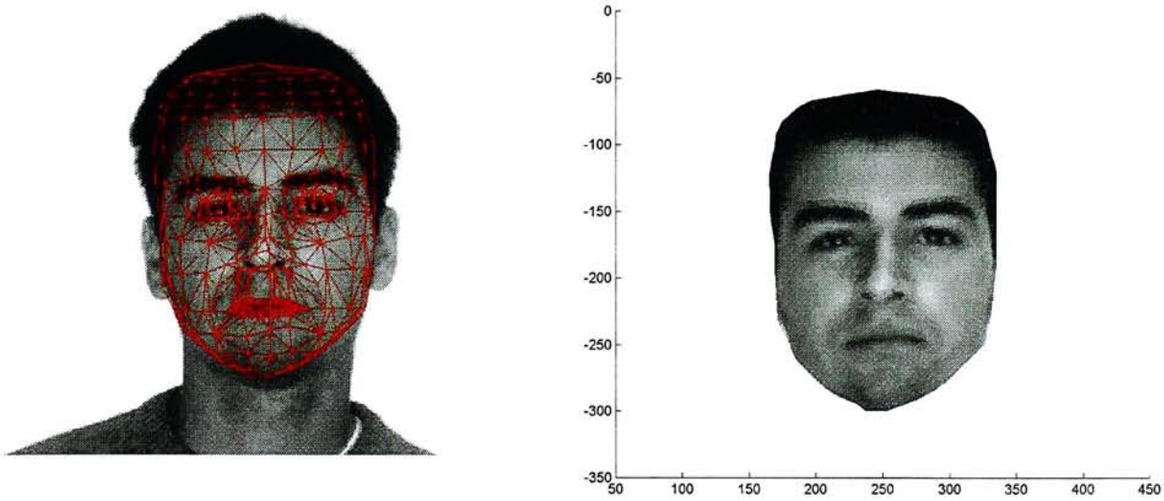


f) Texture mapping image

Figure 2.2: The step changes of WFM fitting

In Figure 2.2, the image of a) is the original generic WFM, which is obtained from University of Tokyo. The target face is shown in e). After the global deformation, the output WFM is shown in b). Since no characteristics can be extracted for the incoming face, b) is not very similar to the real one. So the local deformation should be carried out and get some local information from the data. The result is shown in c) where the red points are the location of nodes after local deformation, while the blue points are the target position of nodes of the incoming data. Therefore, the WFM still needs further adjustments to match the facial organs because the local deformation process is carried out based on the centroids of organs only. The final one is shown in Figure 2.2d). The fitted WFM is drawn back into the original image, it is shown in Figure 2.2e) and the corresponding FM is shown in Figure 2.2f).

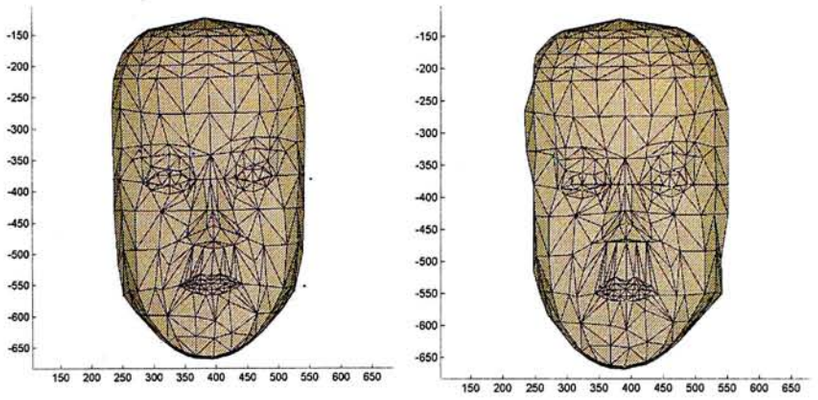
Figure 2.3 is another example of WFM fitting. After the textures are picked up by using the good fitted WFM in a), the FM is obtained and shown in Figure 2.3b).



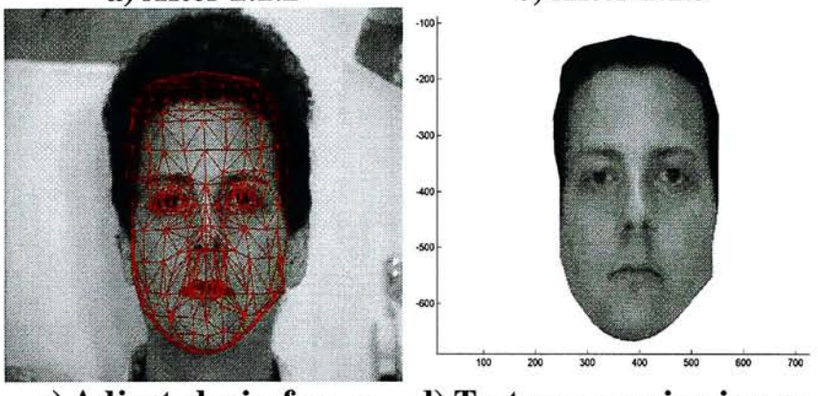
a) Good fitted WFM **b) Texture mapping image**

Figure 2.3: Other examples of WFM fitting

2.4.2. Examples of mis-fitted WFM with the incoming image



a) After 2.1.2 **b) After 2.1.3**



c) Adjusted wireframe **d) Texture mapping image**

Figure 2.4: An example of misfitted wireframe

However, the WFM fitting process cannot keep the local structures of WFM, e.g. curvature, relative position and outline of shape. So the nodes of WFM may not be lying on the edge of the corresponding position. Therefore, it is necessary to use more feature points to model some complicated or curve surfaces. As such, it can ensure that the nodes of WFM can be distributed on the contour of that object image and the local structure of facial organs can be preserved.

2.4.3. Fitted 3D FM

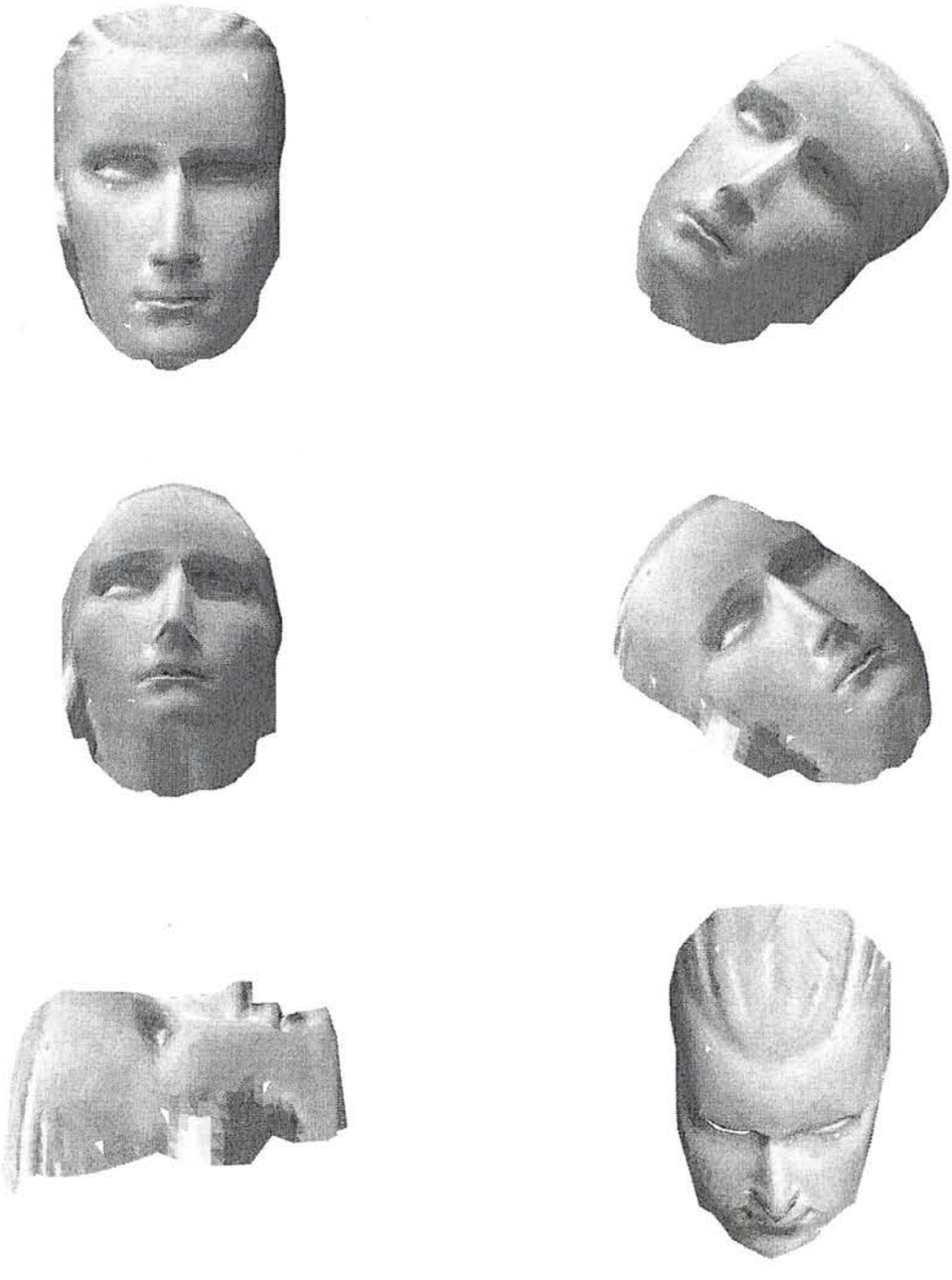


Figure 2.5: Different Views of 3D FM of Model 1

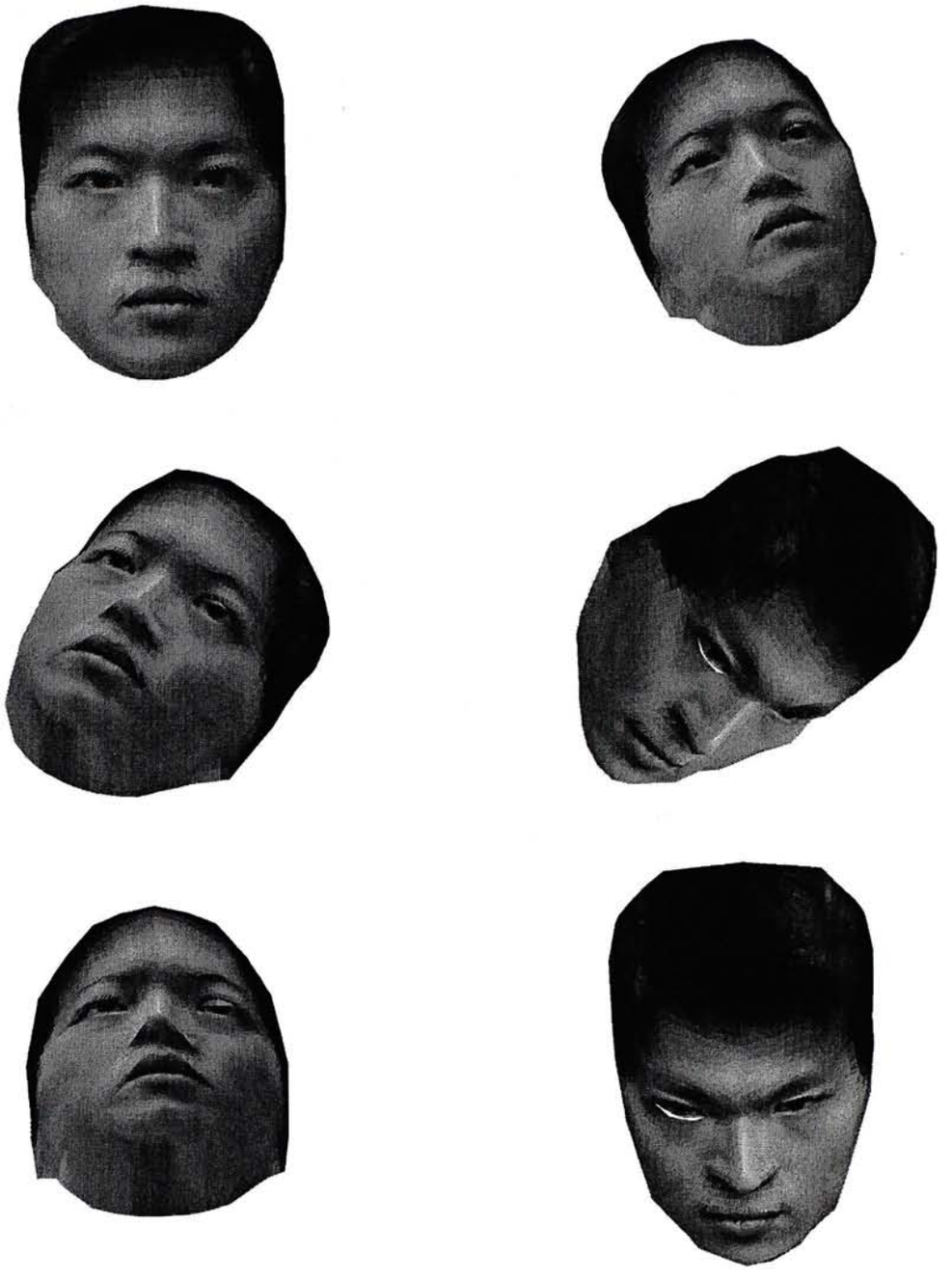
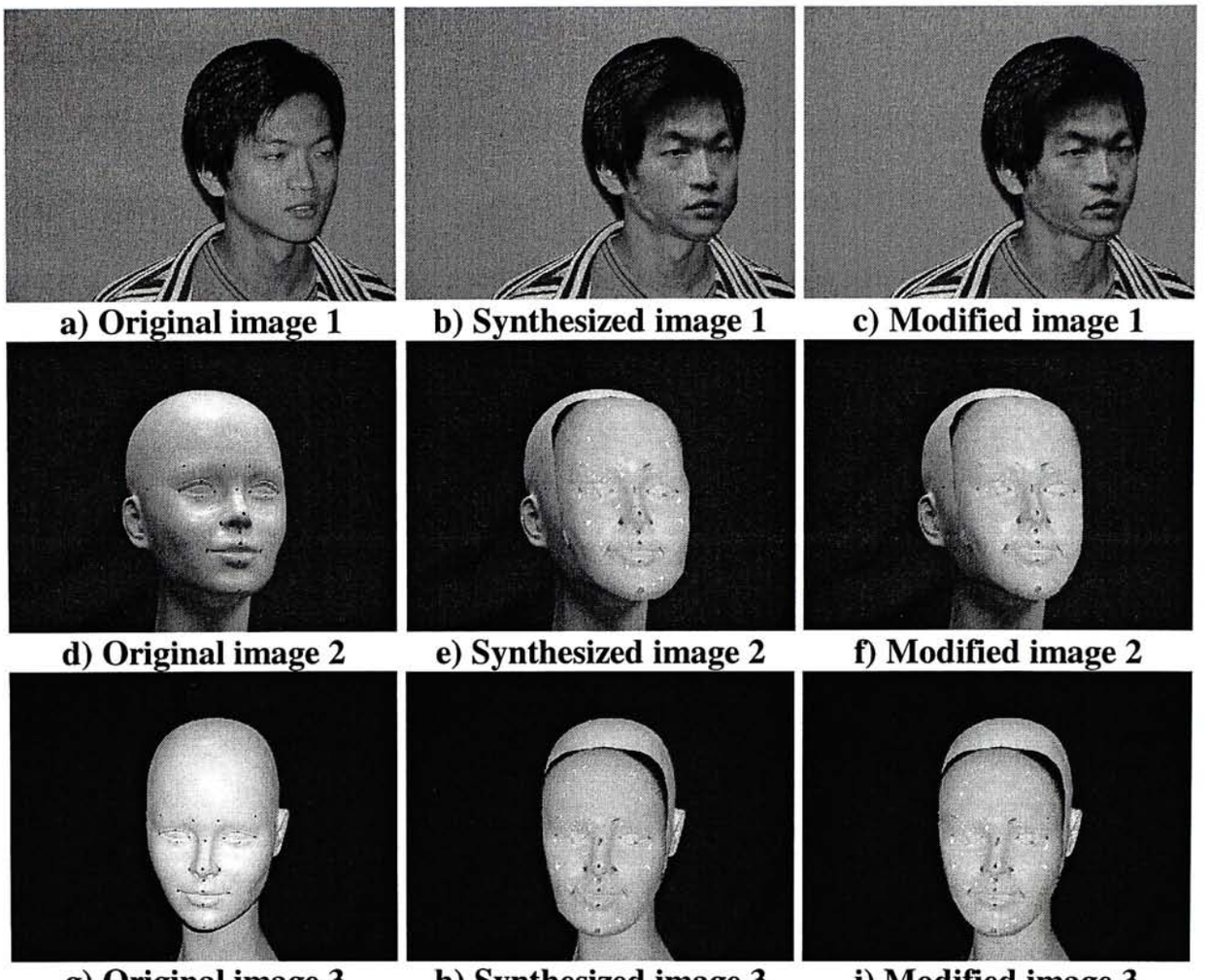


Figure 2.6: Different Views of 3D FM of Model 2

Figure 2.5 and Figure 2.6 are two examples of 3D WFM, they are constructed by using the algorithm, which is described in this chapter.

2.4.3. Effect of mis-fitted WFM after compensation of motion

If the 3D WFM is not good enough, the output image will also not similar to that of original image even the motions of object have been recovered. The following examples are mis-fitted WFM,



a) Original image 1 **b) Synthesized image 1** **c) Modified image 1**

d) Original image 2 **e) Synthesized image 2** **f) Modified image 2**

g) Original image 3 **h) Synthesized image 3** **i) Modified image 3**

Figure 2.7: The output image of mis-fitted WFM after compensation of 3D motion

In Figure 2.7, the left column is the original image, the middle one is the synthesized image. These synthesized images are quite different because the 3D FM does not match the 3D human head very well, especially the cheek. No control points can model this region during the fitting process and so the error in FM fitting will be enlarged after motion. If we want to synthesize the image, not only the 3D motion should be compensated, the fitted WFM is also important. After the synthesized image has slightly modified the cheek in the image manually, the modified image will look similar to the original one. Therefore, only feature points are not enough to interpolate the whole WFM of the human head.

2.5. Summary:

After WFM fitting process, the WFM can fit the facial organs of incoming face. So the generated 3D WFM is located on an acceptable position for these organs and twenty-six control 3D points can be used to interpolate a WFM with more than 300 nodes. Therefore, we can get the texture of patches from the corresponding position of the frontal view image. As such, all textures will be mapped to the patches of WFM and the generation of FM will be completed. It can be used for video-conferencing in the future.

However, the WFM fitting process cannot preserve the local structures of WFM, e.g. curvature, relative position and outline of shape. So the nodes of WFM may not be lying on the edge of the corresponding position. Therefore, it is necessary to use more feature points to model some complicated or curve surfaces. As such, it can ensure that the nodes of WFM can be distributed on the contour of that object image and the local structure of facial organs can be kept.

The most difficult part of WFM fitting is how to handle the nodes, which are located in the homogenous regions, e.g. cheek, forehead and chin. As no feature points can be detected in these regions, the fitting process cannot work in these parts. Therefore, the only way to do so is using the closest control points to model these homogenous regions. However, these interpolations cannot give a good fitted 3D WFM. Although the x- and y-coordinates in these regions are not very important, the z-coordinate of these nodes will affect the appearance significantly. Therefore, some other features instead of these points should be taken to handle these nodes.

Chapter 3

Epipolar Geometry

In this chapter, the concepts and mathematical representations of a camera model under full perspective projection will be introduced and described. A detail description of the epipolar geometry between two images taken by two different cameras or by a single camera at two different locations is also presented.

By using the concept of epipolar geometry, two sets of image points from two image frames can be used to estimate the rigid motion of the target object. The procedure is to calculate the Essential Matrix first and then decompose the matrix into the rotation and translation components. In such way, the pose estimation can be processed by using only two 2D images. Therefore, the obtained motion can be used for regeneration of new image.

3.1. Pinhole camera model and perspective projection

A pinhole camera model can accurately model the geometry and optics of modern CCD camera. A simple pinhole model [16,38] is shown in Figure 3.1.1,

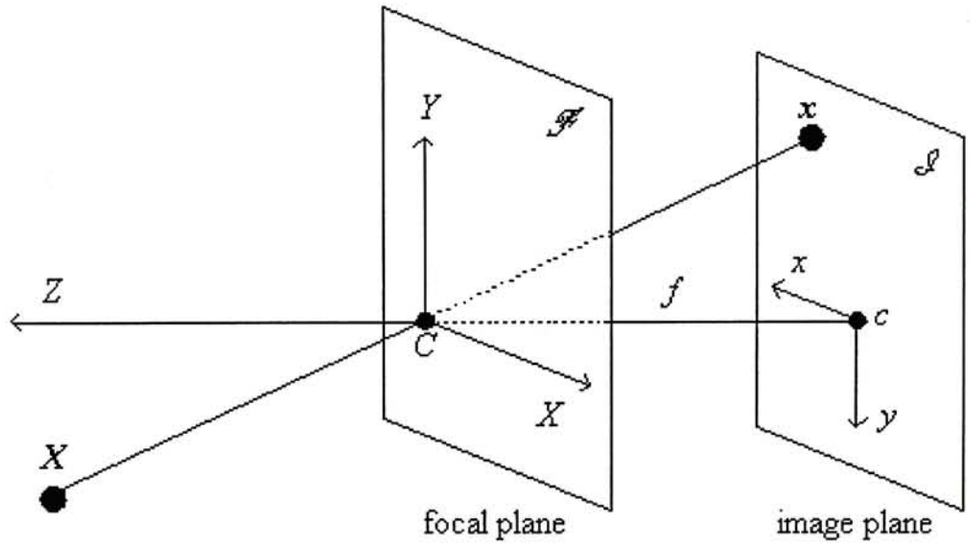


Figure 3.1.1: Pinhole camera model.

Consider a plane $\mathcal{F}$ at a fixed distance f in front of an image plane $\mathcal{I}$, where f is the focal length of the camera. An ideal pinhole C is found in the plane $\mathcal{F}$, which is parallel to $\mathcal{I}$ and is the *focal plane*. The rays of light emitted or reflected from point $\boldsymbol{X}$ goes through the pinhole and forms an inverted image at point $\boldsymbol{x}$ on the image plane. Each point in an object, its corresponding image point and the pinhole are on a straight line. This kind of projection from 3D space to a plane is called *perspective projection.*

In a pinhole camera model, the aperture of the camera is regarded as a point C called *optical center.* The line going through the optical center and perpendicular to the image plane is called *optical axis.* This intersection point c of the optical axis and the image plane is called the *principal point.*

A 3D coordinate system is chosen that the origin is located at the optical center

and the Z axis coincides with the optical axis. X–axis and Y–axis are parallel to the image coordinate x and y but in the opposite direction. The 2D coordinate system in the image plane has the origin located at the principal point c.

It is equivalent in geometry to replace the image $\mathscr{I}$ plane by a virtual image plane $\mathscr{I}'$ on the left hand side of the optical center with the x–axis and y–axis in the same direction with X–axis and Y–axisrespectively, as is shown in Figure 3.1.2.

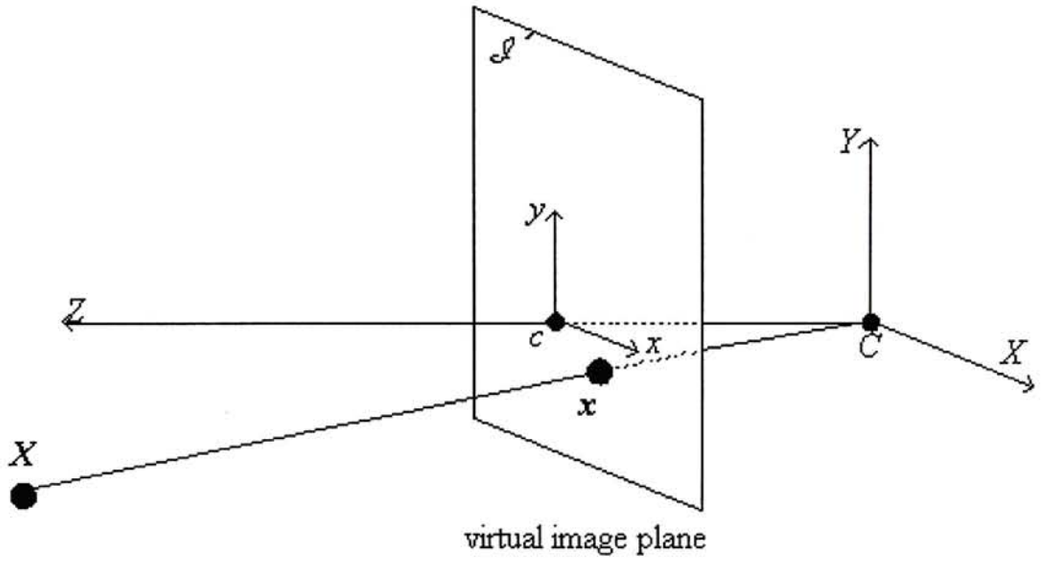


Figure 3.1.2: Virtual image plane in pinhole camera model

According to the geometry property of similar triangles, we can derive easily that a 3D point $\boldsymbol{X}$ with the coordinate (X, Y, Z) will be projected to point $\boldsymbol{x}$ (x, y) on the image that satisfies

$$\frac{x}{X} = \frac{y}{Y} = \frac{f}{Z} \qquad \textit{eq.3.1.1}$$

Given the coordinates of a 3D point and the focal length of the camera, the projected position in the image plane can be calculated easily

$$x = \frac{f}{Z} \cdot X \qquad \textit{eq.3.1.2}$$

$$y = \frac{f}{Z} \cdot Y \qquad \textit{eq.3.1.3}$$

The equation above can be rewritten in a homogeneous matrix form as,

$$\begin{pmatrix} U \\ V \\ S \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

where $x = \frac{U}{S}$ ***eq.3.1.4***

$y = \frac{V}{S}$ ***eq.3.1.5***

(U, V) is non-normalized image point,

S is a scale factor.

From the equation above we can see that an ideal pinhole camera model is a multiple-to-one mapping from 3D world to 2D image. Several 3D points can be projected to the same point in the image plane. Since this is not a one-one mapping, other information have to be used to infer the 3D location from the 2D image.

3.2 Concepts in Epipolar Geometry

The epipolar geometry [16] exists between two camera systems, the general set up is shown in Figure 3.2.1,

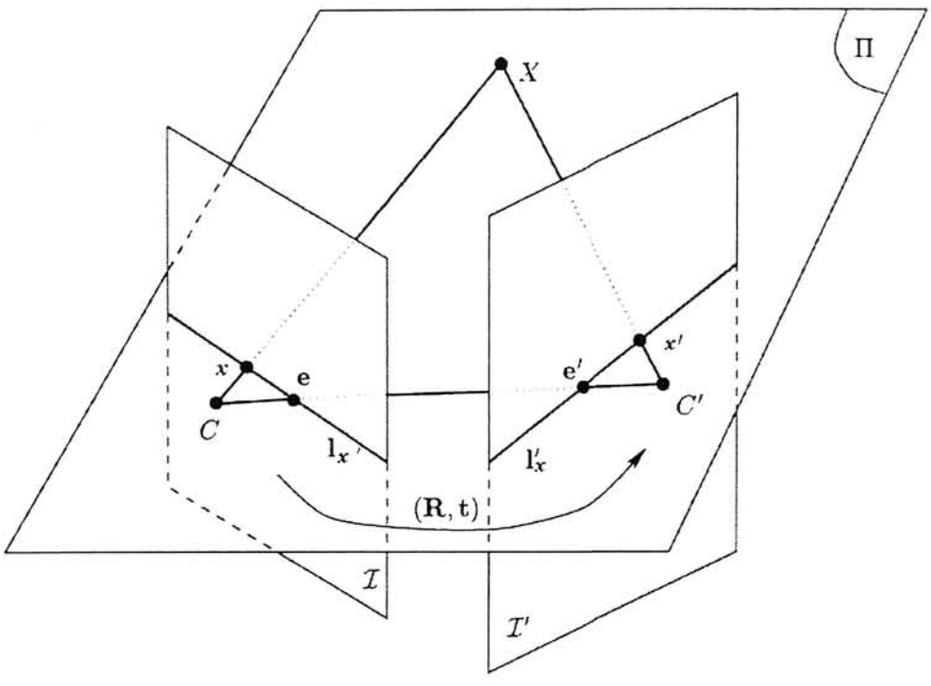


Figure 3.2.1: The Epipolar Geometry

Let C and C' be the *optical centers* of the first and second cameras respectively. Given a point $\boldsymbol{x}$ in the first image, its corresponding point $\boldsymbol{x'}$ in the second image is constrained to lie on a line called the *epipolar line* of $\boldsymbol{x}$, denoted by l'_x. The line l'_x is the intersection of the plane II (*epipolar plane*), defined by $\boldsymbol{x}$, C and C' with the second image plane I'. This is because image point $\boldsymbol{x}$ may correspond to an arbitrary point on the semi-line CX (X may be at infinity) and that the projection of CX on I' is the line l'_x. Furthermore, one observes that all epipolar lines of the points in the first image pass through a common point $\boldsymbol{e'}$, which is called the *epipole*. Epipole $\boldsymbol{e'}$ is the intersection of the line CC' with the image plane I'. This can be easily understood as follows.

For each point $\boldsymbol{x}_k$ in the first image I, its epipolar line l'_{xk} in I' is the intersection of the plane Π^k, defined by $\boldsymbol{x}_k$, C and C', with image plane I'. All epipolar planes Π^k thus form a pencil of planes containing the CC'. They must intersect I' at a common point, which is $\boldsymbol{e}'$. Finally, one can easily see the symmetry of the epipolar geometry. The corresponding point in the first image of each point $\boldsymbol{x}_k'$ lying on l'_{xk} must lie on the epipolar line $l_{x'k}$, which is the intersection of the same plane Π^k with the first image plane I. All epipolar lines form a pencil containing the epipole $\boldsymbol{e}$. The symmetry leads to the following observation.

If $\boldsymbol{x}$ and $\boldsymbol{x}'$ correspond to the projection of the single physical point $\boldsymbol{X}$ in space, then $\boldsymbol{x}$, $\boldsymbol{x}'$, C and C' must lie in a single plane. This is the well-known co-planarity constraint in solving motion and structure from motion problems when the intrinsic parameters of the cameras are known.

The computation significance in matching different views is that for a point in the first image, its correspondence in the second image must lie on the epipolar line in the second image and then the search space for a correspondence is reduced from 2 dimensions to 1 dimension. This is called the *epipolar constraint*.

If the line linking the two optical centers is parallel to one or both of the image planes, the epipole in one or both of the images goes to infinity and the epipolar lines are parallel to each other. Additionally, if the line linking the two optical centers is parallel to the horizontal scanlines of the cameras, the epipolar lines become horizontal too. This is the assumption made by many stereo algorithms.

Assume that the second camera is brought from the position of the first camera

through a rotation $\boldsymbol{R}$ followed by a translation $\boldsymbol{t}$. Thus any point (X, Y, Z) in the first camera coordinate system has coordinates (X', Y', Z') in the second camera coordinate system such that

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \boldsymbol{R} \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} + \boldsymbol{t} \qquad \textit{eq.3.2.1}$$

where $$\boldsymbol{R} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \textit{ and } \boldsymbol{t} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}$$

$\boldsymbol{R}$ has 9 components but there are only 3 degrees of freedom. There are 6 constraints on $\boldsymbol{R}$. Indeed, a rotation matrix $\boldsymbol{R}$ must satisfy,

$$\boldsymbol{R}\,\boldsymbol{R}^{\mathrm{T}} = \boldsymbol{I} \qquad \textit{eq.3.2.2}$$

and $$\det(\boldsymbol{R}) = 1. \qquad \textit{eq.3.2.3}$$

In the following, the epipolar equation was derived with the normalized image coordinates, then extended to include the pixel image coordinates. The unit of normalized image coordinates is same as that of focal length, so they are in mm. While pixel image coordinates are in pixel unit.

3.2.1. Working with normalized image coordinates

Let two images of a space point $\boldsymbol{X} = (X, Y, Z)^{\mathrm{T}}$ be $(x, y, 1)^{\mathrm{T}}$ and $(x', y', 1)^{\mathrm{T}}$ in the first and second normalized images respectively. They are denoted by $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}'$. Let $\boldsymbol{X'} = (X', Y', Z')^{\mathrm{T}}$ be the coordinates of the same space point in the second camera coordinate system. From the pinhole model, we have

$$\tilde{\boldsymbol{x}} = \boldsymbol{X} / Z$$

$$\tilde{\boldsymbol{x}}' = \boldsymbol{X}' / Z'$$

By eliminating the structure parameters $\boldsymbol{X}$ and $\boldsymbol{X'}$ using ***eq.3.2.1***, we obtain

$$\tilde{\boldsymbol{x}} = \frac{1}{Z}\left(Z' \boldsymbol{R}\, \tilde{\boldsymbol{x}}' + \boldsymbol{t}\right)$$

which still contains two unknown structure parameters Z and Z'. The cross product of the above equation with vector $\boldsymbol{t}$ yields

$$\boldsymbol{t} \times \tilde{\boldsymbol{x}} = \frac{Z'}{Z}\left(\boldsymbol{t} \times \boldsymbol{R}\, \tilde{\boldsymbol{x}}'\right)$$

Its dot product with $\tilde{\boldsymbol{x}}$ gives

$$\tilde{\boldsymbol{x}}^T \boldsymbol{t} \times \left(\boldsymbol{R}\, \tilde{\boldsymbol{x}}'\right) = 0 \qquad \textit{eq.3.2.4}$$

Here, the quantity Z'/Z has been removed.

Eq.3.2.4 is very important in solving motion and structure from motion. Geometrically, it is very clear. The three vectors CC', $C\tilde{\boldsymbol{x}}$ and $C'\tilde{\boldsymbol{x}}'$ are coplanar. When expressed in the first camera coordinate system, they are equal to $\boldsymbol{t}$, $\tilde{\boldsymbol{x}}$ and $\boldsymbol{R}\tilde{\boldsymbol{x}}'$ respectively. The co-planarity of the three vectors implies that their mixed product should be equal to 0, which gives ***eq.3.2.4***. Let us define a mapping $(.)_\times$ from a 3D vector to a 3×3 skew symmetric matrix:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}$$

Using this mapping, we can express the cross product of two vectors by the matrix multiplication of a 3×3 matrix and a 3-vector: $\boldsymbol{t} \times \tilde{\boldsymbol{x}} = (\boldsymbol{t})_\times \tilde{\boldsymbol{x}}$, $\forall \tilde{\boldsymbol{x}}$. ***Eq.3.2.4*** can then be rewritten as

$$\tilde{\boldsymbol{x}}^T \boldsymbol{E}\, \tilde{\boldsymbol{x}}' = 0, \qquad \textit{eq.3.2.5}$$

where $\boldsymbol{E} = (\boldsymbol{t})_\times \boldsymbol{R}$.

We call this equation the epipolar equation. Matrix $\boldsymbol{E}$ is known under the name of the *Essential Matrix*. It was first proposed by Longuet-Higgins for structure-from-motion [25]. The essential matrix is determined completely by the rotation and translation between the two cameras. Because $(\boldsymbol{t})_\times$ is skew symmetric, we have $\det((\boldsymbol{t})_\times) = 0$. Thus we have

$$\det(\boldsymbol{E}) = \det((\boldsymbol{t})_\times)\det(\boldsymbol{R}) = 0 \qquad \textit{eq.3.2.6}$$

3.2.2. Working with pixel image coordinates

If two points $\boldsymbol{x}$ and $\boldsymbol{x'}$, expressed in pixel coordinates in the first and second camera, are in correspondence, they must satisfy the following equation, which is derived from ***eq.3.2.5***,

$$\boldsymbol{u}^{\mathrm{T}}\,\boldsymbol{F}\,\boldsymbol{u}' = 0 \qquad \textit{eq.3.2.7}$$

where $$\boldsymbol{F} = \boldsymbol{A}^{-\mathrm{T}}\,\boldsymbol{E}\,\boldsymbol{A}'^{-1} \qquad \textit{eq.3.2.8}$$

$$\boldsymbol{A} = \begin{pmatrix} f\,k_u & f\,k_u\,/\,cos\theta & u_0 \\ 0 & f\,k_v\,/\,sin\theta & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

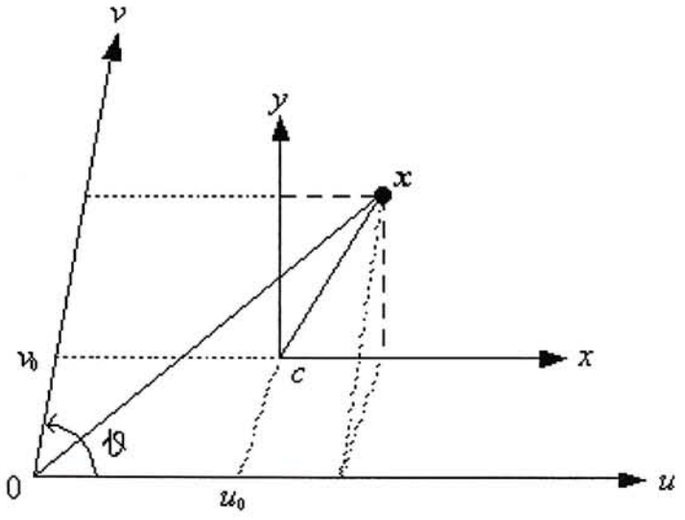


Figure 3.2.2: Camera intrinsic parameters

$\boldsymbol{A}$ and $\boldsymbol{A'}$ are respectively the *intrinsic matrix* of the first and second camera. The unit along the u- and v-axes are assumed to be k_u and k_v with respect to the unit of x- and y-axes. The u- and v-axes may not be exactly orthogonal, and θ is denoted as their angle. Let the coordinates of the principal point be $(u_0, v_0)^T$ The normalized coordinates x are related to the pixel coordinates $\boldsymbol{u}$ by

$$\tilde{\boldsymbol{x}} = \boldsymbol{A}^{-1}\,\tilde{\boldsymbol{u}} \qquad \textbf{\textit{eq.3.2.9}}$$

Plunging it into ***eq.3.2.5*** yields ***eq.3.2.7***. This is a fundamental constraint for two pixels to be in correspondence between two images.

As with the normalized image coordinates, the above ***eq.3.2.7*** can also be derived from the pinhole model. Without loss of generality, we assume that the world coordinate system coincides with the second camera coordinate system.

$$s\,\tilde{\boldsymbol{u}} = \boldsymbol{A}\begin{pmatrix}\boldsymbol{R} & \boldsymbol{t}\end{pmatrix}\begin{pmatrix}\boldsymbol{X'} \\ 1\end{pmatrix}$$

$$s'\,\tilde{\boldsymbol{u}}' = \boldsymbol{A}'\begin{pmatrix}\boldsymbol{I} & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{X'} \\ 1\end{pmatrix}$$

where s and s' are scale factors

The 3×3 matrix $\boldsymbol{F}$ is called the *fundamental matrix*. With this fundamental matrix, we can express the epipolar equation for two unnormalized images in the same normalized images. Since $\det(\boldsymbol{E}) = 0$,

$$\det(\boldsymbol{F}) = 0$$

$\boldsymbol{F}$ is of rank 2. Besides, it is only defined up to a scale factor. If $\boldsymbol{F}$ is multiplied by an arbitrary scalar, ***eq.3.2.7*** still holds. Therefore, a fundamental matrix has only seven degrees of freedom. There are only 7 independent parameters among the 9 elements of the fundamental matrix.

3.2.3. Summary

The essential matrix is the relationship between two images by using calibrated camera, the intrinsic matrix is known. So the image coordinates and the 3D coordinates are in the same scale. While the fundamental matrix is the relationship between two images by using uncalibrated camera, no information about the intrinsic parameters. The image coordinates is in pixel domain and the 3D coordinates are in mm domain.

3.3. 8-point Algorithm

The fundamental matrix is a basic tool for getting the relationship between two uncalibrated cameras, and the 8-point algorithm [20] is a frequently cited method for computing the fundamental matrix from a set of 8 or more point correspondences. The 8-point algorithm for computing the essential matrix [25] was introduced by Longuet-Higgins in 1981. In that paper the essential matrix is used to compute the structure of a scene from two views of calibrated cameras. The great advantage of the 8-point algorithm is that it is linear, hence fast and easily implemented.

Just as in the calibrated case, the fundamental matrix may be used to reconstruct the scene from two uncalibrated views [19], but in this case only up to a projective transformation. Apart from the scene reconstruction, the fundamental matrix may also be used for many other tasks, such as image rectification, computation of projective invariant, outliner detection and stereo matching;

3.3.1 Outline of the 8-point algorithm

The fundamental matrix is defined by the ***eq.3.2.7***

$$\boldsymbol{u'}^{\mathrm{T}}\,\boldsymbol{F}\,\boldsymbol{u} = 0 \qquad \textit{eq.3.3.1}$$

For any pair of matching points $\boldsymbol{u'} \leftrightarrow \boldsymbol{u}$ in two images. Given sufficiently many point matches $\boldsymbol{u}_i' \leftrightarrow \boldsymbol{u}_i$, this ***eq.3.3.1*** can be used to compute the unknown matrix $\boldsymbol{F}$. In particular, writing $\boldsymbol{u}_i = (u_i, v_i, 1)^{\mathrm{T}}$ and $\boldsymbol{u}_i' = (u_i', v_i', 1)^{\mathrm{T}}$ each point match gives rise to one linear equation in the unknown entries of $\boldsymbol{F}$. The coefficients of this equation can be expressed in the following form,

$$\begin{aligned} &u_i\,u_i'\,f_{11} + u_i'\,v_i f_{12} + u_i'\,f_{13} + u_i v_i'\,f_{21} + v_i v_i'\,f_{22} + v_i'\,f_{23} \\ &+ u_i f_{31} + v_i f_{32} + f_{33} = 0 \end{aligned} \qquad \textit{eq.3.3.2}$$

The row of the equation matrix may be represented as a vector

$$\left(u_i u_i',\ u_i' v_i,\ u_i',\ u_i v_i',\ v_i v_i',\ v_i',\ u_i,\ v_i,\ 1 \right) \qquad \textit{eq.3.3.3}$$

From all the point matches, we obtain a set of linear equations of the form

$$A f = 0 \qquad \textit{eq.3.3.4}$$

where f is a 9-vector containing the entries of the matrix F, and A is the equation matrix. The fundamental matrix F, and hence the solution vector f is defined only up to an unknown scale. For this reason, and to avoid the trivial solution f, we assume

$$\| f \| = 1 \qquad \textit{eq.3.3.5}$$

Under these conditions, it is possible to find a solution to the system with as few as 8 point matches. With more than 8 point matches, we have an over-determined system. Assuming the existence of a non-zero solution to this system of equations, we deduce that the matrix A must be rank-deficient. In other words, although A has 9 columns, the rank of A must be at most 8. We seek a least squares solution to this equation set. In particular, we seek the vector f that minimizes $\| Af \|$ subject to the constraint $\| f \| = 1$.

3.3.2. Modification of Fundamental Matrix under noisy data

An important property of the fundamental matrix is that it is singular, in fact of rank 2. Furthermore, the left and right null-space of F are generated by the vectors representing the two epipoles in the two images. Most applications of the fundamental matrix rely on the fact that it has rank 2. The matrix F found by solving the set of linear ***eq.3.3.4*** will not have rank 2, and matrix F should be modified to ensure this constraint. Matrix F is replaced by the matrix F' that minimizes the Frobenious norm $\| F - F' \|$ subject to the condition $\det F' = 0$.

A convenient method to do this is to use the Singular Value decomposition

(*SVD*). In particular, let $\boldsymbol{F} = \boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V}^{\mathrm{T}}$ be the SVD of $\boldsymbol{F}$, where $\boldsymbol{\Lambda}$ is a diagonal matrix $\boldsymbol{\Lambda} = diag(r, s, t)$ satisfying $r \geq s \geq t$. We let $\boldsymbol{F}' = \boldsymbol{U}\ diag(r, s, 0)\ \boldsymbol{V}^{\mathrm{T}}$.

3.3.3. Transformation of Image Coordinates

Image coordinates are sometimes given with the origin at the top-left of the image, and sometimes with the origin at the center. The question immediately occurs whether it makes a difference to the results of the 8-point algorithm for computing the fundamental matrix. More generally, to what extent is the result of the 8-point algorithm dependent on the choice of origin in the image.

3.3.3a. Translation to the mean of points

Suppose that image coordinates $\boldsymbol{u_i}$ in one image and $\boldsymbol{u_i}'$ in the other image are translated by the mean of $\boldsymbol{u}$ and $\boldsymbol{u}'$, the translated image points are $\hat{\boldsymbol{u}}$ and $\hat{\boldsymbol{u}}'$ .respectively. The expressions are as follows,

$$\hat{\boldsymbol{u}}_i = \boldsymbol{T}_t\ \boldsymbol{u}_i \qquad \textit{eq.3.3.6}$$

$$\hat{\boldsymbol{u}}_i' = \boldsymbol{T}_t'\ \boldsymbol{u}_i'$$

where $\boldsymbol{T}_t = \begin{pmatrix} 1 & 0 & \bar{u} \\ 0 & 1 & \bar{v} \\ 0 & 0 & 1 \end{pmatrix}$ and $\boldsymbol{T}_t' = \begin{pmatrix} 1 & 0 & \bar{u}' \\ 0 & 1 & \bar{v}' \\ 0 & 0 & 1 \end{pmatrix}$

Substituting in the ***eq.3.3.1***, we obtain

$$\hat{\boldsymbol{u}}'^{\mathrm{T}}\,\boldsymbol{T}_t'^{-\mathrm{T}}\,\boldsymbol{F}\,\boldsymbol{T}_t^{-1}\,\hat{\boldsymbol{u}} = 0$$

This relationship implies that $\boldsymbol{T}_t'^{-\mathrm{T}}\,\boldsymbol{F}\,\boldsymbol{T}_t^{-1}$ is the fundamental matrix corresponding to the point correspondences $\hat{\boldsymbol{u}}' \leftrightarrow \hat{\boldsymbol{u}}$.

3.3.3b. Normalizing transformation

As a first step, the coordinates in each image are translated so as to bring the centroid of the set of all points to the origin. The coordinates are also scaled. Rather than choosing different scale factors for each point, an isotropic scaling factor is chosen so that the u and v coordinates of the point are scaled equally. Finally, we choose to scale the coordinates so that the average distance of a point $\boldsymbol{u}$ from the origin is equal to $\sqrt{2}$. This means that the average point is equal to $(1,1,1)^T$.

$$\hat{\boldsymbol{u}}_i = \begin{pmatrix} \tilde{u} & 0 & 0 \\ 0 & \tilde{v} & 0 \\ 0 & 0 & 1 \end{pmatrix} \boldsymbol{u}_i \qquad \textit{eq.3.3.7}$$

$$\hat{\boldsymbol{u}}_i' = \begin{pmatrix} \tilde{u}' & 0 & 0 \\ 0 & \tilde{v}' & 0 \\ 0 & 0 & 1 \end{pmatrix} \boldsymbol{u}_i'$$

$$\textit{where}\quad \tilde{u} = \frac{1}{n}\sum_{i=1}^{n}\left|u_i - \bar{u}\right| \quad \textit{and} \quad \tilde{u}' = \frac{1}{n}\sum_{i=1}^{n}\left|u_i' - \bar{u}'\right| ,$$

$$\tilde{v} = \frac{1}{n}\sum_{i=1}^{n}\left|v_i - \bar{v}\right| \quad \textit{and} \quad \tilde{v}' = \frac{1}{n}\sum_{i=1}^{n}\left|v_i' - \bar{v}'\right| ,$$

n is the number of image points.

3.3.4 Summary of 8-point Algorithm

In summary the transformation is as follows,

1. The image points $\boldsymbol{u}_i$ and $\boldsymbol{u}_i$' are detected and translated by ***eq.3.3.6*** so that their centroid is at the origin.
2. The points are then scaled by ***eq.3.3.7*** so that the average distance from the origin is equal to $\sqrt{2}$.
3. The fundamental matrix $\hat{\boldsymbol{F}}$ corresponding to the matches $\hat{\boldsymbol{u}}_i' \leftrightarrow \hat{\boldsymbol{u}}_i$ is found by ***eq.3.3.4***.

4. Set $\boldsymbol{F} = \boldsymbol{T'}^{\mathrm{T}} \hat{\boldsymbol{F}} \boldsymbol{T}$,

where $\boldsymbol{T} = \begin{pmatrix} \tilde{u} & 0 & 0 \\ 0 & \tilde{v} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \overline{u} \\ 0 & 1 & \overline{v} \\ 0 & 0 & 1 \end{pmatrix}$,

$$\boldsymbol{T'} = \begin{pmatrix} \tilde{u}' & 0 & 0 \\ 0 & \tilde{v}' & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \overline{u}' \\ 0 & 1 & \overline{v}' \\ 0 & 0 & 1 \end{pmatrix}.$$

5. Decompose $\boldsymbol{F}$ into $\boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V}^{\mathrm{T}}$ by *SVD*.

6. If $\boldsymbol{\Lambda} = diag(r, s, t)$ where $r \neq s$ and $t \neq 0$, set $\boldsymbol{\Lambda'} = diag(r', s', 0)$

   where $r' = s' = 0.5*(r + s)$.

7. Set $\boldsymbol{F} = \boldsymbol{U}\,\boldsymbol{\Lambda'}\,\boldsymbol{V}^{\mathrm{T}}$.

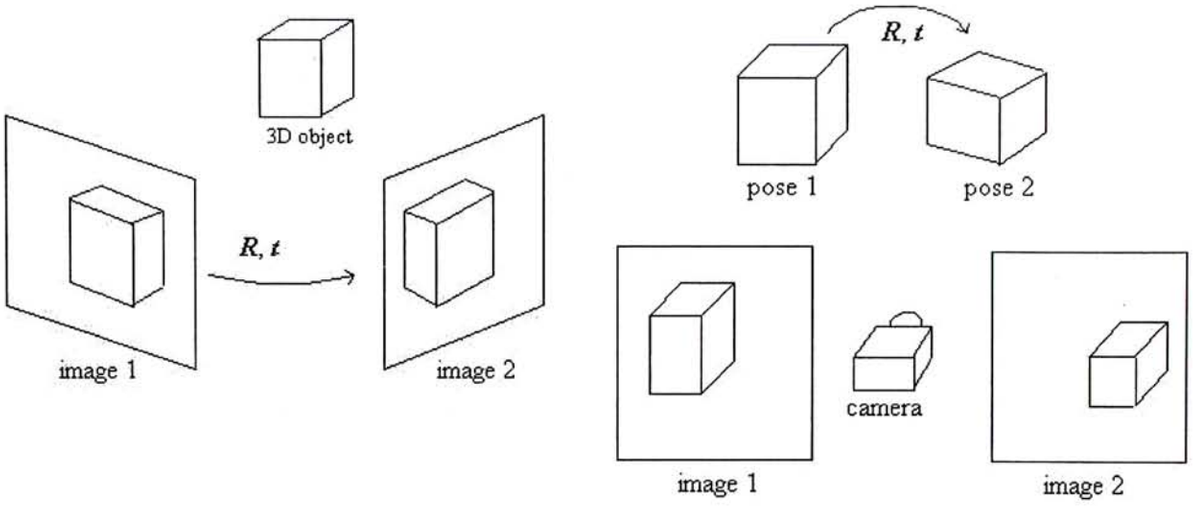
3.4. Estimation of Object Positions by Decomposition of Essential Matrix

A non-iterative algorithm to solve the problem of relative camera placement was given by Longuet-Higgins [25]. However, Hartley has derived the 8-point Algorithm [20] of Longuet-Higgins in order to fix notation and to gain some insights into its properties. Since it is dealing with homogenous coordinates, the result is only determined up to scale. By using the matrix obtained from 8-point Algorithm, the rotation and translation components can be decomposed from it. So the rigid motion of object will be estimated.

3.4.1. Algorithm Derivation

In the Hartley's paper [19], the target problem is that there are two uncalibrated cameras, one is situated at the origin of object space coordinates and the other is displaced from it. In my case, there are only one fixed calibrated camera and one moving object. The problem is almost the same and shown in Figure 3.4.1, two different views of image will be captured. But their transform is different and inverse. The two poses may be represented by the transformation that they perform translation and rotation of the 3D point.

When another pose i is needed to estimate, the $\boldsymbol{R}_i$, $\boldsymbol{t}_i$ is describing the transformation between the pose 1 and pose i. If the transform only considers the two successive poses, the error of the previous pose will be accumulated into the next pose. Therefore, it can minimize the accumulative error if we always compare between the pose 1 and pose i.



a) stationary object with moving camera **b) moving object with stationary camera**

Figure 3.4.1: Two similar target problem

$$(u,v,w)^{\mathrm{T}} = (x,y,z)^{\mathrm{T}} \qquad \textit{eq.3.4.1}$$

$$(u',v',w')^{\mathrm{T}} = \boldsymbol{R}\left((x,y,z)^{\mathrm{T}} + (t_x,t_y,t_z)^{\mathrm{T}}\right) \qquad \textit{eq.3.4.2}$$

where $\boldsymbol{R}$ is a rotation matrix, $(u, v, w)^{\mathrm{T}}$ and $(u', v', w')^{\mathrm{T}}$ are the homogeneous coordinates of the image points, and $(x, y, z)^{\mathrm{T}}$ and $(x', y', z')^{\mathrm{T}}$ are non-homogeneous object space coordinates. $\boldsymbol{t} = (t_x, t_y, t_z)^{\mathrm{T}}$ is the translation matrix between two poses.

$$(u,v,w)^{\mathrm{T}} = (\boldsymbol{I} \mid 0)(x,y,z,1)^{\mathrm{T}} = \boldsymbol{D}_1(x,y,z,1)^{\mathrm{T}} \qquad \textit{eq.3.4.3}$$

$$(u',v',w')^{\mathrm{T}} = (\boldsymbol{R} \mid \boldsymbol{R}\,\boldsymbol{t})(x,y,z,1)^{\mathrm{T}} = \boldsymbol{D}_2(x,y,z,1)^{\mathrm{T}} \qquad \textit{eq.3.4.4}$$

where $(\boldsymbol{I} \mid 0)$ and $(\boldsymbol{R} \mid \boldsymbol{R}\,\boldsymbol{t})$ are 3×4 matrices divided into a 3×3 block and a 3×1 column and $\boldsymbol{I}$ is a identity matrix.

In epipolar geometry, given a point $(u, v, w)^{\mathrm{T}}$ in image 1, the corresponding point $(u', v', w')^{\mathrm{T}}$ in image 2 must lie on a certain epipolar line, L. To determine this line, one may identify two points in L, namely the camera origin $(0, 0, 0, 1)^{\mathrm{T}}$ and the point at infinity, $(u, v, w, 0)^{\mathrm{T}}$. The images of these two points under $\boldsymbol{D}_2$ are $\boldsymbol{R}\,\boldsymbol{t}$ and $\boldsymbol{R}(u, v, w)^{\mathrm{T}}$ respectively and the line that passes through two points is given in homogeneous coordinates by the cross product,

$$(p,q,r)^{\mathrm{T}} = \boldsymbol{R}\boldsymbol{t}\times\boldsymbol{R}(u,v,w)^{\mathrm{T}} = \boldsymbol{R}\left(\boldsymbol{t}\times(u,v,w)^{\mathrm{T}}\right) \qquad \textit{eq.3.4.5}$$

where $(p, q, r)^{\mathrm{T}}$ represents the line $pu' + qv' + rw' = 0$. Representing by $\boldsymbol{S}$ the matrix

$$\boldsymbol{S} = \boldsymbol{S}_T = \begin{pmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{pmatrix} \qquad \textit{eq.3.4.6}$$

eq.3.4.5 can be rewritten as follows,

$$(p,q,r)^{\mathrm{T}} = \boldsymbol{R}\,\boldsymbol{S}\,(u,v,w)^{\mathrm{T}} \qquad \textit{eq.3.4.7}$$

As the point $(u', v', w')^{\mathrm{T}}$ with corresponding point $(u, v, w)^{\mathrm{T}}$, which must lie on the epipolar line,

$$(u',v',w')\,\boldsymbol{Q}\,(u,v,w)^{\mathrm{T}} = 0 \qquad \textit{eq.3.4.8}$$

where $\boldsymbol{Q} = \boldsymbol{R}\,\boldsymbol{S}$. This relationship is derived by Longuet-Higgins's paper [25].

Theorem 1

A 3×3 real matrix $\boldsymbol{Q}$ can be factorized as the product of a rotation matrix and non-zero skew symmetric matrix if and only if $\boldsymbol{Q}$ has two equal non-zero singular values and one singular value equal to 0.

Theorem 2

Suppose the matrix $\boldsymbol{Q}$ can be factorized into a product $\boldsymbol{R}\,\boldsymbol{S}$ where $\boldsymbol{R}$ is orthogonal and $\boldsymbol{S}$ is skew-symmetric. Let the Singular Value Decomposition of $\boldsymbol{Q}$ be $\boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V}^{\mathrm{T}}$, then up to scale factor the factorization is one of the followings:

$$\boldsymbol{S} \approx \boldsymbol{V}\,\boldsymbol{Z}\,\boldsymbol{V}^{\mathrm{T}} \qquad \textit{eq.3.4.9}$$

$$\boldsymbol{R} \approx \boldsymbol{U}\,\boldsymbol{E}\,\boldsymbol{V}^{\mathrm{T}} \quad \text{or} \quad \boldsymbol{U}\,\boldsymbol{E}^{\mathrm{T}}\,\boldsymbol{V}^{\mathrm{T}} \qquad \textit{eq.3.4.10}$$

$$\boldsymbol{Q} \approx \boldsymbol{R}\,\boldsymbol{S} \qquad \textit{eq.3.4.11}$$

where $\Lambda = diag(k, k, 0)$

$$\boldsymbol{E} = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{Z} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Furthermore $\|\boldsymbol{t}\| = 1$, *which is a normalized vector. Therefore, there are eight possible rotation / translation pairs that must be considered based on the four possible choices of* $\boldsymbol{R}$ *and two possible signs of* $\boldsymbol{t}$. *Therefore,*

$$\boldsymbol{U}\,\boldsymbol{E}\,\boldsymbol{V}^{\mathrm{T}}\,\boldsymbol{V}\,(0, 0, 1)^{\mathrm{T}} = \boldsymbol{U}\,(0, 0, 1)^{\mathrm{T}}$$

Theorem 3

In practical, the matrix $\boldsymbol{Q}$ *will not be factorized exactly in the required manner because of inaccuracies of measurement. In this case, the requirement will be to find the matrix closest to* $\boldsymbol{Q}$ *that does factor into a product of* $\boldsymbol{R}\,\boldsymbol{S}$.

Let $\boldsymbol{Q}$ *be any* 3×3 *matrix and* $\boldsymbol{Q} = \boldsymbol{U}\,\Lambda\,\boldsymbol{V}^{\mathrm{T}}$ *be its Singular Value Decomposition in which* $\Lambda = diag(r, s, t)$ *and* $r \geq s \geq t$.

Define the matrix $\boldsymbol{Q}' = \boldsymbol{U}\,\Lambda'\,\boldsymbol{V}^{\mathrm{T}}$, *where* $\Lambda' = diag(k, k, 0)$ *and* $k = (r + s) / 2$. *Then* $\boldsymbol{Q}'$ *is the matrix closest to* $\boldsymbol{Q}$ *in Frobenius norm which satisfies the condition* $\boldsymbol{Q}' = \boldsymbol{R}\,\boldsymbol{S}$, *where* $\boldsymbol{R}$ *is a rotation and* $\boldsymbol{S}$ *is skew-symmetric.*

3.4.2. Algorithm Outline

The algorithm for computing relative camera locations for calibrated cameras is as follows,

1. Find $\boldsymbol{Q}$ by using 8-point algorithm, which is shown in chapter 3.3.4.

2. Find the Singular Value Decomposition $\boldsymbol{Q} = \boldsymbol{U}\,\boldsymbol{\Lambda}\,\boldsymbol{V}^{\mathrm{T}}$, where $\boldsymbol{\Lambda} = diag(a, a, 0)$.
3. The transformation matrices for the two cameras are $\boldsymbol{D}_1 = (\boldsymbol{I} \mid 0)$ and $\boldsymbol{D}_2$ is equal to one of the eight following matrices,

$(\boldsymbol{U}\boldsymbol{E}\boldsymbol{V}^{\mathrm{T}} \mid \boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$ $\quad$ $(-\boldsymbol{U}\boldsymbol{E}\boldsymbol{V}^{\mathrm{T}} \mid \boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$

$(\boldsymbol{U}\boldsymbol{E}\boldsymbol{V}^{\mathrm{T}} \mid -\boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$ $\quad$ $(-\boldsymbol{U}\boldsymbol{E}\boldsymbol{V}^{\mathrm{T}} \mid -\boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$

$(\boldsymbol{U}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{V}^{\mathrm{T}} \mid \boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$ $\quad$ $(-\boldsymbol{U}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{V}^{\mathrm{T}} \mid \boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$

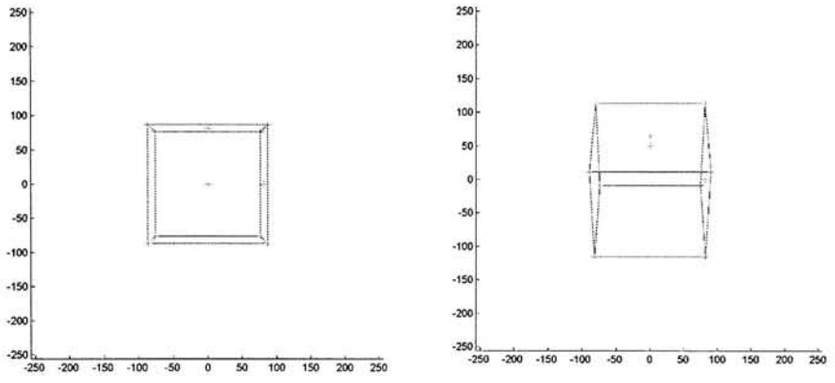
$(\boldsymbol{U}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{V}^{\mathrm{T}} \mid -\boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$ $\quad$ $(-\boldsymbol{U}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{V}^{\mathrm{T}} \mid -\boldsymbol{U}(0, 0, 1)^{\mathrm{T}})$

The choice between the eight transformations for $\boldsymbol{D}_2$ is determined by the requirement that the point locations must lie in front of both cameras.

3.5. Noise Sensitivity

By using the concept of the Essential Matrix and the technique of decomposition of the Essential Matrix, it can used to estimate the rigid motion of the target object. However, it is needed to know whether the method is good for motion estimation of human head and the performance under noise. Such that the noise sensitivity of this approach can be known.

In all experiments, a cubic is used to model the human head under simulation, the image size is 512×512 and the projected cubic is about 200×200 in pixel units. There are 11 feature points are chosen for undergoing the Hartley's method (decomposition of essential matrix). The corresponding cubic is shown in Figure 3.5.1,



a) Frontal view **b) Elevation view**

Figure 3.5.1: The image of cubic

3.5.1. Rotation vector of model

The following table shows the calculated angle and the rotation vector $(k_x, k_y, k_z)^T$ under the gaussian noise level is 0.5 pixel.

Actual case				Calculated case			
Angle	k_x	k_y	K_z	Angle	k_x	k_y	k_z
30	1	0	0	29.9963	0.9974	-0.0017	0.0005
-30	0	1	0	30.5259	-0.0153	-0.9975	-0.0025
40	0	0	1	39.9240	0.0105	0.0111	0.9967
20	1	0	1	20.7763	0.7117	-0.0005	0.6898
-50	1	1	1	49.6437	-0.5752	-0.5676	-0.5821

Table 3.5.1: The comparison between the real and calculated rotation vector

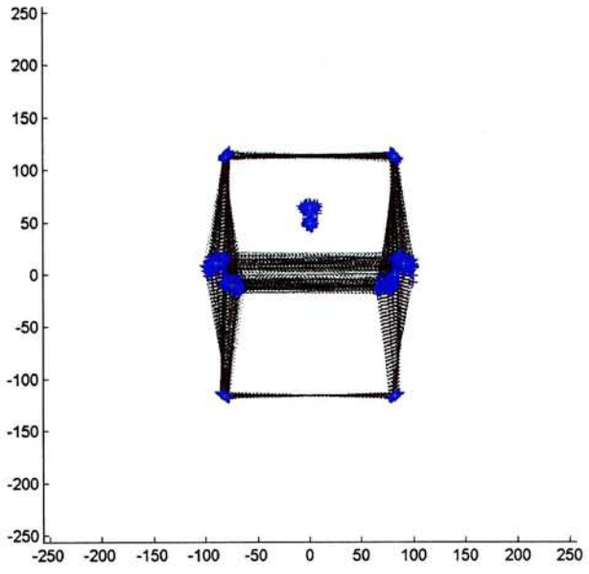
From the result, we know that the error of calculated angle is within 1 degree and the direction of the calculated vector is almost the same as the actual case.

3.5.2. The projection of rotated model

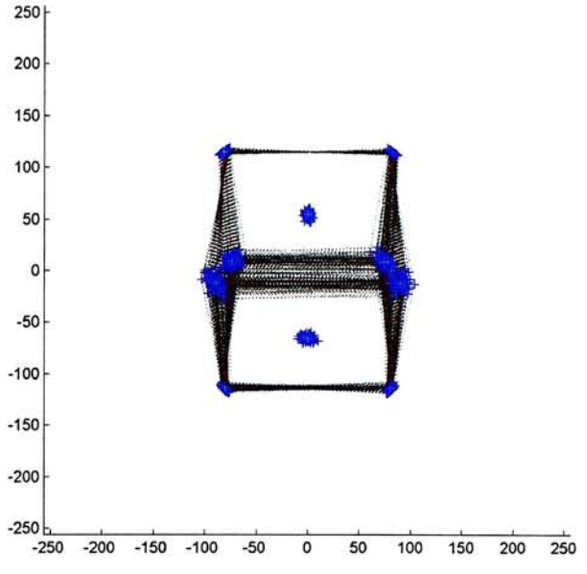
In this experiment, the 3D cubic was rotated for several degrees about three different axes, x-axis, y-axis and z-axis. Because this method can only give the direction of translation vector, no magnitude is involved. So no translation will be considered in the simulation.

In the simulated data, about 0.5 pixel gaussian noises are added to the image feature points. The results are obtained by repeating 100 times for adding different gaussian noise. The red lines are the actual image lines and the black dot lines are the calculated image lines. The green crosses are the actual image points and blue dots are the calculated image points

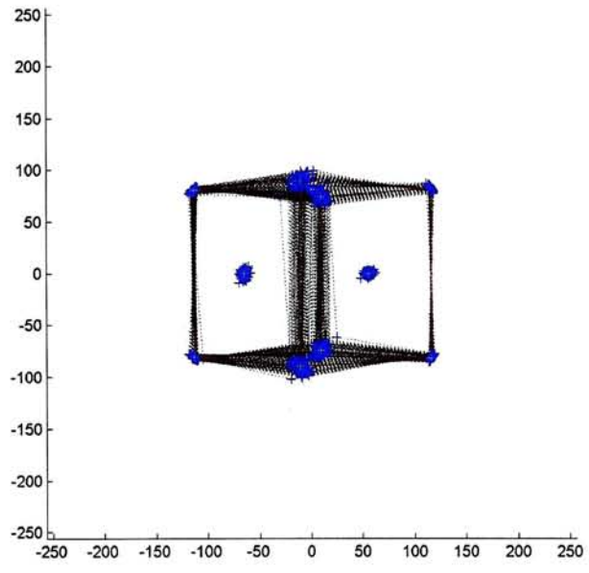
From the following output images, we know that the error of calculated image coordinates is within 5 pixel. Thus, when we use this method to generate the image for video conferencing, it can work for noise level which is under 0.5 pixel on image coordinates.



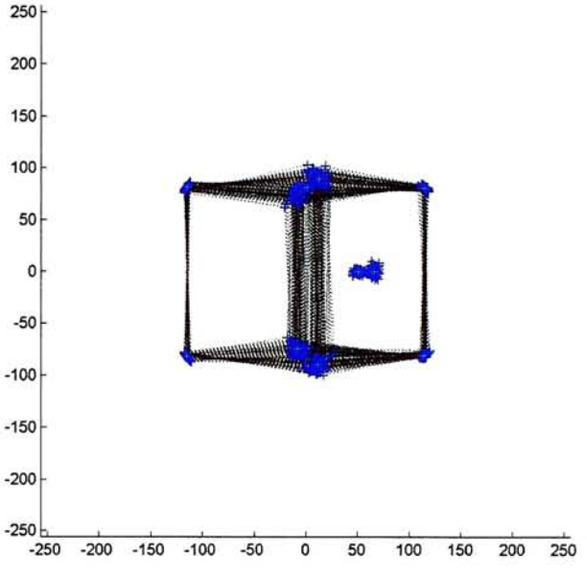
Rotate 50 about x-axis



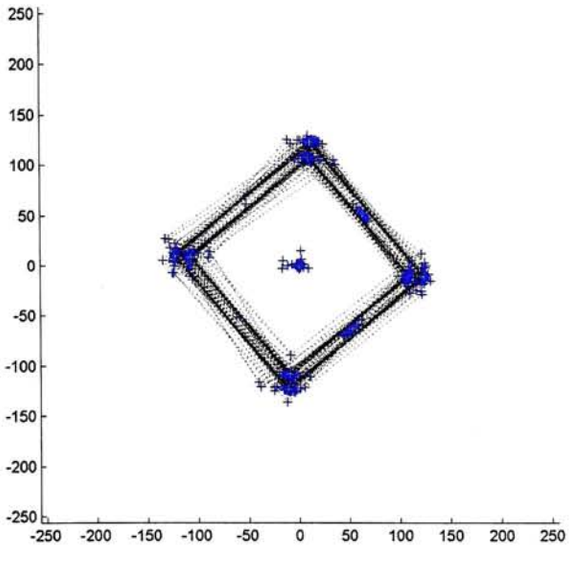
Rotate –50 about x-axis



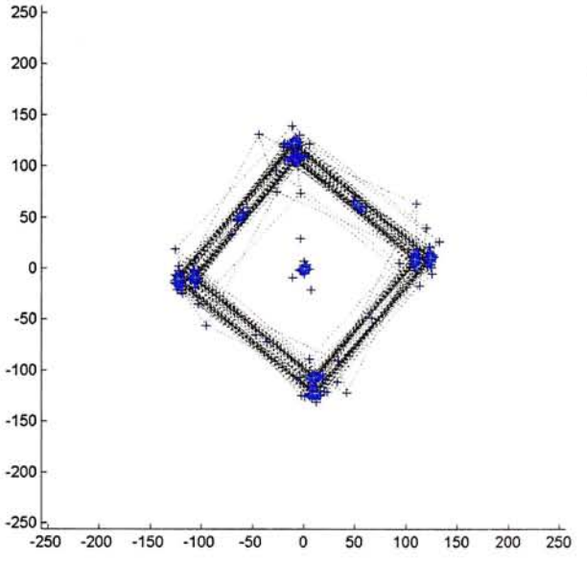
Rotate 50 about y-axis



Rotate –50 about y-axis



Rotate 50 about z-axis



Rotate –50 about z-axis

Figure 3.5.2: The distribution of the feature under different rotation axis

3.5.3. Noisy image

The following graph shows that the results of Hartley's method after adding noise to the simulated data. Each result is obtained by adding 100 times gaussian noise for one orientation and there are 100 different orientations, which are rotated between –50 and 50 degrees about the x, y and z axis for having different rotation angles and rotation axes.

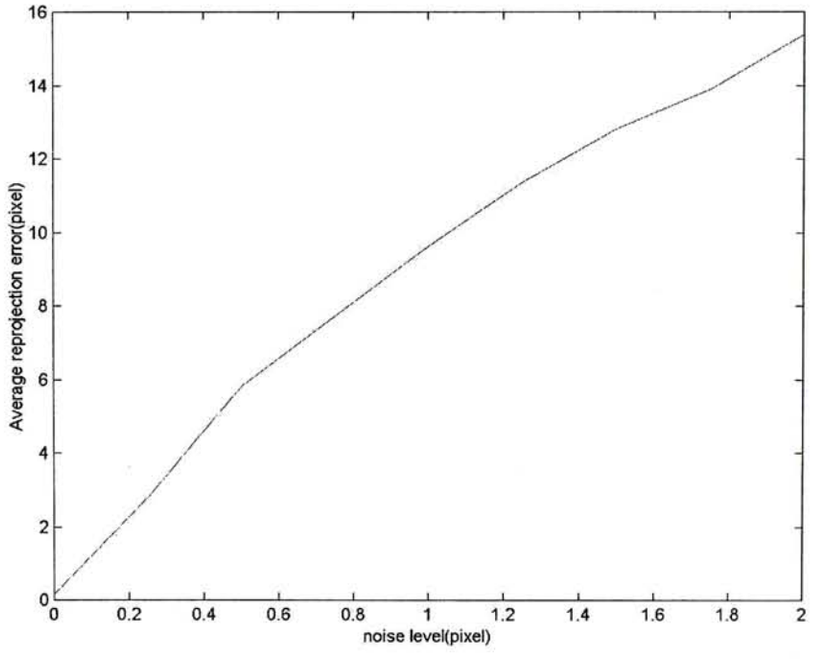


Figure 3.5.3: Performance of Hartley's method under different noise levels

The above result shows that if the image points with 1 pixel noise, the error of image coordinates is over 8 pixels. When it is compared to the size of 200×200 cubic, the error will be too large and cannot be accepted.

3.5.4. Summary

From the result of this chapter, we know that only using 2D image points to estimate the 3D motion is not very robust under noise. Because the essential matrix is too sensitive to noise, some other features should be introduced for improving the

performance. One approach is that the obtained 3D information of WFM can be used to generate the projection points and then check the error, such that the error can be minimized.

Weaknesses of the method on Decomposition of Essential Matrix

1. At least eight correspondences by using linear method

There are a limited number of rigid feature points on a human face. It is very difficult to search more than eight feature points to calculate the Fundamental Matrix and minimize the error by least square method.

2. Fundamental Matrix cannot be formed by planar points

The difference of depth of model is very small compared to the distance from the camera and model. Especially for the rigid control points, they are almost planar. These planar points cannot estimate the Fundamental Matrix correctly.

3. Translation vector without magnitude

The obtained translation vector is without magnitude. It is not enough to reconstruct the same image by this unit translation vector. It is because the position of object will also affect the view which can be seen under perspective projection.

An example is shown in Figure 3.5.4 where a cubic is placed on three poses. These three poses are without rotation, only with translation. But the projections of cubic from these three poses in the image plane are totally different, it is not only the translation problem, the views of cubic are also different.

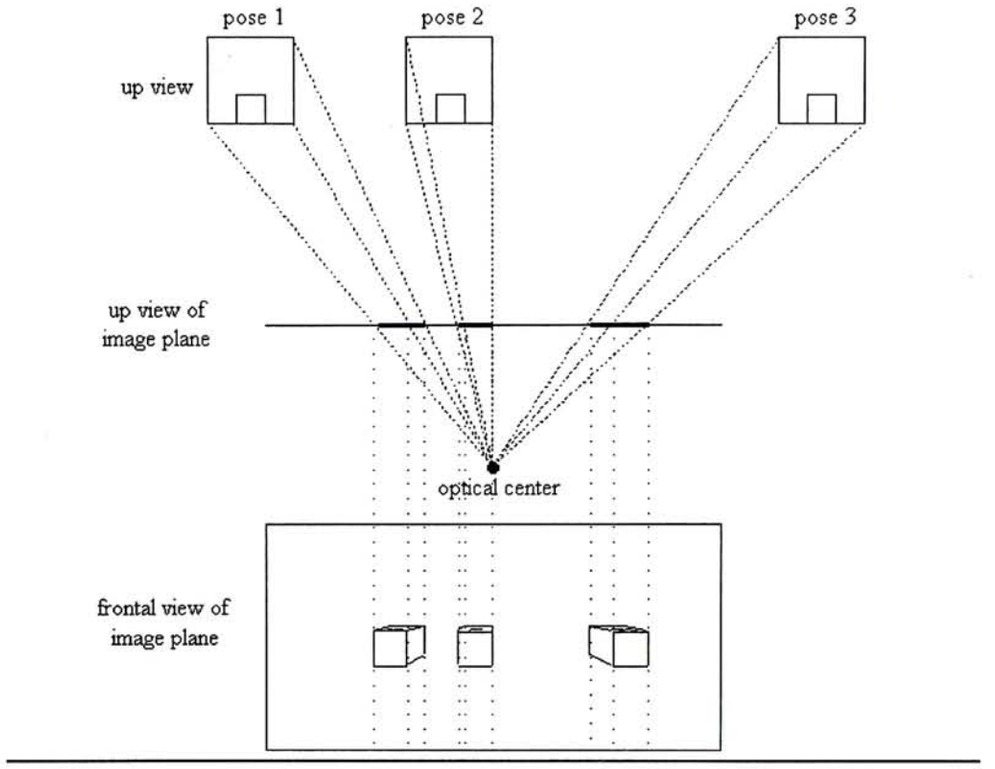


Figure 3.5.4: The perspective effect on translation of object

Chapter 4

Pose Estimation

After the process of WFM fitting, the next step is to detect the motion of target object and then transmit them to the receiver. In order to get the motion parameters, two types of motion have to be considered, rigid motion (pose) and non-rigid motion (facial deformation parameters). In this chapter, pose estimation technique will be proposed which is rigid motion of head. By using the information of 3D WFM and the incoming 2D rigid feature points, the changed pose can be detected and then the transformation matrix can be calculated. If the 3D WFM is transformed by this matrix to recover the motion, the regenerated image will have similar pose to that of the real one. The matrix is about 11 to 12 elements, so transmission of this matrix can save a lot of bandwidth.

There are two methods to do, Linear Method and Two Stage Algorithm. The obtained matrix from the Linear Method is a mapping transformation matrix while Two Stage Algorithm gives the affine transformation matrix.

4.1. Linear Method

4.1.1. Theory:

Due to the weaknesses of the decomposition of fundamental matrix [19,38], it is not practical to be used in video conferencing. If the 3D information of the model, which is shown in Figure 4.1.1, is obtained by using stereo vision [16], a simple linear method can be used instead of the decomposition of essential matrix method for estimating the transformation.

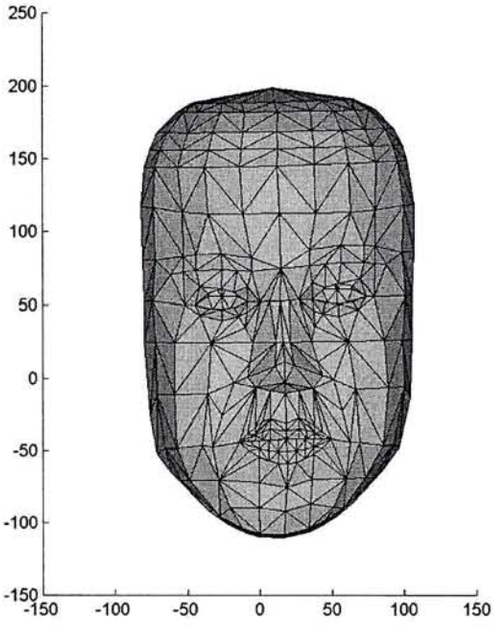


Figure 4.1.1: WFM of human head

The estimated mapping transformation matrix involves the rigid motion and projection of the object. If the WFM of the target head and the incoming image feature points are available, the mapping transformation matrix (*Projection matrix*) [38] can be found and the same synthesized image can be generated. The mechanism is shown in Figure 4.1.2,

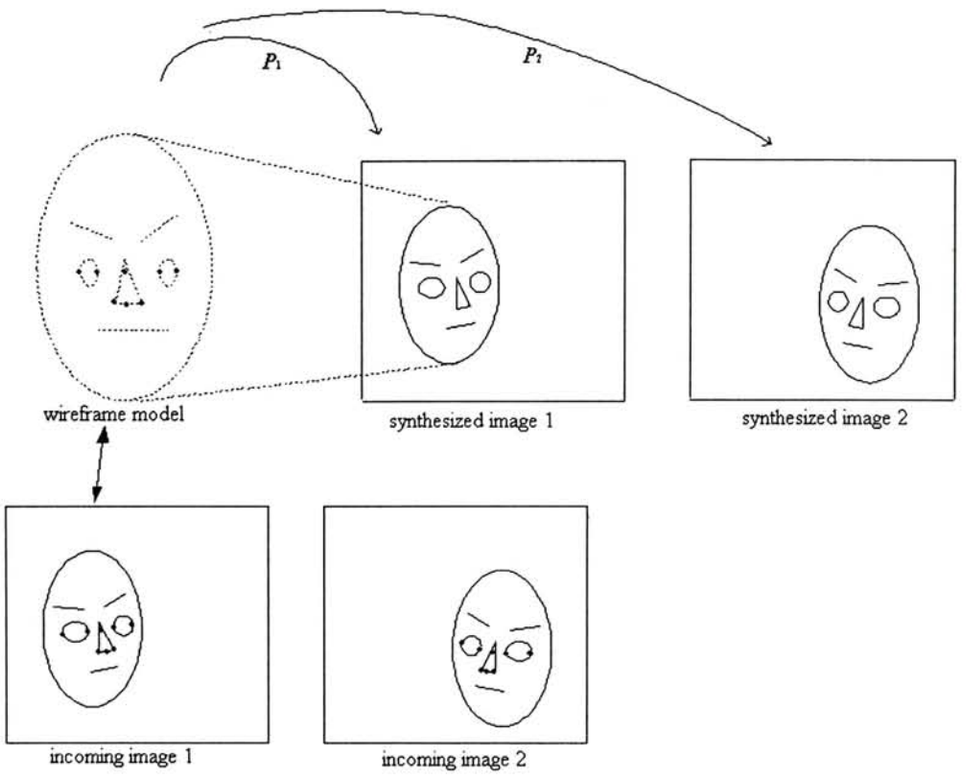


Figure 4.1.2: mechanism of the linear method

By comparing the difference between the control points from image 1 and WFM at pose 0. The projection matrix P_1 can be found and used to generate the image 1. Also, image 2 can be synthesized by using P_2. As such, a sequences of rigid motion can be estimated. The general form of projection matrix is,

$$\boldsymbol{P}\,\boldsymbol{X}_i = s\,\boldsymbol{x}_i'$$

$$\begin{pmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix} = s \begin{pmatrix} x_i' \\ y_i' \\ 1 \end{pmatrix} \qquad \textit{eq.4.1.1}$$

where $\boldsymbol{x}_i' = (x_i', y_i', 1)^{\mathrm{T}}$ is the image point of $\boldsymbol{X}_i'$ and $\boldsymbol{X}_i$ is the 3D point of the WFM without any motion. $\boldsymbol{P}$ is 3×4 projection matrix. By eliminating the scale factor s from the ***eq.4.1.1***, the first and second rows of the system should be divided by the third row, such that each corresponding image point will has only two equations,

$$P_{11}X + P_{12}Y + P_{13}Z + P_{14} - x_i'\left(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}\right) = 0$$

$$P_{21}X + P_{22}Y + P_{23}Z + P_{24} - y_i'\left(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}\right) = 0$$

If 6 corresponding points are detected, the above linear system can be solved. For the noisy data, the corresponding points should be larger than 6 because this linear method is not very robust under noise in general. The matrix form of the above linear system can be grouped and shown as follows,

$$\begin{pmatrix}
X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 1 & -x_1'X_1 & -x_1'Y_1 & -x_1'Z_1 & 1 \\
0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1'X_1 & -y_1'Y_1 & -y_1'Z_1 & 1 \\
X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 1 & -x_2'X_2 & -x_2'Y_2 & -x_2'Z_2 & 1 \\
0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 & -y_2'X_2 & -y_2'Y_2 & -y_2'Z_2 & 1 \\
X_3 & Y_3 & Z_3 & 1 & 0 & 0 & 0 & 1 & -x_3'X_3 & -x_3'Y_3 & -x_3'Z_3 & 1 \\
0 & 0 & 0 & 0 & X_3 & Y_3 & Z_3 & 1 & -y_3'X_3 & -y_3'Y_3 & -y_3'Z_3 & 1 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\
X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 1 & -x_n'X_n & -x_n'Y_n & -x_n'Z_n & 1 \\
0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -y_n'X_n & -y_n'Y_n & -y_n'Z_n & 1
\end{pmatrix}
\boldsymbol{p} =
\begin{pmatrix}
0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0
\end{pmatrix}$$

$$\boldsymbol{B}\,\boldsymbol{p} = 0 \qquad \textit{eq.4.1.2}$$

where $n \geq 6$ and $\boldsymbol{p} = \begin{pmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{21} & P_{22} & P_{23} & P_{24} & P_{31} & P_{32} & P_{33} & P_{34} \end{pmatrix}^{\mathrm{T}}$

The linear system will then be solved by Singular Value Decomposition.

4.1.2. Normalization:

If the image coordinates of the input points are without normalization, the error on feature detection will greatly affect the performance of ***eq.4.1.2***. It is because the matrix $\boldsymbol{B}$ with some columns includes image points. If the image points are too large when compared to the 3D points, the columns containing the image points will dominate the linear system and the ill-condition will occur. The same problem will also exist when the image points are too small. Therefore, if we want to avoid this

ill-condition problem, the input image points should be normalized [20] before solving the linear system. As a result, the scale of each column is almost the same. Then the solution will be more accurate and robust when it was compared to that without normalization one. The normalization of image point is as follows,

$$\tilde{\boldsymbol{x}}' = \begin{pmatrix} 1/\hat{x}' & 0 & 0 \\ 0 & 1/\hat{y}' & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & \bar{x}' \\ 0 & 1 & \bar{y}' \\ 0 & 0 & 1 \end{pmatrix} \boldsymbol{x}'$$

where $\hat{\boldsymbol{x}}' = \frac{1}{n}\sum_{i=0}^{n-1} |\boldsymbol{x}_i - \bar{\boldsymbol{x}}|$ is the mean distance and n is the number of correspondences.

$\tilde{\boldsymbol{x}}' = (\tilde{x}', \tilde{y}', 1)^{\mathrm{T}}$ is the normalized and offset image point of $\boldsymbol{x}'$ [20] and $\boldsymbol{X}$ is the 3D point of the WFM without any motion. $\bar{x}'$ and $\bar{y}'$ is the mean of the x' and y' respectively. So the average distance from the origin is almost equal to $\sqrt{2}$. By using $\tilde{\boldsymbol{x}}'$ instead of $\boldsymbol{x}'$ in ***eq.4.1.1,*** it will become this form,

$$P_{11}X + P_{12}Y + P_{13}Z + P_{14} - \tilde{x}_i'\left(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}\right) = 0$$

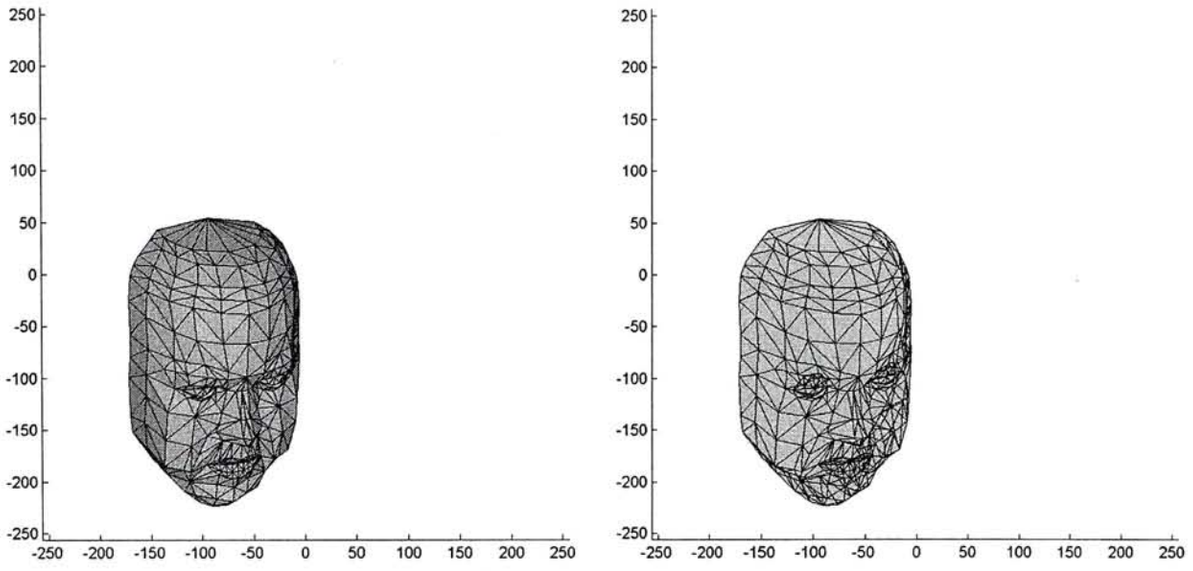
$$P_{21}X + P_{22}Y + P_{23}Z + P_{24} - \tilde{y}_i'\left(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34}\right) = 0$$

$$\tilde{\boldsymbol{B}}\ \boldsymbol{p} = 0 \qquad \textit{eq.4.1.3}$$

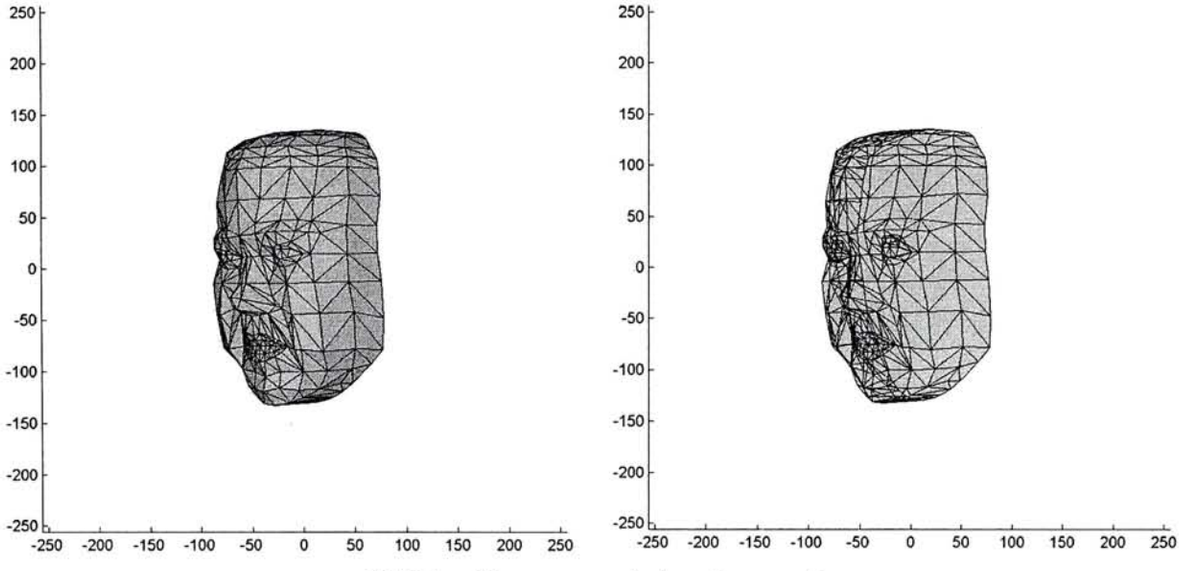
4.1.3. Experimental Results:

4.1.3a. Synthesized image by linear method without normalization

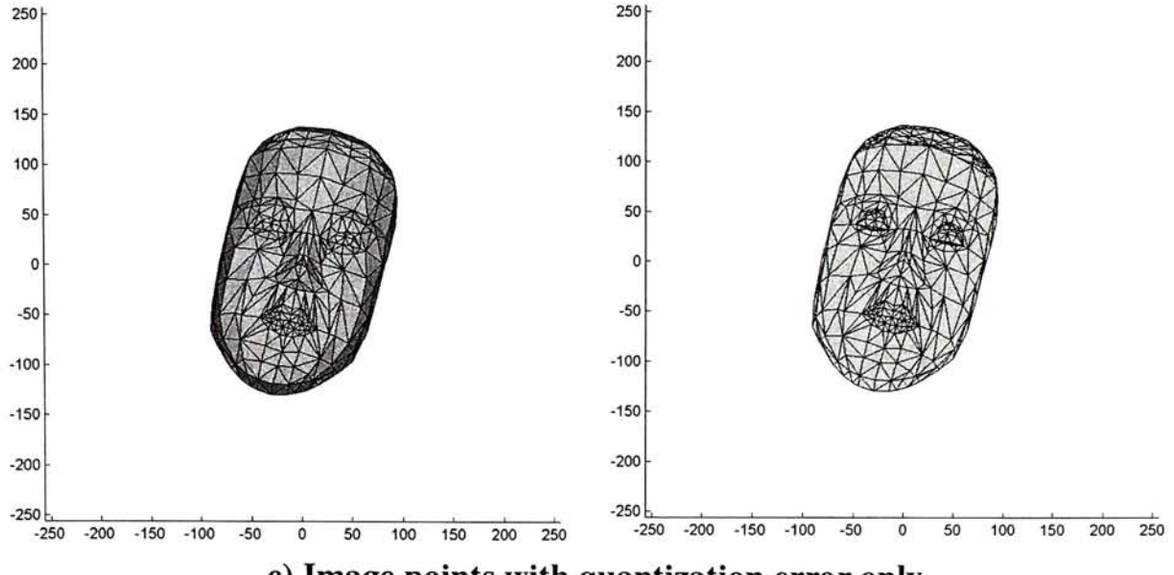
Figure 4.1.3 shows the performance of Linear Method without normalization, the left column is the original image and the right one is the synthesized image. Because the projection matrix is a mapping function, the depth of WFM will be lost after it is transformed by this matrix. As such, the 3D WFM will be changed to 2D planar surface.



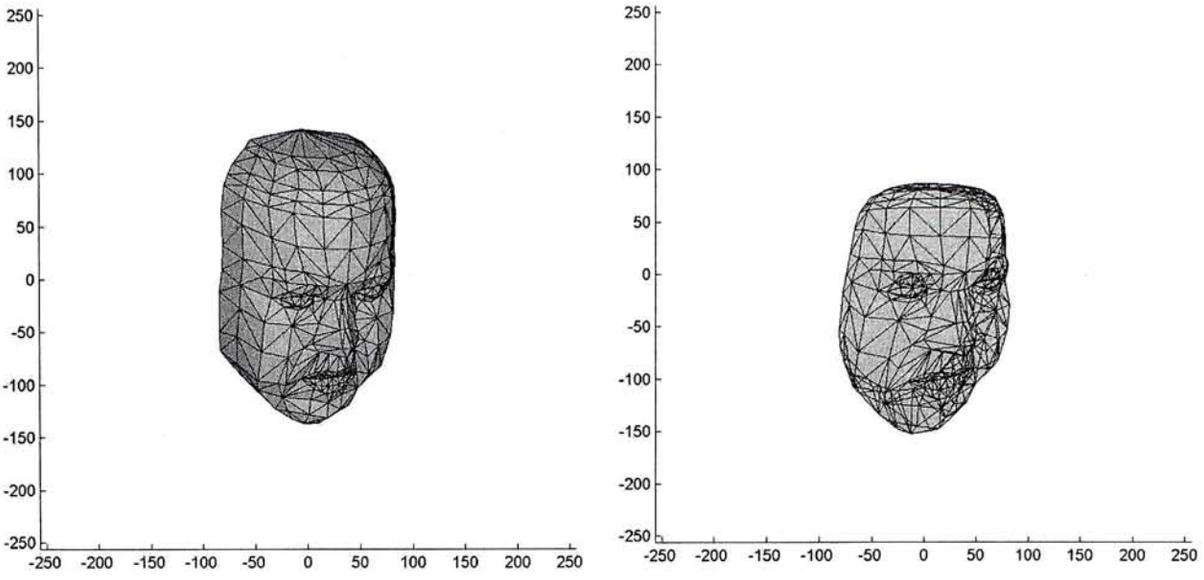
a) Ideal image points of pose 1



b) Ideal image points of pose 2



c) Image points with quantization error only



d) Image points with 1 pixel noise

Figure 4.1.3: The comparison between the original and synthesized images

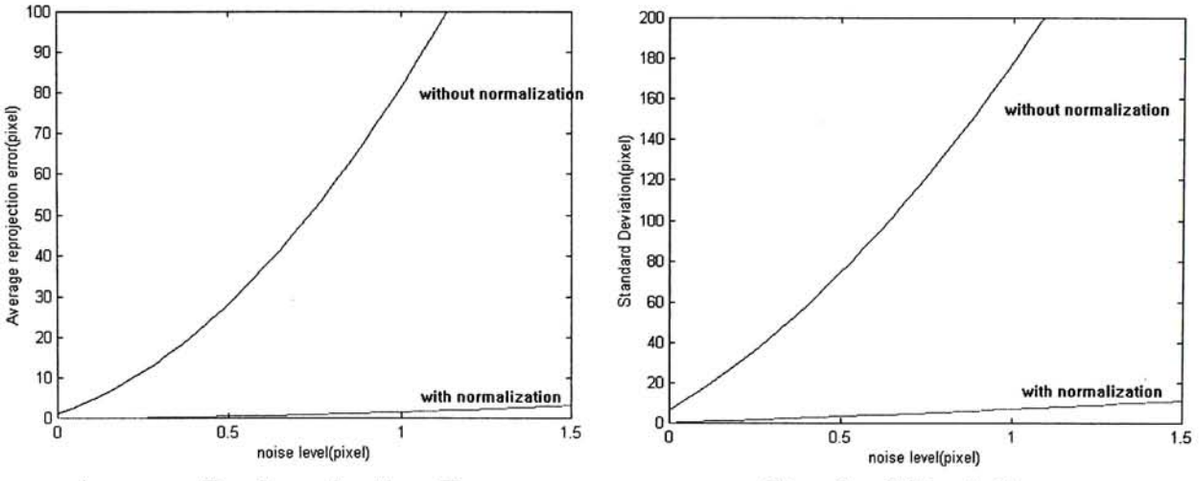
4.1.3b. Performance between linear method with and without normalization

Since all images are in pixel domain, the quantization noise will be generated in the digitization process. Therefore, the quantization error should be considered in the simulation and the image points should be digitized before processing. As such, we can know whether the method can give acceptable result under quantization noise and whether it is practical in real condition. In this simulation, the size of WFM is about 200×300. For the case without normalization, the average backprojection error of all feature points under quantization noise is 10.2625. This result is obtained after 1500 random orientations. It is a very poor result and not acceptable in real practice.

For the case with normalization, the average backprojection error of all feature points under quantization noise is 0.3630. These random orientations are the same as that of without normalization. It is also under 1500 orientations. According to

Figure 4.1.5, the output result is quite good and it can be used under noise level which is less than 2.

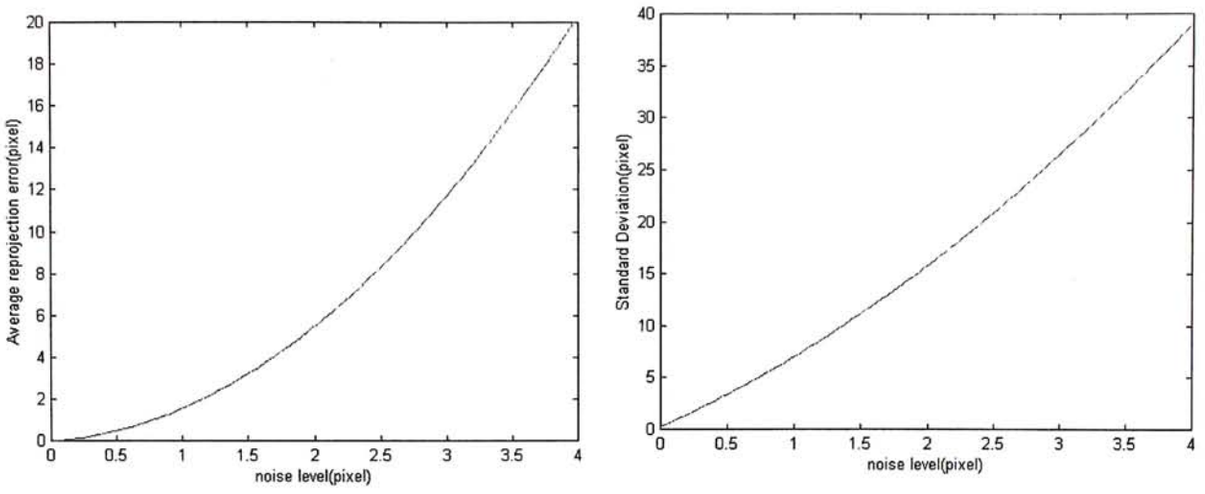
The results of Figure 4.1.4 and Figure 4.1.5 are obtained from the simulation for 2000 random orientations. Each orientation involves three iterations for inputting the same level of different gaussian noise to the control feature points. Therefore, the results are simulated from about 6000 times to check the performance of the linear method.



Average Backprojection Error **Standard Deviation**

Figure 4.1.4: The performance between normalized and without normalized image points

From the above Figure 4.1.4, the linear method without normalization is not very robust and the error cannot be accepted if the noise level is higher than 0.5. Therefore, the linear method should undergo normalization. The more detailed results of normalized one are shown in the following figures.



Average Backprojection Error **Standard Deviation**

Figure 4.1.5: The performance of linear method with normalization

4.1.3c. Performance of Linear method under quantization noise with different transformation components

If the transformation does not involve the z-component, e.g. no translation and no rotation about the z-direction, the performance of the linear method will be better. In the case without normalization, the quantization error of image points will have less effect if the motion does not involve z-component. The average backprojection error will be reduced from 10.2625 to 5.9741. Therefore, it is known that the performance of linear method is highly related to the motion.

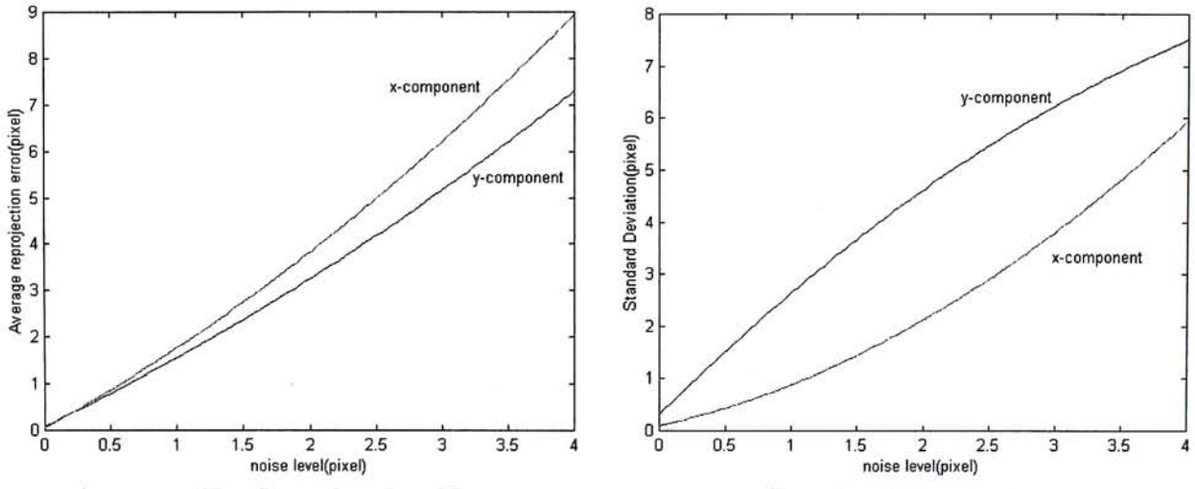
	Without normalization				**With normalization**			
Components	xy	xz	yz	xyz	xy	xz	yz	xyz
Aver. Backprojection error	5.974	14.056	15.375	10.263	0.359	0.371	0.404	0.363

Table 4.1.1: Results only considering the quantization noise

The results on Table 4.1.1 are obtained after 3000 orientation under quantization noise. If the z-component is also involved in the transform, the performance is worsened. Undoubtedly, we know that when more components involved, the analysis will be more difficult. However, when the transformation with x- and z-

components, or y- and z-components, the errors are higher than that of three components. Therefore, we know that the performance of linear method is greatly affected by the z-component during the transformation.

Figure 4.1.6 shows the performances of linear method if the transformation only involves x- or y-component. The result of z-component is not shown in the figure because the error is too large as compared to the others.

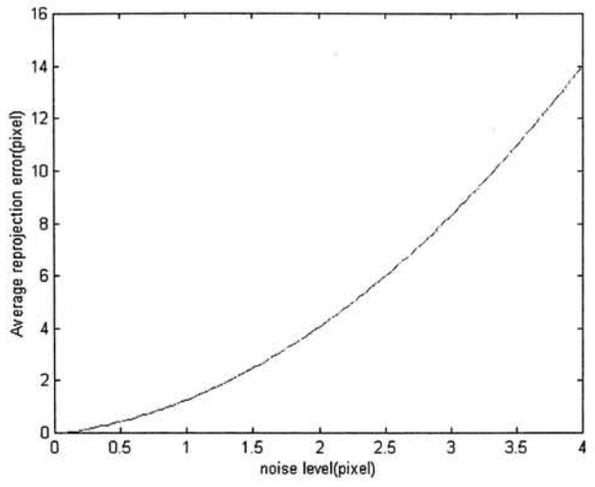


Average Backprojection Error **Standard Deviation**

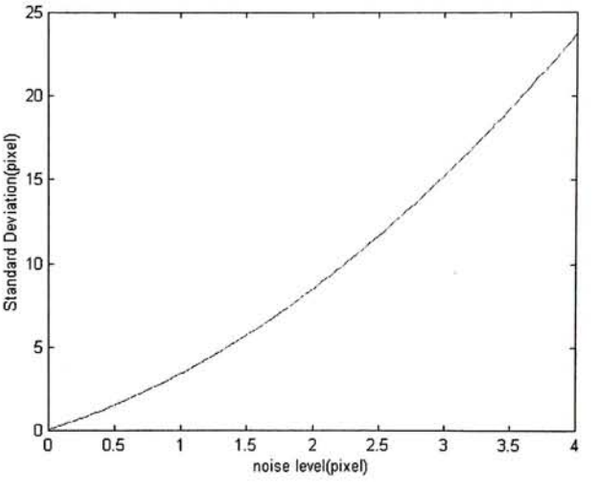
Figure 4.1.6: The performance of linear method with normalization

4.1.3d. Performance of normalized case with no transformation in z-component

The graph of performance versus noise level is shown in Figure 4.1.7 in which the transformations only involve x- and y-components. From the result, we know that the results are better than before. Therefore, large errors come from the z-component in the transform.



Average Backprojection Error



Standard Deviation

Figure 4.1.7: Performance of normalized linear method on the motion without z component

4.1.4. Summary:

For the case of the normalized image points, the overall performance will be better and the acceptable noise level will be higher than that of without normalization. It can be more robust and accurate. Since the coefficients of each variable are almost the same, the weighting will also be the same for sharing the noise. Otherwise, the error will be accumulated in the variables which have smaller coefficients.

By using this method to recover the motion, it can give a very good result when the noise level is lower than 1. The output transformation matrix can obtain almost the same synthesized image as the real one. However, if the noise increases and the motion involves the z-direction or axis, the performance of linear method will be affected and become worse. Especially for pure z-axis rotation, the obtained transformation matrix cannot be kept rigid and it will greatly destroy the structure of the WFM. So the synthesized image will be much different with the real one. Even though the image points are normalized, the result is still unacceptable if the noise level becomes larger.

Also, in this method, it is only necessary to input six image feature points with corresponding 3D feature points, which are known for WFM fitting. The required correspondences are reduced to six. However, a good 3D WFM model should be prepared in this approach. The obtained projection matrix is only a mapping transformation matrix which is not a rigid transformation. After 3D coordinates are multiplied by the projection matrix, the 3D WFM will be changed to 2D planar surface. So the depth information will be lost and the normal vectors of all nodes will be in the same direction.

However, the lighting effect is based on the angle between light source and the normal vectors of these surfaces. Therefore, if the lighting effect should be considered after the motion of WFM, the planar surface cannot estimate the normal vectors correctly and the lighting effect cannot be taken place to modify the texture color. So the appearance of fixed texture of WFM will not be natural. As such, if the lighting effect will be happened in the changing WFM, the projection matrix will not very suitable for giving the constraint to estimate the normal vector.

4.2. Two Stage Algorithm

For 3D model based video conferencing, estimation of the rigid motions [16,38] of the human head and the non-rigid motions of the face are usually required. In this chapter, we focus on the image motions corresponding to the rigid motions of the human head on video images. It is discovered that the image motions in image *i* can be computed through estimating an affine transformation matrix between the pose 1 and pose *i*. A novel Two Stage Algorithm [8] for estimating the transform is proposed. It assumes at least four 3D control points are estimated by stereo vision from the first pose, and the images of these 4 points are detected in image *i*. Then this transform can be estimated by minimizing the backprojection errors of the 4 image points. This is more robust and accurate than methods based on estimating the rigid motions. To verify our method, a 3D WFM model was used to model a human head for the simulated video image sequence. Our experiments show that the backprojection errors of our method can be as low as the noise level when we use 10 control points.

4.2.1. Introduction

In this chapter, we focus on the effects of rigid motion of the human head on the video images. A logical approach to tackle this problem is by estimating the 3D rigid rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ of the human head. The method of Essential Matrix Decomposition by Longuet-Higgins [25] for stereo vision can be used to estimate the $\boldsymbol{R}$ and $\boldsymbol{t}$. However, the method is too complicated and not robust in the presence of noise. The factorization of the Fundamental Matrix by Singular Value Decomposition is proposed by Richard I. Hartley [19]. It is simpler and more robust, but is still not robust enough for our purpose.

On the other hand, the image motions in image frame i due to 3D rigid motion can be computed through estimating a non-rigid transform from the first pose to pose i. This transform may not correspond to the original $\boldsymbol{R}$ and $\boldsymbol{t}$, but will give minimum backprojection errors if the head model is projected from the estimated pose i. This makes the problem easier. We show that the transform can be determined by a linear method using 6 corresponding points. However, this method is not practical too as it is sensitive to noise.

We proposed a novel Two Stage algorithm to estimate the transform. The first stage is an iterative method which minimizes the backprojection errors of 4 points under a constant distance assumption. The second stage is a non-linear method which minimizes the mean square backprojection errors without the above assumption. The result of the first stage is used as the initial value of the transform in the second stage. To verify our methods, a 3D WFM was used to model a human head for the simulated video image sequence. The actual number of control points used is 10 for least square fitting in the algorithm. Our experiments show that backprojection errors over all points using this algorithm can be as low as the noise level.

4.2.2. The Two Stage Algorithm

It is required to find the rigid transform between pose 1 and pose i which minimizes the backprojection errors in image i. We proposed a Two Stage algorithm to tackle this problem. The first stage is a linear iterative algorithm which minimizes the backprojection errors. We make a constant distance assumption to make the process linear. This algorithm is very robust to initial

estimation errors and noise. The second stage is a Newton-Raphson algorithm which minimizes the mean square backprojection error by using the results of the first stage as the initial values. The constant distance assumption is deleted to improve the accuracy of this non-linear process. Usually this stage can further reduce the error, but it needs a set of good initial estimates to work properly.

4.2.2a. Stage 1

Since the 3D information of the model is known, the iterative system can be set up for checking the error of the projection points. Therefore, the transformation matrix will be fine tuned to get a more accurate result. Iterative steps can be used to solve the problem with arbitrary initial values. The iterative algorithm has three main steps:

i) Image projection

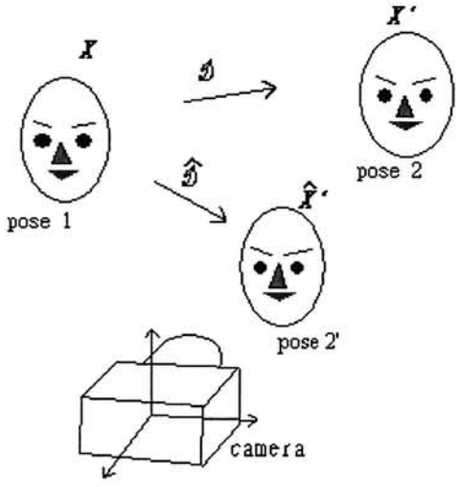


Figure 4.2.1: The general idea of stage 1 algorithm

As shown in Figure 4.2.1, our objective is get the transformation which can gives the same image. Under perspective projection, not only the object from exactly pose 2 can gives the same image, but the transformation can also outputs the

same image if the object located on that estimated pose, which has the same line of sight.

Let $\boldsymbol{X} = (X, Y, Z, 1)^T$ be a known 3D point and $\boldsymbol{x} = (x, y, f)^T$ be the 2D image point of $\boldsymbol{X}$ at pose 1, while $\boldsymbol{X'} = (X', Y', Z', 1)^T$ and $\boldsymbol{x'} = (x', y', f)^T$ be the corresponding 3D and 2D point at pose 2. The world coordinates system is located at camera position. Assuming that the camera has no skew, with 1:1 aspect ratio and a known focal length f from the camera calibration, we have the following,

$$\boldsymbol{X}' = \mathcal{D}\,\boldsymbol{X} = \begin{pmatrix} \boldsymbol{D}_{3\times4} \\ 0\ 0\ 0\ 1 \end{pmatrix}\boldsymbol{X}$$

$$\boldsymbol{x}' = f\,\boldsymbol{X}' / Z' \qquad \textbf{\textit{eq.4.2.1}}$$

where $\mathcal{D}$ is 4×4 transformation matrix.

Let $\hat{\boldsymbol{x}}' = (\hat{x}', \hat{y}', f)^T$ be the estimated 2D image point at pose 2', while $\hat{\boldsymbol{X}}' = (\hat{X}', \hat{Y}', \hat{Z}', 1)^T$ be the corresponding 3D point. While subscript k indicates the k^{th} iteration.

$$\hat{\boldsymbol{X}}' = \hat{\mathcal{D}}_k\,\boldsymbol{X} = \begin{pmatrix} \hat{\boldsymbol{D}}_k \\ 0\ 0\ 0\ 1 \end{pmatrix}\boldsymbol{X}$$

$$\hat{\boldsymbol{x}}' = f\,\boldsymbol{X}' / \hat{Z}' \qquad \textbf{\textit{eq.4.2.2}}$$

where $\hat{\mathcal{D}}_k$ is the corresponding 4×4 transformation matrix of this synthesized image.

ii) Error estimation

By ***eq.4.2.1*** and ***eq.4.2.2***, the error projection can be obtained,

$$\boldsymbol{\varepsilon} = \boldsymbol{x}' - \hat{\boldsymbol{x}}'$$

$$\boldsymbol{\varepsilon} = \boldsymbol{x}' - \frac{f}{\hat{Z}'}\hat{\boldsymbol{D}}_k \boldsymbol{X} \qquad \textit{eq.4.2.3}$$

$$\boldsymbol{\varepsilon} = \frac{f}{Z'}\boldsymbol{D}\boldsymbol{X} - \frac{f}{\hat{Z}'}\hat{\boldsymbol{D}}_k \boldsymbol{X}$$

if $Z' \approx \hat{Z}'$ and $-\boldsymbol{D}_\varepsilon = \boldsymbol{D} - \hat{\boldsymbol{D}}_k$

$$\boldsymbol{\varepsilon} = -\frac{f}{\hat{Z}'}\boldsymbol{D}_\varepsilon \boldsymbol{X} \qquad \textit{eq.4.2.4}$$

where $\boldsymbol{D}_\varepsilon = [\, D_{\varepsilon ij} \,]$ is a unknown 3×4 matrix.

Equating the errors given by ***eq.4.2.3*** and ***eq.4.2.4***,

$$\boldsymbol{x}' - \frac{f}{\hat{Z}'}(\, \hat{\boldsymbol{D}}_k - \boldsymbol{D}_\varepsilon \,)\boldsymbol{X} = 0 \qquad \textit{eq.4.2.5}$$

Decompose ***eq.4.2.5*** into three components,

$$X\, D_{\varepsilon 11} + Y\, D_{\varepsilon 12} + Z\, D_{\varepsilon 13} + D_{\varepsilon 14} + \left(\hat{Z}' x' / f - \hat{D}_{k11} X - \hat{D}_{k12} Y - \hat{D}_{k13} Z - \hat{D}_{k14}\right) = 0$$

$$X\, D_{\varepsilon 21} + Y\, D_{\varepsilon 22} + Z\, D_{\varepsilon 23} + D_{\varepsilon 24} + \left(\hat{Z}' y' / f - \hat{D}_{k21} X - \hat{D}_{k22} Y - \hat{D}_{k23} Z - \hat{D}_{k24}\right) = 0$$

$$X\, D_{\varepsilon 31} + Y\, D_{\varepsilon 32} + Z\, D_{\varepsilon 33} + D_{\varepsilon 34} + \left(\hat{Z}' - \hat{D}_{k31} X - \hat{D}_{k32} Y - \hat{D}_{k33} Z - \hat{D}_{k34}\right) = 0$$

If 4 corresponding feature points are detected, the above linear system can be solved to obtain the values of

$$D_{\varepsilon 11}, D_{\varepsilon 12}, D_{\varepsilon 13}, D_{\varepsilon 14}, D_{\varepsilon 21}, D_{\varepsilon 22}, D_{\varepsilon 23}, D_{\varepsilon 24}, D_{\varepsilon 31}, D_{\varepsilon 32}, D_{\varepsilon 33}, D_{\varepsilon 34}$$

iii) Feedback value

$$\hat{\boldsymbol{D}}_{k+1} = \hat{\boldsymbol{D}}_k - \boldsymbol{D}_\varepsilon \qquad \textit{eq.4.2.6}$$

By ***eq.4.2.6***, the error of $\hat{\boldsymbol{D}}_k$ is $\boldsymbol{D}_\varepsilon$. The estimated $\hat{\boldsymbol{D}}_{k+1}$ was used for the

k+1th iteration from the steps (i) to (iii). Because the transformation matrix is only up to a scale, one of them can be selected by fixing the depth. However, the constructed matrix will lose the property of rigid 3D motion. It is only an affine transformation matrix. The general form is as follows:

$$\begin{pmatrix} \hat{D}_{11} & \hat{D}_{12} & \hat{D}_{13} & \hat{D}_{14} \\ \hat{D}_{21} & \hat{D}_{22} & \hat{D}_{23} & \hat{D}_{24} \\ \hat{D}_{31} & \hat{D}_{32} & \hat{D}_{33} & N \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where N is the arbitrary value about the changed depth of 3D model.

4.2.2b. Stage 2

To minimize the backprojection error of feature points by ***eq.4.2.3***, the objective function is obtained,

$$\min_{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3} \xi = \sum_{i=1}^{n} \left[\left(\frac{\boldsymbol{d}_1^{\mathrm{T}} \boldsymbol{X}_i}{\boldsymbol{d}_3^{\mathrm{T}} \boldsymbol{X}_i} - x_i' \right)^2 + \left(\frac{\boldsymbol{d}_2^{\mathrm{T}} \boldsymbol{X}_i}{\boldsymbol{d}_3^{\mathrm{T}} \boldsymbol{X}_i} - y_i' \right)^2 \right] \qquad \textit{eq.4.2.7}$$

where $\boldsymbol{D} = \begin{pmatrix} \boldsymbol{d}_1^{\mathrm{T}} \\ \boldsymbol{d}_2^{\mathrm{T}} \\ \boldsymbol{d}_3^{\mathrm{T}} \end{pmatrix}$,

n is number of feature points.

By using Newton-Raphson search method [29] to minimize the ***eq.4.2.7*** and using the obtained solution from stage 1 to be the initial value of this optimization process. Therefore, the solution will be further optimized. The obtained transformation matrix will be more accurate.

4.2.3. Summary of the Two Stage Algorithm

Initial conditions:

Camera is assumed calibrated. Ten 3D control points are estimated by stereo vision and their images are detected in image i where motions of image points have to be estimated. Assume $\hat{X}' = X$ for the 1st iteration.

Stage 1

Assume a constant distance $Z' \approx \hat{Z}'$ during the iteration.

1. The synthesized image points are generated by ***eq.4.2.2***.
2. The transformation error is obtained by ***eq.4.2.5***.
3. The feedback value is computed by ***eq.4.2.6***.
4. Repeat step(1) to (3) until the $\varepsilon <$ threshold.

Stage 2

5. The result of stage 1 is used to be the initial value for the non-linear optimization by ***eq.4.2.7***. The constant distance assumption is deleted.

4.2.4. Experimental results

In the simulation using a 3D WFM of a human face, all the image feature points are added with a gaussian noise with a standard deviation of δ pixels. The image projection of the 3D model is about 150x260 pixels, and the image size is 512x512 pixels.

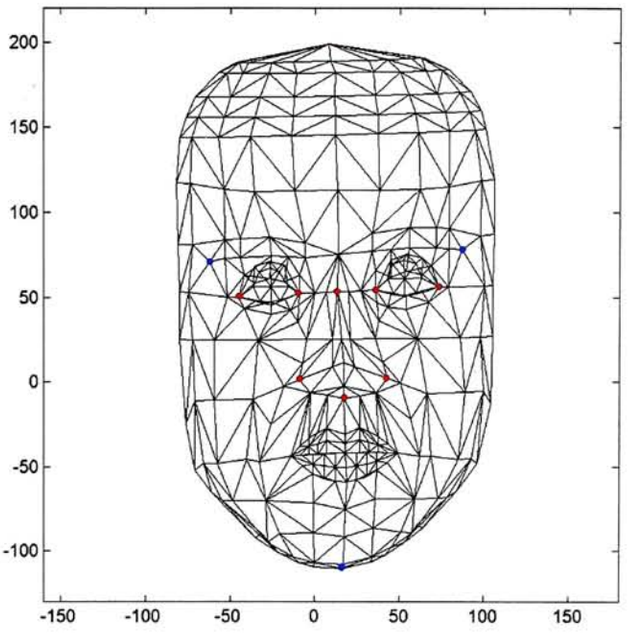


Figure 4.2.2: 3D WFM and the selected rigid control points

There are total 469 feature points and the diagram of the WFM is shown in Figure 4.2.2. Only ten feature points are selected to be control points, which are red and blue in color. In the simulated experiment, ten out of eleven control points will be selected to be control points. However, in the real experiment, only the red color points are the control feature points.

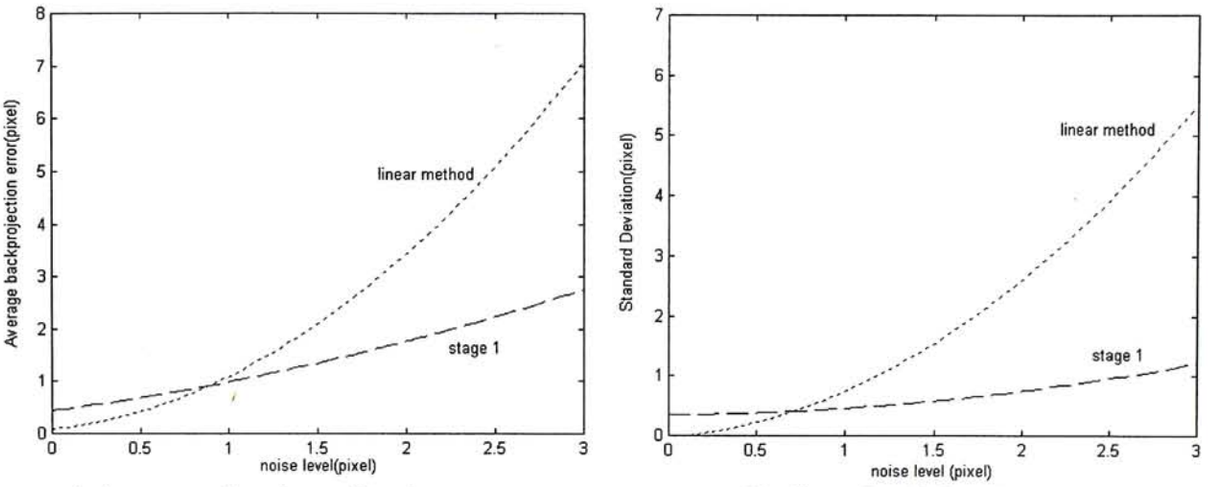
Figure 4.2.3 shows that the comparison between the linear and stage 1 methods at different noise levels. The result is obtained by repeating the simulation for 10000 times with random rotation and translation within ± 30° and ± 80 pixels. The time for estimating the motions of all the feature points between two images is about 0.088 sec on a PC (PIII450). The average backprojection error is defined as follows,

$$\Delta = \frac{1}{n}\sum_{i=1}^{n} \left| (\bar{\boldsymbol{x}}_i - \hat{\boldsymbol{x}}_i) \right|$$

where n is number of feature points,

$\bar{\boldsymbol{x}}_i$ is no noise feature points,

$\hat{x}_i$ is estimated feature points.



a) Average backprojection error **b) Standard Deviation**

Figure 4.2.3: Comparison between two methods, linear and stage 1 method

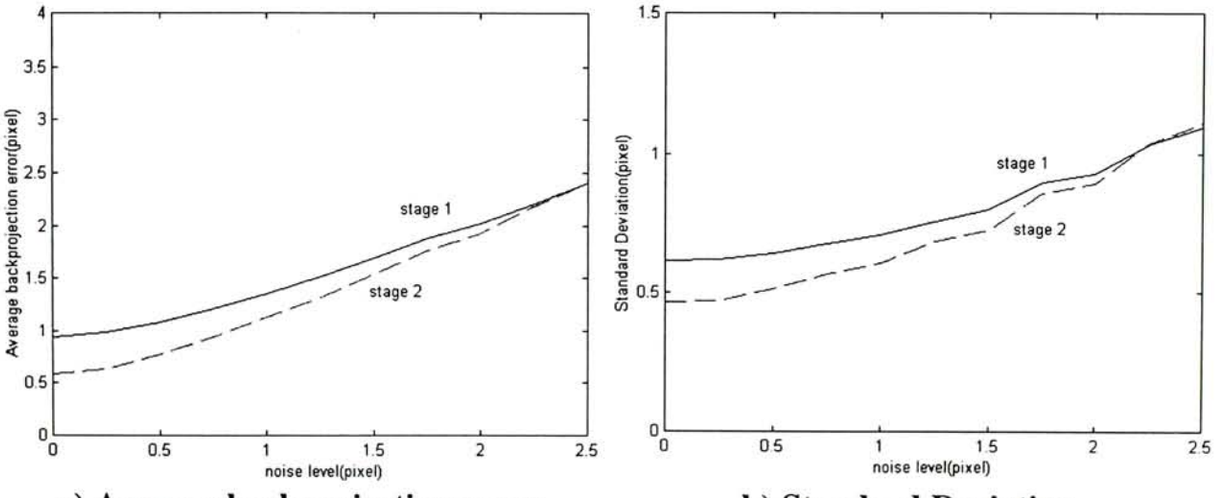
Based on the result, we know that the average error of backprojection in stage 1 method is very close to the noise level. Although it cannot get the same correct result as the linear method in the ideal image points, it can use the minimization of error to correct the estimated orientation in higher noise level. After the noise level is higher than 0.75, their performances are almost the same and the stage 1 is better beyond this noise level. Therefore, unless the image feature points are ideal, stage 1 method will be better than the linear method. Also, for having further optimization, to obtain stable or robust results will be one major reason for choosing initial value. Therefore, we will use the result of stage 1 method to be the initial value of the stage 2 method, which is a non-linear optimization.

To see whether the stage 2 method can further optimize the estimated pose, the quantization noises are added to image points. The average backprojection error of stage 1 is 1.2117 pixel and the standard deviation is 0.7869 pixel. After stage 2, the error is 0.7361 and the standard deviation is 0.5628 pixel. Therefore, the errors have been minimized and the stage 2 can be used for further optimization.

	Stage 1			*Stage 1 + Stage2*		
Involved components	***R***	***t***	***R, t***	***R***	***T***	***R, t***
Aver. Backprojection error	1.1969	0.2870	1.2117	0.7369	0.3028	0.7869
Standard deviation	0.7694	0.1523	0.7869	0.5170	0.1766	0.5628

Table 4.2.1: Comparison of performances between two stages under quantization noise

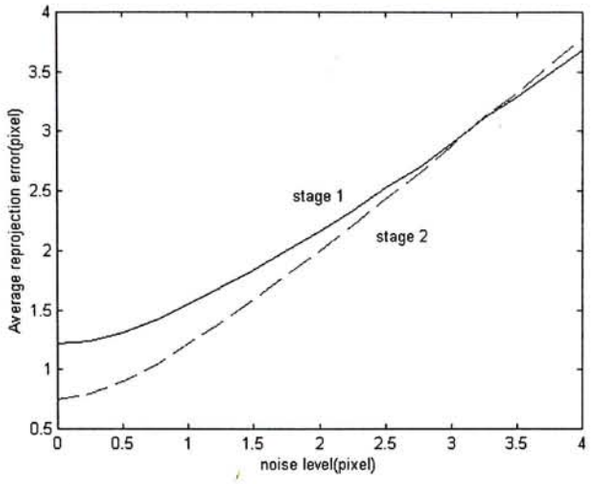
Figure 4.2.4 shows that the comparison between stage 1 and stage 2 methods at different noise levels after 3000 times. The rotation angle and displacement are varied between ± 40° and ± 150 pixels respectively. From the figure, we know that the performance of stage 2 method is better than that of stage 1. Therefore, the results have been further optimized and the error is very close to the noise level.

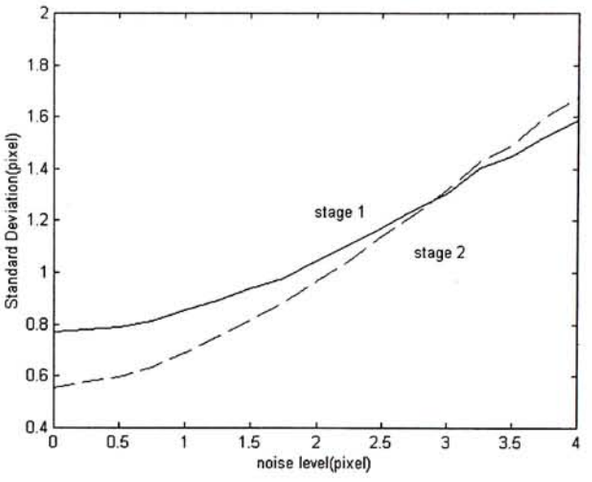


a) Average backprojection error **b) Standard Deviation**

Figure 4.2.4: Comparison between two stages

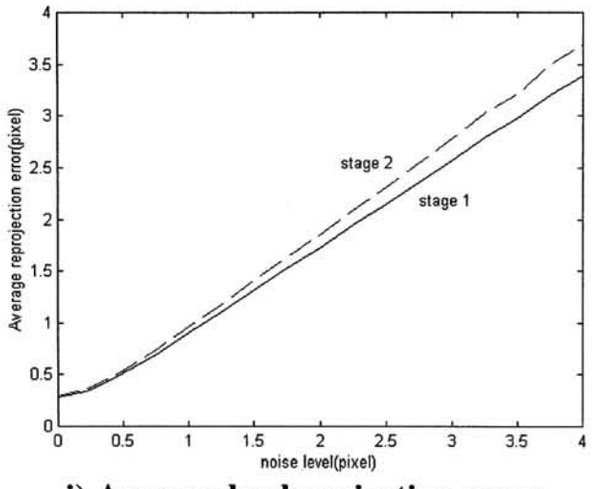
Figure 4.2.5a shows the results of two stages under both quantization and gaussian noise in different noise level. While Figure 4.2.5b and Figure 4.2.5c show the performance when no rotation and translation respectively.

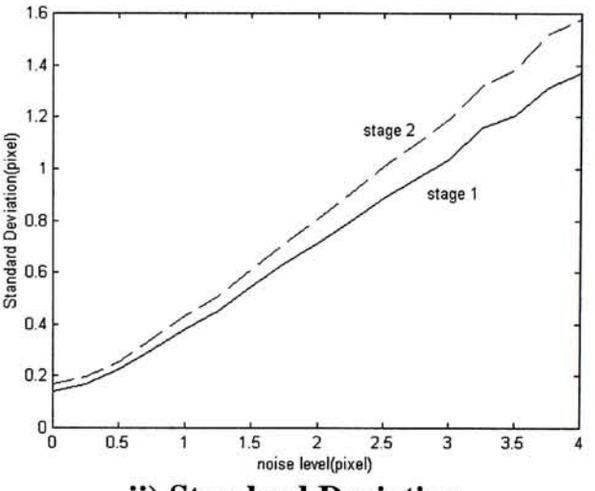




i) Average backprojection error **ii) Standard Deviation**

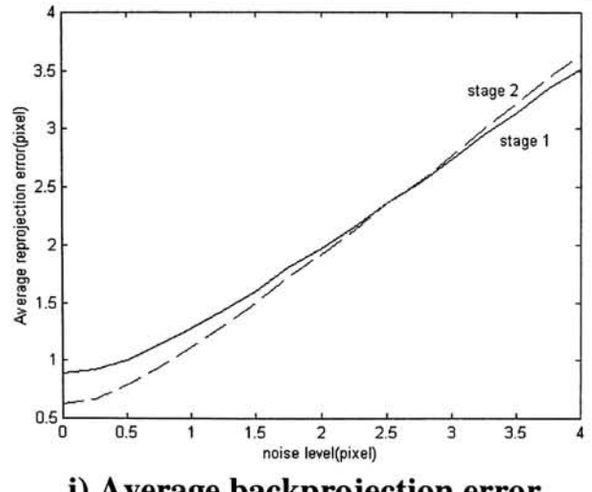
a) Performance under quantization and gaussian noise

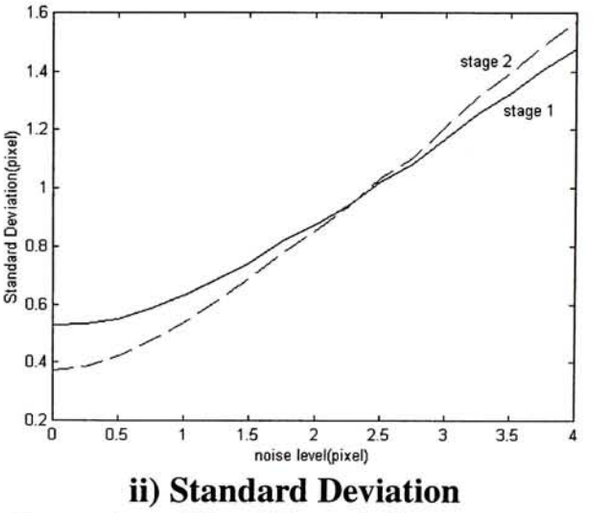




i) Average backprojection error **ii) Standard Deviation**

b) Under quantization and gaussian noise without rotation



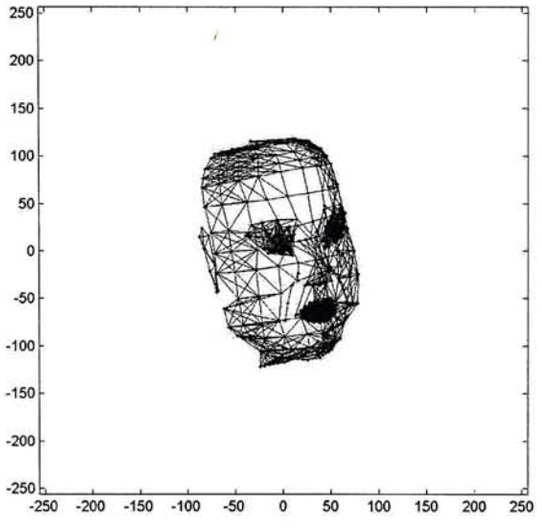


i) Average backprojection error **ii) Standard Deviation**

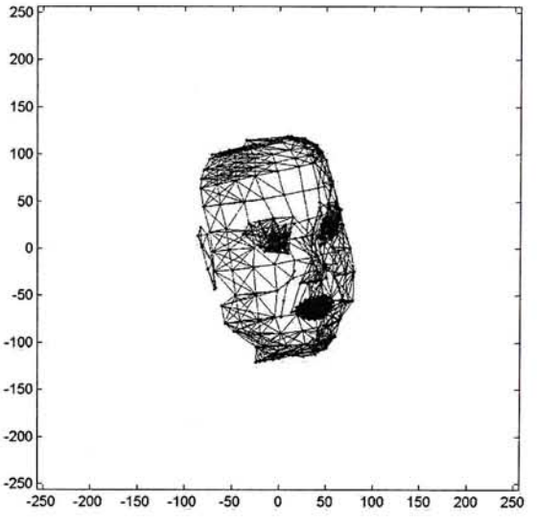
c) Under quantization and gaussian noise without translation

Figure 4.2.5: The detail comparison between two stages

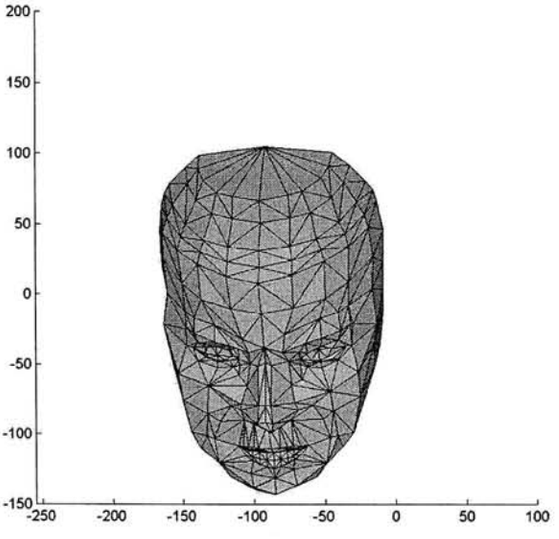
The Two Stage algorithm will be used to minimize the backprojection error of the feature points. The original and reconstructed head models are shown in Figure 4.2.6. The original and reconstructed images are practically the same visually even with a noise of 1 pixel.



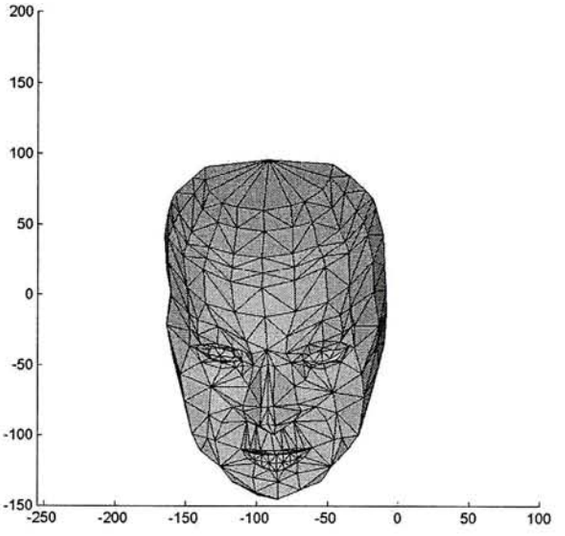
a) Original image of pose 2



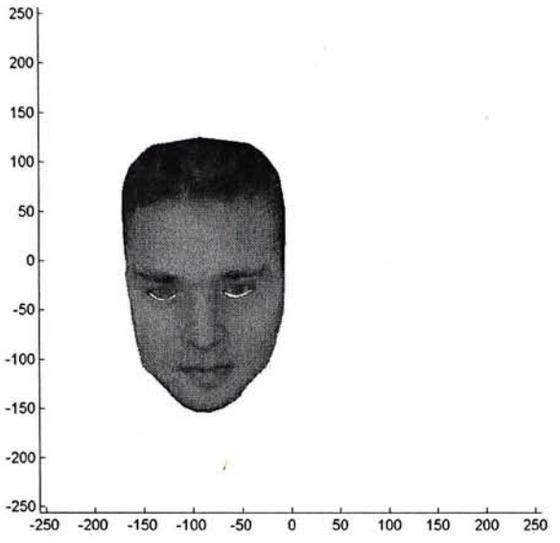
b) Synthesized image of pose 2



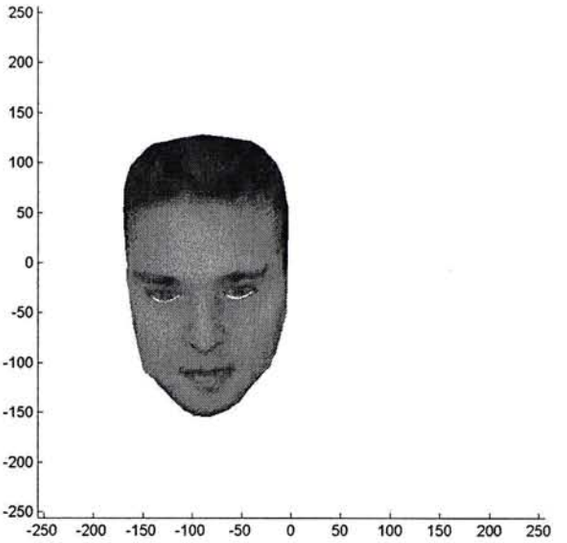
c) Original image of pose 3



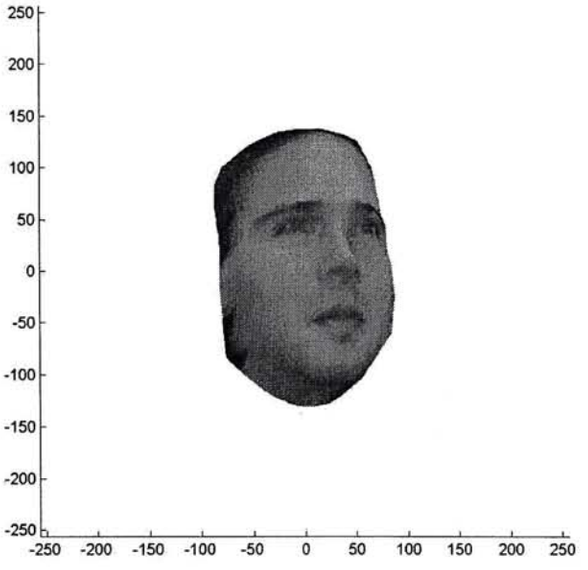
d) Synthesized image of pose 3



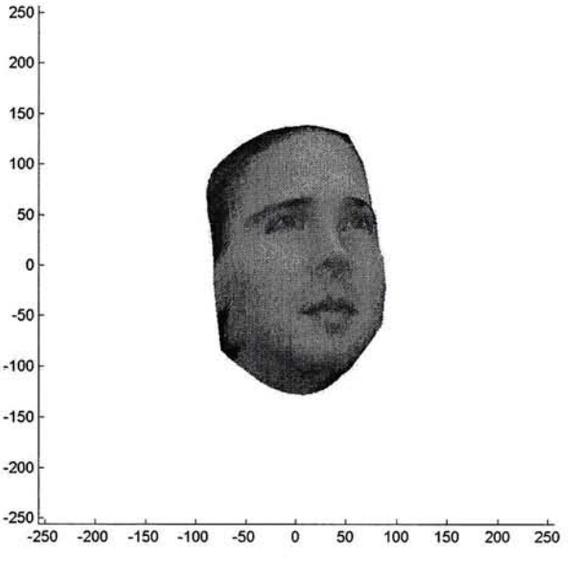
e) Original image of pose 4



f) Synthesized image of pose 4



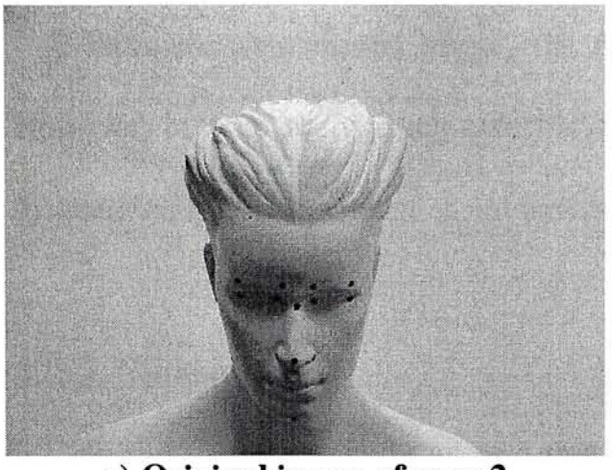
g) Original image of pose 5



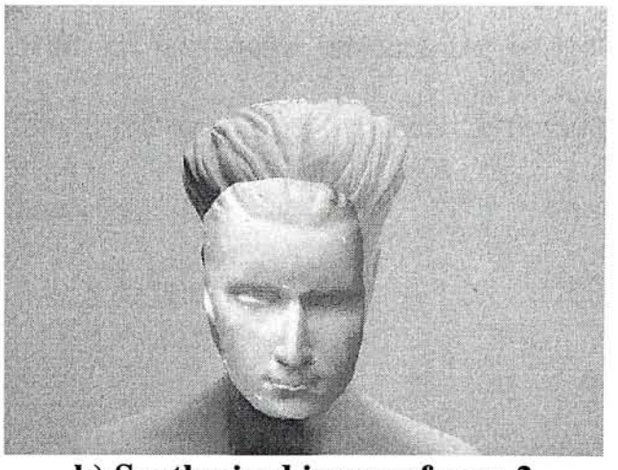
h) Synthesized image of pose 5

Figure 4.2.6: The results of simulated image

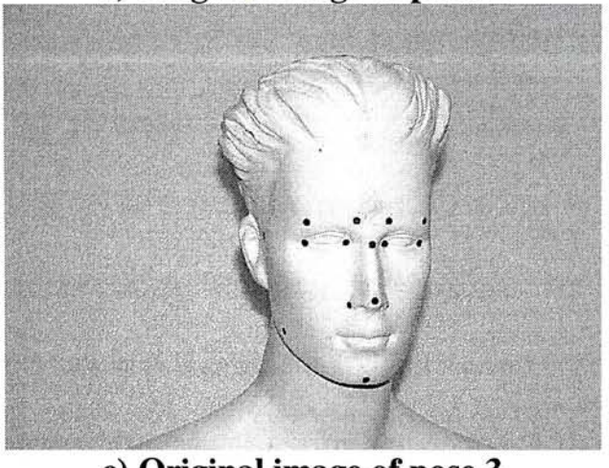
By using real image, the image should undergo the quantization noise. Therefore, the feature points will be non-ideal. The re-synthesized images are shown in Figure 2.4.7, where the left columns are the original real images and the right columns are the corresponding synthesized images by using Two Stage Algorithm. The synthesized images are generated by projecting the texture mapped WFM into the original images. The target of synthesized image considers the human face only, and so background is not the synthesized result.



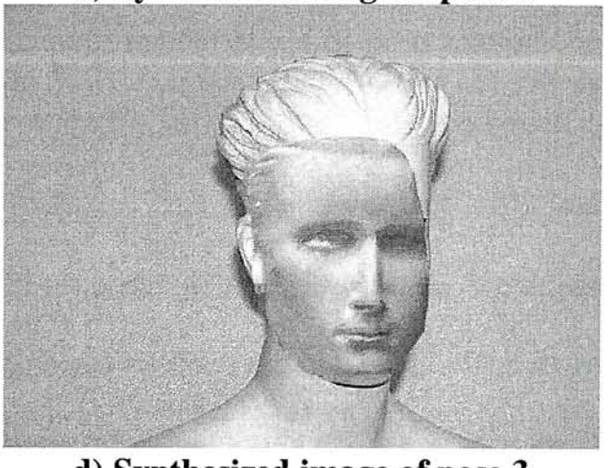
a) Original image of pose 2



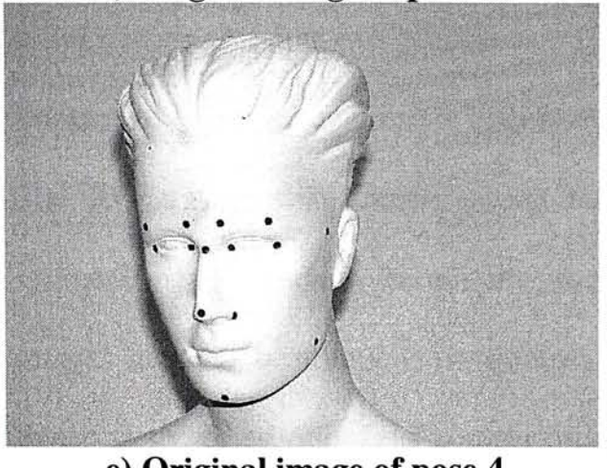
b) Synthesized image of pose 2



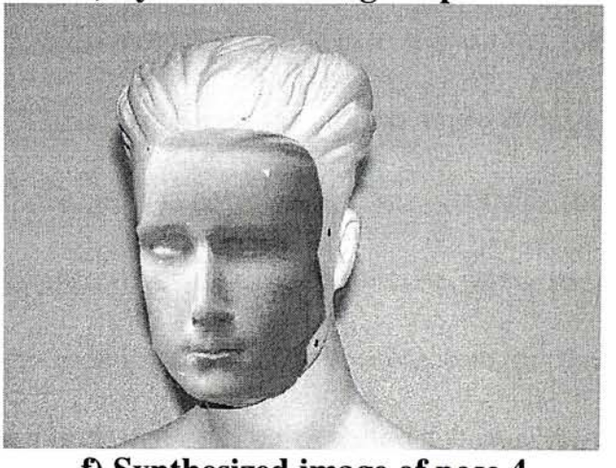
c) Original image of pose 3



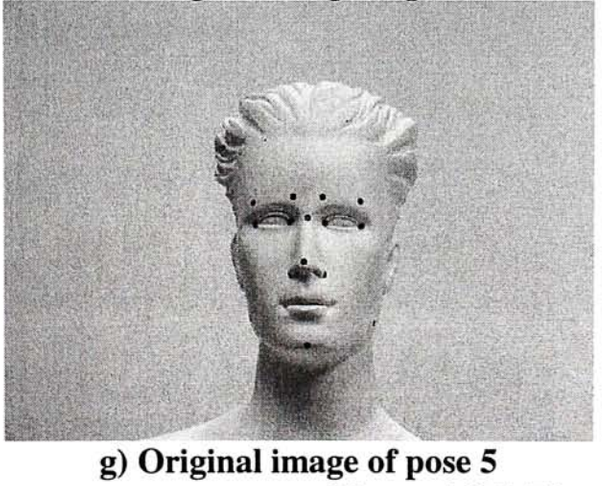
d) Synthesized image of pose 3



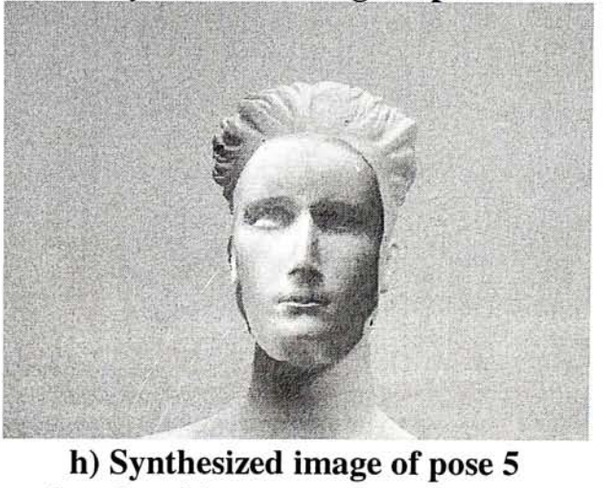
e) Original image of pose 4



f) Synthesized image of pose 4



g) Original image of pose 5



h) Synthesized image of pose 5

Figure 4.2.7: The results of real image

From the synthesized image, the position of face quite matches the image. However, because the lighting effect is different, the gray level between the texture mapped WFM and the image is not matched exactly.

4.2.5. Summary

In this chapter, it is discovered that we do not need to estimate rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ of the rigid body in order to account for the effect of rigid motions on the 2D images. Instead, we only need to compute an affine transformation which is not constrained to have an orthogonal sub-matrix. This is much more efficient and robust than estimating the $\boldsymbol{R}$ and $\boldsymbol{t}$. We had proposed a novel Two Stage Algorithm to tackle this problem and very good experimental results were obtained. The stage 2 process often can further optimize the result of stage 1 process, so a very good result can be get after two processes. It is in general more robust to noise than the linear method when the noise is larger than 1 pixel or when the rotation angle is large. The Essential matrix decomposition methods are generally not very robust for synthesizing the backprojection image. Eight corresponding points between two images have to be found. Six control image points have to be detected for the linear method and only four control image points have to be detected for our Two Stage Algorithm.

Chapter 5

Facial Motion Estimation and Synthesis

Although the algorithm on pose estimation is introduced in previous chapter, it can only recover the rigid motion. The synthesized image is still different from the real one because the target person will have different expressions or facial motions on his face. Therefore, non-rigid motions should also be considered in the motion estimation process.

When the images of a human face are synthesized with different expressions at the receiving end, the output images can be produced by deforming the 3D *FM* with the received facial motion parameters. There are a number of ways to synthesize the facial movements on the 3D *FM*. The two common ones are the "Clip and Paste"[1,26] and facial structure deformation methods. However, the compression ratio of "Clip and Paste" is not as good as that of the latter because the former still needs to transmit partial image. The latter has to estimate the motion parameters of facial features and use these parameters to deform the 3D FM at the receiving end for image synthesis.

One of the popular systems is Facial Action Coding System (*FACS*) [13], which was originally used in psychological studies and is a main approach on facial animation. *FACS* defines a set of minimal basic actions called Action Units (*AU*)

performable on a human face such as inner brow raise, outer brow raise, etc. Each facial action can be decomposed as a combination of Action Units. There are 46 defined Action Units in the *FACS*. The deformation of the WFM is done through the movement of nodes according to the deformation rules defined by the *AU*s. However, the recognition of *AU*s [12,14,15], which are appeared on the incoming face, should be processed before synthesis of facial expression. In such way, it is difficult to be performed in real time.

Our proposed new approach is a Muscle-based method which is very similar to the *FACS*. The difference is that it considers the direction of each muscle only. The movement of nodes is limited by the directions of their respective connected muscles. If all muscles are defined and the non-rigid points were back-transformed to the frontal view for comparison, there is no need to recognize the facial action before the synthesis of facial image. Because all the connected muscles of that feature points are known, the motion of points will be constrained by the characteristic of *Motion Units* (*MU*s). So the movement of muscles can be estimated by the 2D motion of facial image points.

5.1. Facial Expression based on face muscles

The function of the decoder in the 3D model-based coding system is to synthesize the output images by using the 3D WFM of the face and the analysis parameters. In synthesis of facial expression, every node of WFM should be completely defined in each motion. If the nodes are defined based on the face muscles, the motion will be more natural. Since each face muscle controls a particular region and moves in a particular direction, the nodes of WFM can be modeled by these face muscles. The expression can be done automatically and naturally.

5.1.1. Overview of Facial Action Coding System

Facial expression is an important mean of communication between people. This naturally leads to a significant amount of research on facial expressions in computer vision and computer graphics.

In the literature, it is common that the visually distinguishable facial movements are described by using the Facial Action Coding System (*FACS*) [1,13,28], which was proposed by Ekman and Friesen [13] for psychological studies. *FACS* describes a set of minimal basic actions called *Action Units* (*AU*) performable on a human face such as inner brow raise, outer brow raise and so forth. There are 46 defined Action Units in the model-based coding system. The deformation of the WFM is done through the movement of vertices according to the deformation rules defined by the *AU*s. The deformation rules are in turn defined by using physical and anatomical knowledge.

FACS is an animated method using combinations of these 46 action units to generate a large set of possible facial expressions. For example, happiness expression is considered to be a combination of "pulling lip corners (*AU*12+13) and/or mouth opening (*AU*25+27) with upper lip raiser (*AU*10) and a bit of furrow deepening (*AU*11)." However, this is just one type of smile. There are many variations of the above motions, each having a different intensity of actuation.

A trained human *FACS* coder decomposes an observed expression into the specific *AU*s that produce the expression. *FACS* is coded from the video and the code provides precise specification of the dynamics (duration, onset, and offset time) of facial movement in addition to the morphology (the specific facial actions which occur).

Based on the different distributions and different kinds of face muscles, the affected region (nodes) and the motion direction for each node can be defined. Therefore, the definition of *AU* and the combination of *AU* should be adjusted well enough in order to get a good facial expression. The following table gives the function of each *AU*,

AU	***FACS* Name**	***AU***	***FACS* Name**
1	Inner Brow Raiser	24	Lip Pressor
2	Outer Brow Raiser	25	Parting of Lips
4	Brow Raiser	26	Jaw Drop
5	Upper-Lid Raiser	27	Mouth Sketch
6	Cheek Raiser	28	Lip Suck
7	Lids Tightener	29	Jaw Thrust
8	Lips Together	30	Jaw Sideways
9	Nose Wrinkler	31	Jaw Clencher
10	Upper-Lip Raiser	32	Lip Bite
11	Nasolabial Furrow Deeper	33	Cheek Blow
12	Lip Corner Puller	34	Cheek Puff
13	Cheek Puffer	35	Cheek Suck

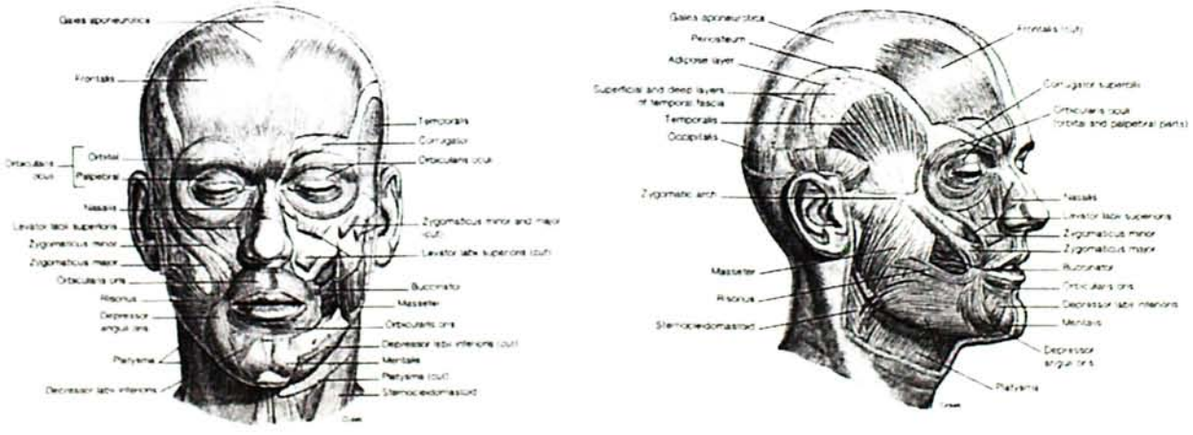
14	Dimpler	36	Tongue Bulge
15	Lip Corner Depressor	37	Lip Wipe
16	Lower-Lip Depressor	38	Nostril Dilator
17	Chin Raiser	39	Nostril Compressor
18	Lip Puckerer	41	Lip Droop
19	Tongue Out	42	Eyelid Slit
20	Lip Stretcher	43	Eyes Closed
21	Neck Tightener	44	Squint
22	Lip Funneler	45	Blink
23	Lip Tightener	46	Wink

Table 5.1.1: Lists of Single Facial Action Units

5.1.2. Distribution of Motion Unit

In this thesis, we propose to use the concept of motion unit (MU) to tackle the problem of expression animation.

According to an anatomy book [18], the structure of face muscles is shown in Figure 5.1.1. A set of *MU*s is defined, each involves the movements of several muscles. It should be noted that *MU* is Muscle-Based while *AU* is Action-Based, e.g. open left eye, raise right brow and etc.



a) Front view **b) Side view**

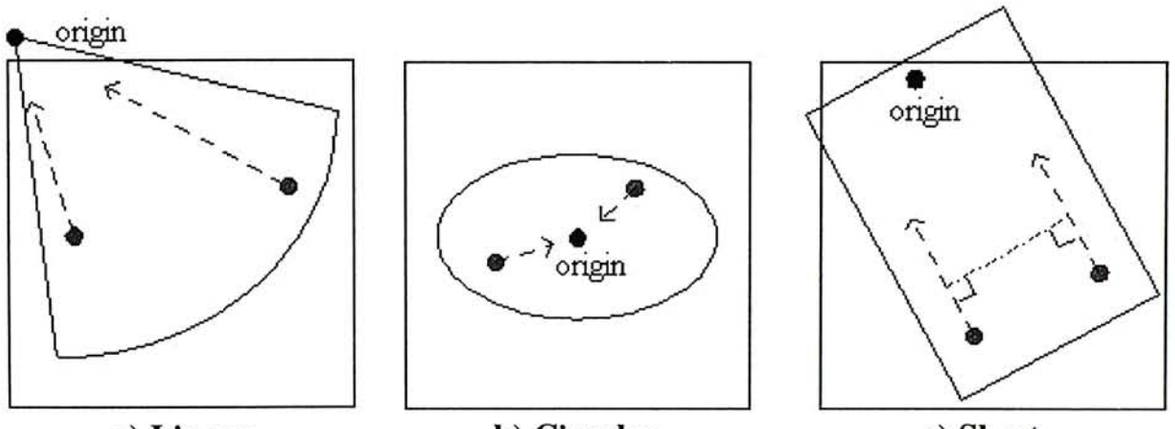
Figure 5.1.1: The Anatomy of Face

The muscles of the facial expression can be grouped according to the orientation of the individual muscle fibers. Based on the contraction of the muscles, three types of muscles can be discerned as the primary motion muscles:

i) Linear muscle, which pulls in an angular direction, is contracted towards the static node (*origin*) of attachment on the bone.

ii) Circular or sphincter muscle, which squeezes and contracts around an *imaginary central point*, like the tightening of a string bag. This squeezing can be described as occurring uniformly about a point of contraction.

iii) Sheet muscle, which behaves as a series of linear muscles spreading over an area, is a broad flat area of muscle fiber strands and does not emanate from a point source.



a) Linear **b) Circular** **c) Sheet**

Figure 5.1.2: Three Types of Muscles

There are a total of twenty-two *MU*s in our algorithm. Each *MU* is defined as having a given *motion direction* and an *origin* [28] or *end-point*, which will not move even if the muscle contracts. There are eighteen sheet (unidirectional) *MU*s and four circular *MU*s.

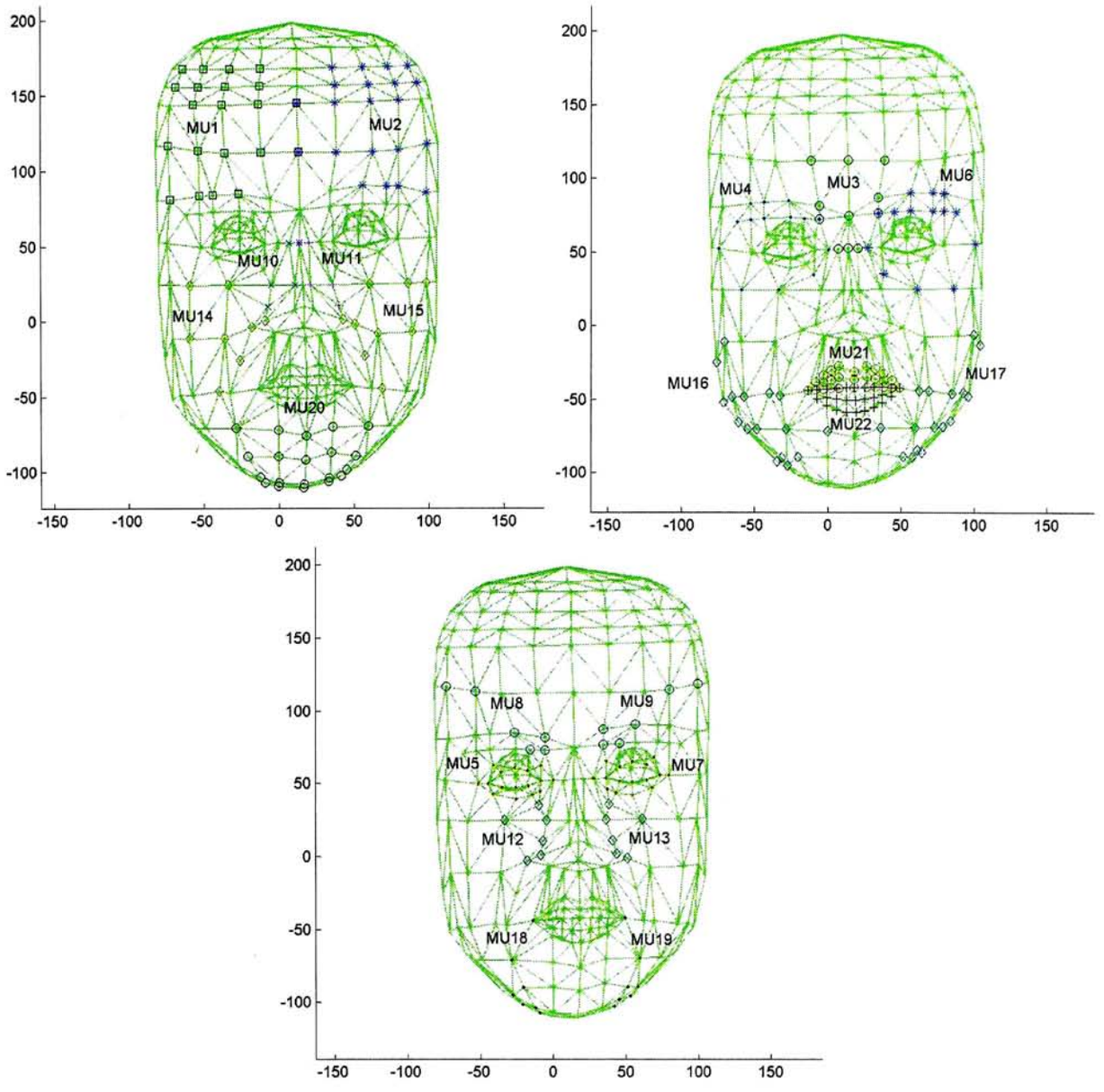


Figure 5.1.3: Distribution of motion unit in face

Figure 5.1.3 shows that the distribution of *MU*s over the 3D WFM. There are totally 22 *MU*s and their corresponding directions and origins are shown in the following figure. The red line is the direction of that particular *MU* and the blue circle is the origin.

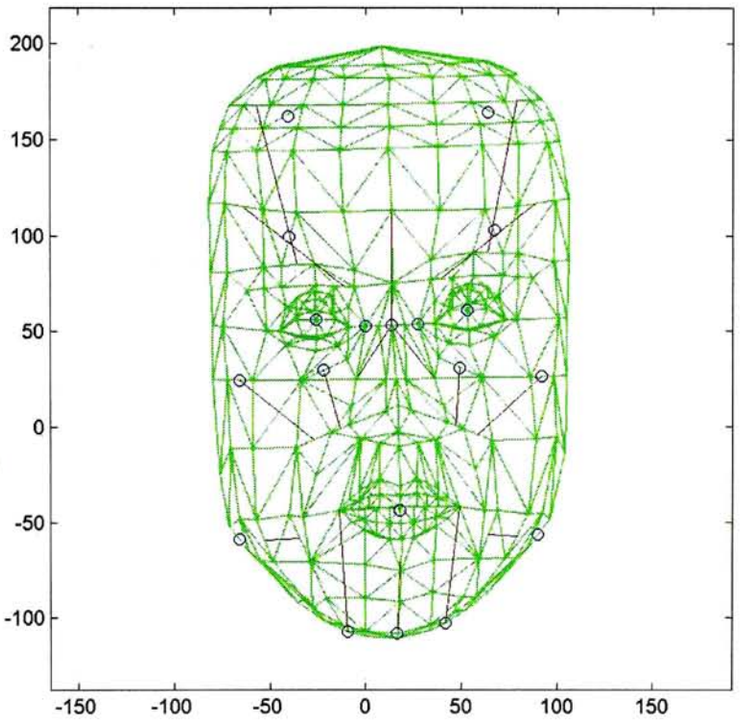


Figure 5.1.4: Distribution of origin and direction of each *MU*

Based on the characteristics of muscle, those further away points will have more power to move in that effective region. Figure 5.1.5 is an example of contraction of muscle, the upper one is not contracted and the lower one is contracted. A grid represents a cell and the end-point is in cell *a* (the left most cell). If the muscle is consisted of ten cells and every cell is contracted for 1 unit, cell *b* will move 10 units. So, the displacement will be accumulated in the most further away one.

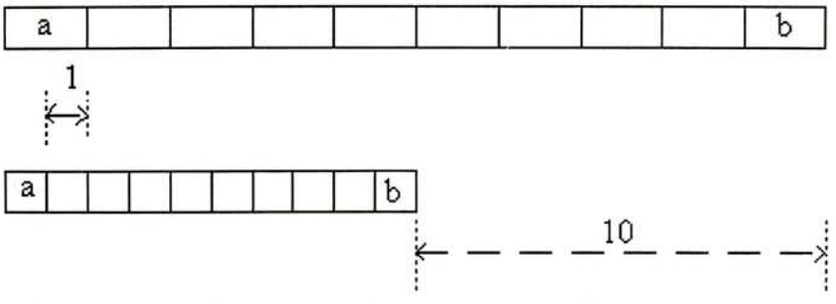


Figure 5.1.5: Example of contraction of muscle

5.1.3. Algorithm

In my animator, most of the muscles are defined as sheet muscles and the others are circular muscles. There is no linear muscle in that animator because linear muscle is only a sector shape of circular muscle. Therefore, only two types of muscles have been used in the animation,

1) Sheet (unidirectional) *MU*

2) Circular *MU*

a) For Unidirectional MU

A unidirectional *MU* is a series of almost parallel fibers which spread over an area. All the nodes on this *MU* can only move in one direction. The updating equation is,

$$\boldsymbol{p}_{ij}' = \boldsymbol{p}_{ij} + s_i \cdot 2^{\frac{\Delta \boldsymbol{D}_{ij}}{\tau}} \cdot \boldsymbol{v}_i \qquad \textit{eq.5.1.1}$$

where $\Delta \boldsymbol{D}_{ij} = \left| \boldsymbol{p}_{iend} - \boldsymbol{p}_{ij} \right|$, *eq.5.1.2*

$\boldsymbol{p}_{ij} = (x_{ij},\ y_{ij})'$,

$\boldsymbol{v}_i$ is the direction vector of the ith *MU*,

s_i is the changing factor, +ve sign indicate expansion of *MU*,

$\boldsymbol{p}_{iend}$ is the end-point of the ith *MU*,

subscript *i* indicates *MU* and *j* indicate the target point.

As such, the node which is only connected to i^{th} *MU* can only move in $\boldsymbol{v}_i$ direction. Therefore, it is a constraint by the *Unidirectional MU*.

b) For Circular MU (eyes)

A circular *MU* contracts around an imaginary central point. This converged point is the same as the origin but it will not locate on the face physically. It is only a pseudo-origin and named centroid to distinguish from the origin, which is physically located on the face. If the muscles of eyes are contracted or relaxed, these muscles only change slightly in x-direction. There is very small or even no change about x-direction. So the x-direction will not be considered in the updating equation which is,

$$y_{ij}' = y_{iend} + s_i \cdot (y_{ij} - y_{iend}) \qquad \textbf{eq.5.1.3}$$

In such way, the nodes on the circular *MU* can only move vertically and the scale of change is according to the distance from the centroid of that *MU*.

c) For Another Circular MU (mouth)

Due to the complicated motion of the mouth, more parameters are needed such that it can be modeled well. The updating equation is,

$$x_{ij}' = x_{ij} + b \cdot (x_{ij} - x_{iend}) \qquad \textbf{eq.5.1.4}$$

$$y_{ij}' = y_{ij} + s_i \cdot \left[(x_{ij} - x_{iend})^2 - 0.5a \cdot \left(|x_{il} - x_{ep}| + |x_{ir} - x_{ep}| \right)^2 \right] \qquad \textbf{eq.5.1.5}$$

where b indicate the changing degree in x-direction,

if $a = 0$ fixed mid point (close the mouth),

if s_i indicate the curvature of lip

Eq.5.1.4 is the updating equation for the circular *MU*. Due to the complicated and large motion of mouth, it is very difficult to keep the structure of lips and the smooth curve around the lips. Therefore, a function of quadratic curve is used to preserve this structure.

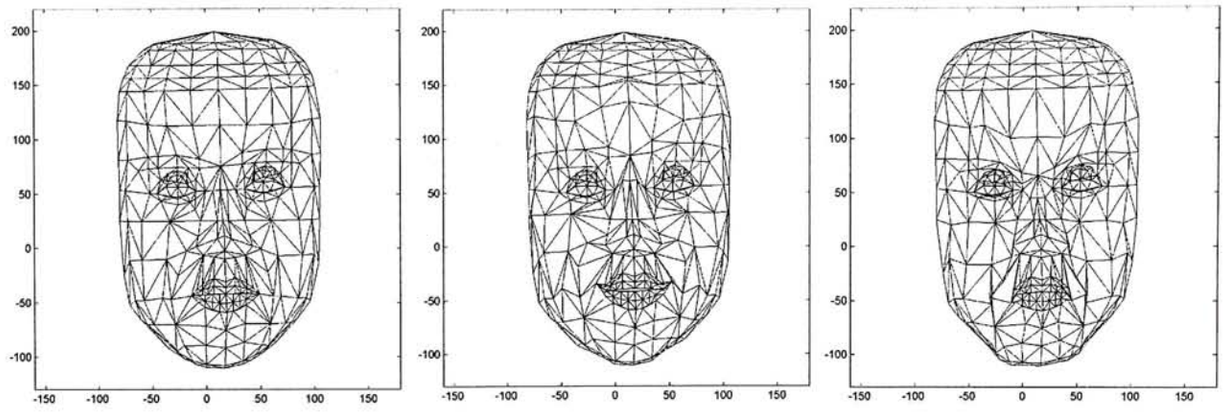
5.1.4. Experimental Results

Table 5.1.2 is the list of scales of *MU* in two expressions, smile and angry. By substituting these scales into ***eq.5.1.1***, ***eq.5.1.3***, ***eq.5.1.4*** and ***eq.5.1.5***, the deformed WFM will get the expressions which are shown in Figure 5.1.6 and Figure 5.1.7,

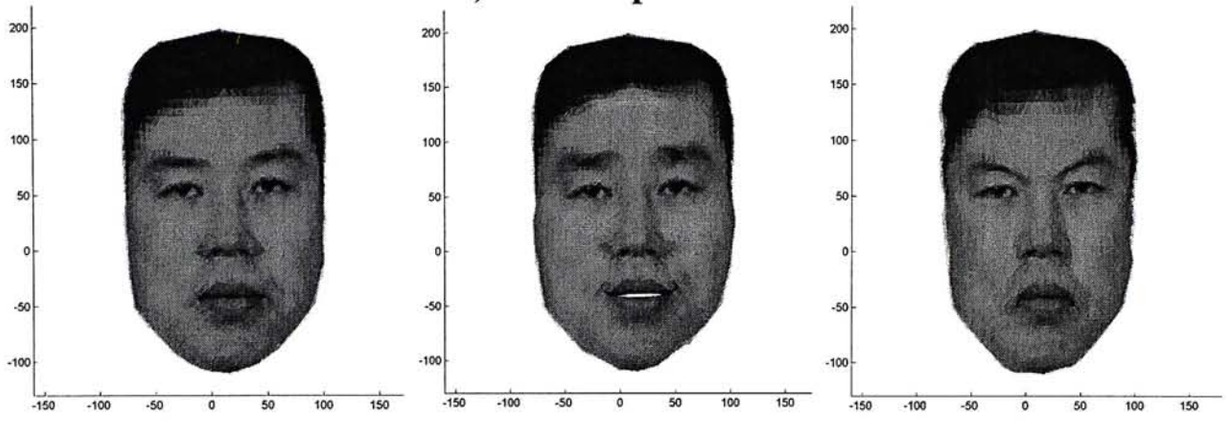
	Laugh	**Angry**
S_1	-3	0
S_2	-3	0
S_3	5	-8
S_4	1.05	0.9
S_5	1.2	0.8
S_6	1.05	0.9
S_7	1.2	0.8
S_8	0	5
S_9	0	5
S_{10}	0	0
S_{11}	0	0
S_{12}	0	0
S_{13}	0	0
S_{14}	-3	3
S_{15}	-3	3
S_{16}	0	2
S_{17}	0	2
S_{18}	3	-3
S_{19}	3	-3
S_{20}	0	0
S_{21}	-0.003	-0.005
A_{21}	1	0
B_{21}	-0.05	-0.05
S_{22}	0.004	-0.005
A_{22}	1	0
B_{22}	-0.05	-0.05

Table 5.1.2: Suggested scales of each *MU* to generate expression

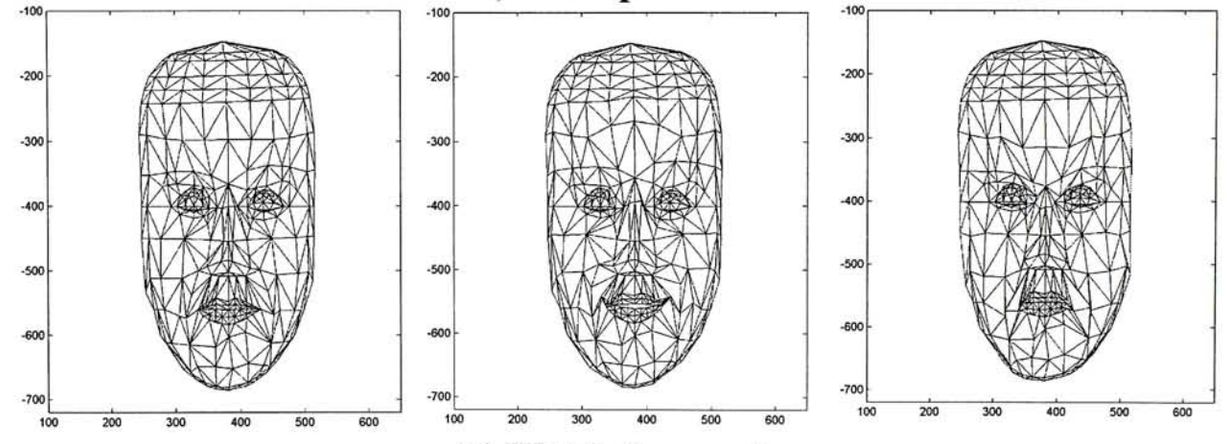
From Figure 5.1.6 and Figure 5.1.7, they show that the generic expressions can be used for everyone, however, the appearance of expression will be different from the original one because only one set of expressions has been declared. So by using this generic expression, there is no difference in expression among all users. For example, person 1 laughs with a large rise of upper lip but person 2 laughs with a slight rise of upper lip, then the two different expressions cannot be distinguished.



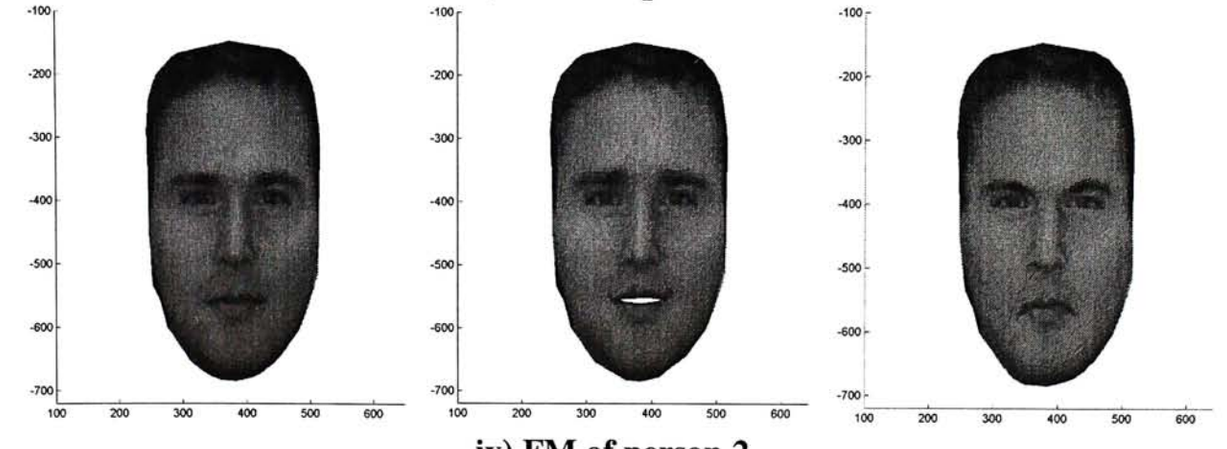
i) WFM of person 1



ii) FM of person 1



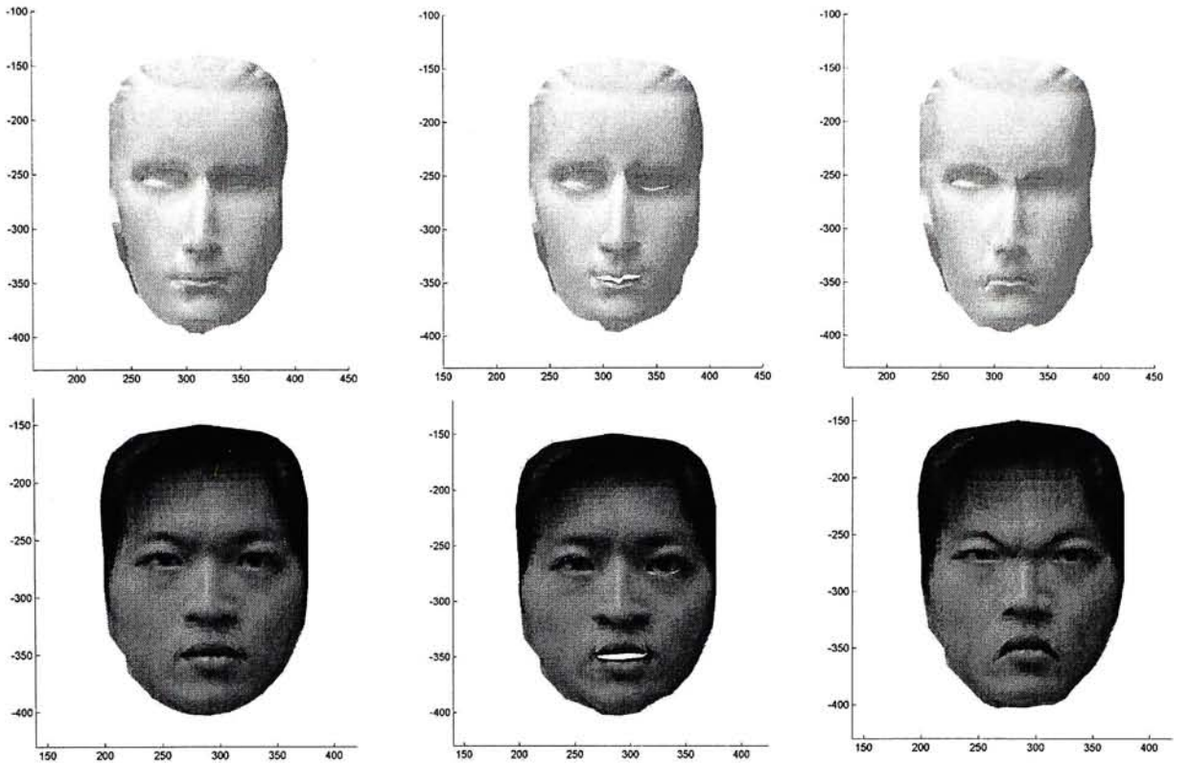
iii) WFM of person 2



iv) FM of person 2

a) Dead pan **b) Smile** **c) Angry**

Figure 5.1.6 : Synthesis of difference facial expression



a) Dead pan **b) Smile** **c) Angry**

Figure 5.1.7 : Other examples by using the same parameters

Some examples about the action of organs

	*MU*s	Corresponding Scale
Raise right outer brow	1, 8	-3, -5
Raise left outer brow	2, 9	-3, -5
Lower right inner brow	3, 8	-8, 4
Lower left inner brow	3, 9	-8, 4
Widen face	14, 15, 18, 19	-3, -3, 3, 3
Contracted face	14, 15, 16, 17, 18, 19	3, 3, 2, 2, -3, -3
Larger right eye	4, 5	1.05, 1.2
Smaller right eye	4, 5	0.9, 0.8
Larger left eye	6, 7	1.05, 1.2
Smaller right eye	4, 5	0.9, 0.8
Smile mouth	21, 22	-0.003(a=1; b=0.05), 0.004(a=1; b=0.05)
Angry mouth	21, 22	-0.005(a=0; b=0.05), -0.005(a=0; b=0.05)

Table 5.1.3: Other suggested *MU*s and corresponding scale on the action of organs

Besides the generation of whole expression, each facial organ can be modeled separately. By following the suggested scales of *MU*s which are shown in Table 5.1.3, different facial motions can be taken place in 3D FM. For example, smaller

right eye will be produced if the *MU*4 with 0.9 and *MU*5 with 0.8. However, these action-based motions will be very similar to that of *FACS*.

5.1.5. Summary

The WFM will be deformed by varying the scale of each *MU*. By combining different *MU*s and mapping texture on the WFM, the FM can give different expressions. However, due to the complicated motion of mouth, more *MU*s must be used to model it and have more freedom on this region.

The movements of nodes are limited by the directions of their connected muscles. If all muscles are defined as different *MU*s and the non-rigid points were back-transformed to the frontal view for comparison, there is no need to recognize the facial action before the synthesis of facial image. Because all the connected muscles of that feature points are known, the movement of muscles can be estimated by the 2D motion of facial image points. As such, the Muscle-Based Approach can give an acceptable facial image. In other words, it is not necessary to follow the same approach as the *FACS*, which is needed to recognize the motion of Action Unit before synthesis of facial image.

5.2. Detection of Facial Expression by Muscle-based Approach

In Facial Action Coded System (*FACS*) [1,13], the procedure is to detect the feature points first, and then carry out recognition [12,14,15] based on the action unit. The detected action unit will be transmitted and regenerated in the receiver side. However, if we can use the concept of motion invariant on face muscles, the facial muscles will give some information about the motion vector for estimating and regenerating the change of expression. Since the movement of nodes is limited by the directions of their connected muscles, it is not necessary to carry out recognition before synthesis. The deformation process can be done following the motions of each *MU*.

5.2.1. Theory

Actually, the motion direction of each facial muscle is defined by human anatomy. Each muscle has a known special function, direction and controllable region. Therefore, the motion direction and magnitude of each node is defined for deformation. It should be noted that a node may be connected to more than one muscle or *MU*. The control point can be used to calculate these combinations of vectors. In addition, if the corresponding scale of vector can be estimated, every point in the same region will be changed in different magnitudes according to its Euclidean Distance from the *origin* (tendon point). It is just like the muscle mechanism, the further point from origin will move more because the displacement of the point will accumulate the changing among the cell. So the displacement will increase with distance. Therefore, based on this mechanism, all the nodes can be modeled by using the above two factors: direction and magnitude. By following

the changing rules of *MU*s, the expression can be produced by deforming the 3D WFM.

5.2.2. Algorithm

By using the same *MU*s, which are defined in the previous section, all information of nodes are declared by the connected *MU*s. Therefore, the motion direction and displacement of each node are constrained by their connected *MU*s. In the previous section, there are totally twenty-two *MU*s being defined. Since the changing scale (*contraction and relaxation*) of each *MU* is estimated by the motion of control points, the motion parameter of each *MU* can be found if at least one control point is connected to that *MU* for detecting the motion. Otherwise, the motion of *MU* cannot be calculated.

There are three kinds of muscles, *Sheet (unidirectional)*, *Circular (Sphincter)* and *Mouth Muscle*. The procedures are divided into two steps: the sheet and circular one will be done in the same system and then the mouth muscle will be processed.

5.2.2a. For Sheet Muscle (Unidirectional MU)

Sheet muscle consists of strands of fibers which lay on flat bundles. Whereas the linear muscle has a radical component, a sheet muscle neither emanates from a point source nor contracts to a localized node. In fact, the muscle is a series of almost parallel fibers spreading over an area. For sheet muscle, we can model it by using very simple vector combination. The equation is shown as below,

$$\boldsymbol{p}_{ij}' = \boldsymbol{p}_{ij} + a_1 \Delta_{1j} \boldsymbol{v}_1 + a_2 \Delta_{2j} \boldsymbol{v}_2 + a_3 \Delta_{3j} \boldsymbol{v}_3 \ldots\ldots + a_{16} \Delta_{16j} \boldsymbol{v}_{16} \qquad \textit{eq.5.2.1}$$

$$\delta \boldsymbol{p}_{ij} = a_1 \Delta_{1j} \boldsymbol{v}_1 + a_2 \Delta_{2j} \boldsymbol{v}_2 + a_3 \Delta_{3j} \boldsymbol{v}_3 \ldots\ldots + a_{16} \Delta_{16j} \boldsymbol{v}_{16} \qquad \textit{eq.5.2.2}$$

where $\Delta_{ij} = (\boldsymbol{p}_{ij} - \boldsymbol{p}_{iend}) \cdot \boldsymbol{v}_i = |\boldsymbol{p}_{ij} - \boldsymbol{p}_{iend}| \cos\theta$,

$\boldsymbol{p}_{iend}$ is the origin of *MU i*,

$\boldsymbol{p}_{ij}$ is the feature point *j* with the controllable region of *MU i*,

$\boldsymbol{v}_i$ is the direction unit vector of *MU i*.

Because two *MU*s do not have any control point to indicate the change and two other *MU*s are circular, there are only 16 unidirectional *MU*s. The vector combination of expression will then have 16 components. In the ***eq.5.2.1***, if $\boldsymbol{p}_{ij}$ is only the combination of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, the other vectors will be zero.

5.2.2b. For Circular (Sphincter) Muscle

Unlike linear muscle model, the circular muscle contracts around an imaginary central point. As a result, the surface surrounding the mouth is drawn together like the tightening of material at the top of the string bag. This converged point is named *centroid* to distinguish from the *origin*, which is physically located on the face. The equation for detecting the circular muscle is shown as follow,

$$\delta \boldsymbol{p}_{ij} = a_i (\boldsymbol{p}_{ij} - {}_i\boldsymbol{p}_{cent}) \qquad \text{for } i = 4, 6 \qquad \textit{eq.5.2.3}$$

where ${}_i\boldsymbol{p}_{cent}$ is the centroid of the *MU i*, which is circular muscle.

If the coefficients of each vector of unidirectional *MU*s and scale of circular *MU*s are known, the corresponding expression can be generated by these coefficients. The linear system can be set up as the following,

$$\begin{bmatrix} \delta x_1 \\ \delta y_1 \\ \delta x_2 \\ \delta y_2 \\ \vdots \\ \vdots \\ \delta x_n \\ \delta y_n \end{bmatrix} = \begin{bmatrix} \Delta_{11} v_{1x} & \Delta_{21} v_{2x} & 0 & 0 & 0 & 0 & \cdots\cdots \\ \Delta_{11} v_{1y} & \Delta_{21} v_{2y} & 0 & 0 & 0 & 0 & \cdots\cdots \\ 0 & \Delta_{22} v_{2x} & \Delta_{32} v_{3x} & 0 & 0 & 0 & \cdots\cdots \\ 0 & \Delta_{22} v_{2y} & \Delta_{32} v_{3y} & 0 & 0 & 0 & \cdots\cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \Delta_{4n} v_{4x} & 0 & \Delta_{6n} v_{6x} & \cdots\cdots \\ 0 & 0 & 0 & \Delta_{4n} v_{4y} & 0 & \Delta_{6n} v_{6y} & \cdots\cdots \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ \vdots \\ \vdots \\ \vdots \\ a_{18} \end{bmatrix} \quad \textit{eq.5.2.4}$$

$$\delta \boldsymbol{p} = \boldsymbol{B} \cdot \boldsymbol{a}$$

where row indicated the point and column indicated the *MU*,

n is the number of control point .

After the linear system has been solved, the coefficient of each *MU* will be calculated and the motion of other nodes can be known with both magnitude and direction. This kind of face modeling is just like the contraction and relaxation of muscles.

5.2.2c. For Mouth Muscle

After WFM is deformed by using ***eq.5.2.4***, the deformed WFM should undergo another estimation, which is about the motion of mouth. There are four control points around the mouth. Undoubtedly, the mouth muscle is only a kind of circular muscle, but the complicated motion of the mouth makes it very difficult to model if only one circular muscle has been used. Therefore, the mouth muscle can be divided into upper and lower lips to allow more freedom on motion.

$$\delta \boldsymbol{p}_{ij} = \left(\boldsymbol{p}_{cent}' - \boldsymbol{p}_{cent}\right) \cdot \left| x_i - x_c \right| \qquad \textit{eq.5.2.5}$$

where $\boldsymbol{p}_{cent}$ is the centroid of mouth,

$|x_i - x_c|$ is the magnitude from the centroid in x-direction only.

In modeling the mouth muscle, the distance or magnitude measured in 3D space is not as good as that in 1D space, which is in x-direction only. It is because the motion of mouth can preserve the relative shape of the lips. In order to keep this property, y and z cannot be used in measuring the distance. The following figure shows this property,

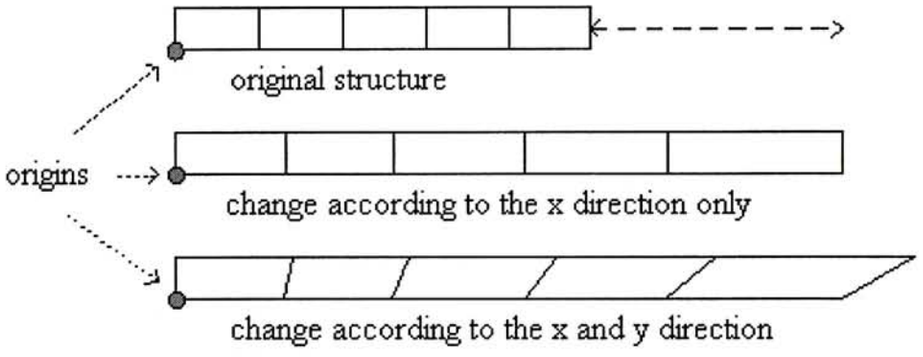


Figure 5.2.1: An example on different distance measurement

In Figure 5.2.1, if the local structure of the model is kept, the distance cannot be calculated in 3D or 2D space for considering the displacement. As similar to that figure, the top one is the original structure of model. When the model is extended, the degree of change is considered by the distance from the origin. For the case of considering both x and y components, the upper layer will move more than that of the lower layer. As such, the local structure cannot be preserved but it will not happen when considering the distance only by x-component. It is because the upper layer is of same distance as the lower layer. It is the reason why only x-direction has been considered for calculating the distance.

5.2.3. Steps of Algorithm

1. ***Initiation***: all the nodes and *MU*s are defined by the human anatomy. As such,

the motion and affected region will be constrained.

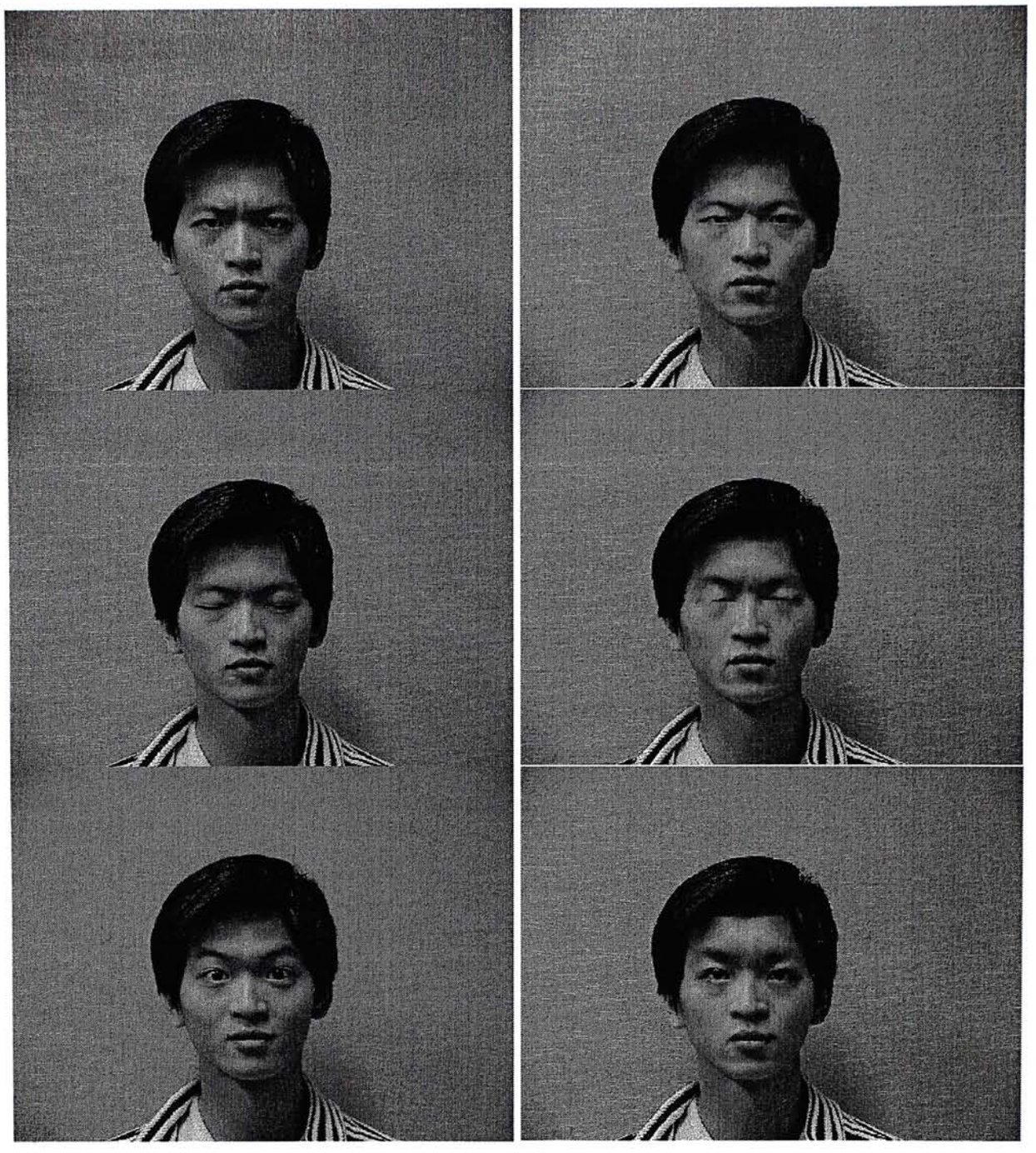
2. ***Detection***: all the control points are detected. Then the motion of control points, which followed the characteristics of physical muscles, can be estimated.
3. ***Analysis***: all the motions of *MU*s are detected by ***eq.5.2.4*** and then ***eq.5.2.5***, and so the contraction and relaxation of each *MU* will be known.
4. ***Synthesis***: follows the motion parameters of *MU*s to deform the 3D WFM and the facial expression can be shown on the FM.

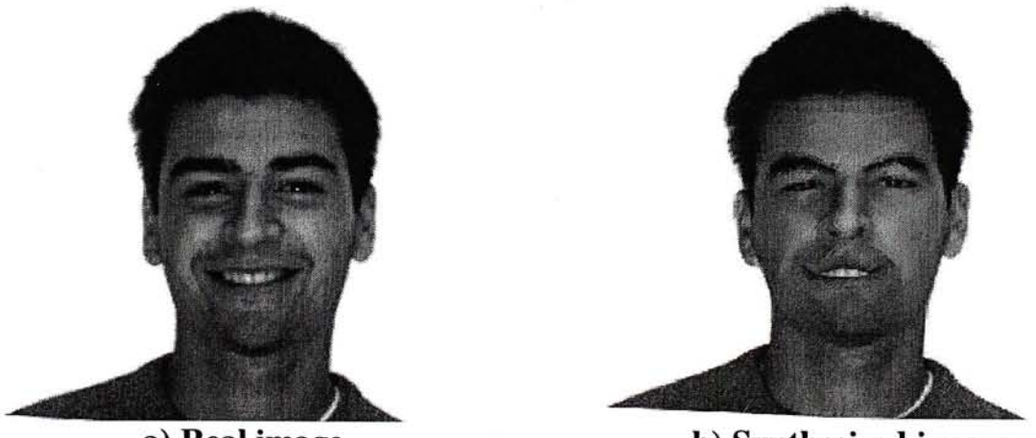
5.2.4. Experimental Results

Two persons with several expressions are used to test the performance of our proposed Muscle-based Method. The deadpan images of these two persons are shown in Figure 5.2.2.

Figure 5.2.2: Deadpan image of examples used in experiments

Figure 5.2.3 shows the results of synthesizing different expressions. Using the Muscle-based Method, the parameters of each *MU* are estimated on the original image. They are used to deform the 3D FM at the receiving end. The synthesized image is obtained by projecting the facial model (deformed WFM after texture mapping) onto the image plane.





a) Real image **b) Synthesized image**

Figure 5.2.3: Results of experiments on Muscle Based Method of expression coding and synthesis. The left image is the original image while the right one is the corresponding synthesized facial image.

5.2.5. Summary

From the above results, the expression detection based on facial motion has to be improved. For having higher image quality, the structure of *MU*s should be slightly adjusted. For example, the connected muscles of nodes should be redefined and some *MU*s may be divided into several more *MU*s for having higher freedom of motion. In this way, the expressions can look more natural and similar to the real one. Also, there is no consideration about the movement of eyeballs and jaw. While eyelids and mouth should be redefined, the definitions of these circular muscles are not enough to model all the facial motion of the organs because they give many different motions. Therefore, the deformed motion should be estimated according to other features like contour, furrow and wrinkle since only feature points cannot represent the motion well.

Chapter 6

Conclusion

In this thesis, we focus on the wireframe fitting and motion estimation on 3D Model Based Video Conferencing. By using these techniques, it is possible to use twenty-six 3D points to modify the wireframe model for fitting an incoming user. Also, by using a set of image points corresponding to those control points, the rigid motions of the head are computed accurately by a novel algorithm. The facial motions of the face are estimated using a set of newly defined *MU*. Unlike the *FACS*, the Muscle Based Approach does not require the difficult recognition of facial expression. Thus, the image at the sending end can be synthesized at the receiving end according to these motion parameters. Very good experimental results are obtained on rigid motion estimation of the head. However, the WFM fitting and the facial motion (non-rigid) estimation are still needed to have further improvements. Some suggestions are proposed in the section 6.4. The following is some detailed comments on WFM fitting, motion estimation and synthesis.

6.1. WFM Fitting

In fitting the WFM to the incoming human face. Only twenty-six 3D control points are used to interpolate a WFM of 300 nodes. Therefore the generated WFM can fit a particular user for videoconferencing. Hence, the WFM is adjusted to match the frontal view of the target face. However, it may not be a good fitting for

that 3D object. The depth error of 3D WFM cannot be detected in a 2D frontal view.

In creating a WFM, the most difficult part is how to handle the nodes which are located on the homogenous regions like cheek, forehead and chin. As no feature points can be detected in these regions, the fitting process cannot adjust the depth of these regions. Therefore, some other features instead of those points should be used to tackle this problem, for example, contour of face, occluding boundary and shadow of cheek.

6.2. Pose Estimation

In this thesis, two methods are proposed to recover the rigid motion of the face: Linear Method and Two Stage Algorithm. The former method requires six corresponding points while the latter one requires only four corresponding points. Both of the obtained matrices are not Euclidean transformation, but both of them can give a very similar image by using this non-Euclidean transformation. In general, Two Stage Algorithm is more robust to noise than the Linear Method when the noise is larger than 1 pixel or when the rotation angle is large.

The obtained matrix from the Linear Method is a mapping transformation matrix while Two Stage Algorithm gives the affine transformation matrix. In the Linear Method, the depth information of WFM is lost and the normal vectors of all nodes will be in the same direction. Therefore, the lighting effect cannot be implemented and the facial image will not appear natural. In the case of Two Stage Algorithm, the affine transformation is not a pure rigid transformation. However,

the affine transformation still involves the 3D motions of object. The depth of 3D points can be kept. Hence, the normal vector of patches can be used to generate the lighting effect.

6.3. Facial Motion Estimation and Synthesis

Based on different actions of *MU*s, the facial motion can be modeled by deforming the WFM. As such, the Muscle-Based Approach can also give an acceptable image. In estimating the facial motion, the direction of *MU*s and its connected nodes is well defined. Since the movement of nodes is constrained by the directions of their connected muscles, there is no need to recognize the facial expression. Because all the connected muscles of a set of feature points are known, the motion of points will be constrained by the characteristic of *MU*s. Movement of muscles can then be estimated by the 2D motion of facial image points.

Moreover, some *MU*s should be modeled by other features, such as wrinkle, furrow and contour. In this way, it can help to detect the motion of muscles which are located in the homogenous regions. The movement of eyeballs and jaw should also be considered for having more facial motions.

6.4. Discussion on Future Improvements

Despite the discussion above, some problems still exist in this system because there is limited time to handle all problems in details. In fact, the ideas on each topic can be further improved in the existing algorithm.

6.4.1. WFM Fitting

In model fitting, it is quite difficult to model the nodes which are too far away from the control feature points, for example, forehead, cheek and chin. No feature point can represent these regions because these regions are homogenous in nature. Hence, no distinguishable feature can be detected and the fitting process will be very difficult to perform well. However, if we can use the occluding boundary of the human head, the approximate structure of those homogenous regions can be obtained. Thus, we use the boundary instead of points to model those parts of the WFM to fit the incoming human head.

The basic idea is that although there are homogenous regions, a boundary can be obtained when the intensity or graylevel of those regions is different between that of the background. If the human head rotates along several arbitrary axes, different occluding boundaries will be observed. When we use the techniques on rigid motion estimation, the corresponding 3D pose of the human head can be found and recovered by transforming the WFM. Another boundary can be obtained by these synthesis images. Therefore, the corresponding boundary of the WFM can be modeled by using the information of observed occluding boundary. Since the coordinates of boundary nodes can be varied to fit that boundary, it will be more accurate than using the closest feature point only.

For the nodes which are not laying on the boundary in the frontal view, we can only adjust the depth of parameter of the boundary nodes. Since they are within the homogenous region, the exact x-y plane position of those nodes is meaningless. Hence, variation of the depth can be compensated for the error on the structure. Figure 6.1 shows that if the occluding boundary of WFM is not the same as the original boundary of real image, this occluding boundary of WFM will suffer error in depth of nodes. By changing the depth z, the output occluding boundary can be corrected and the WFM can fit that person better.

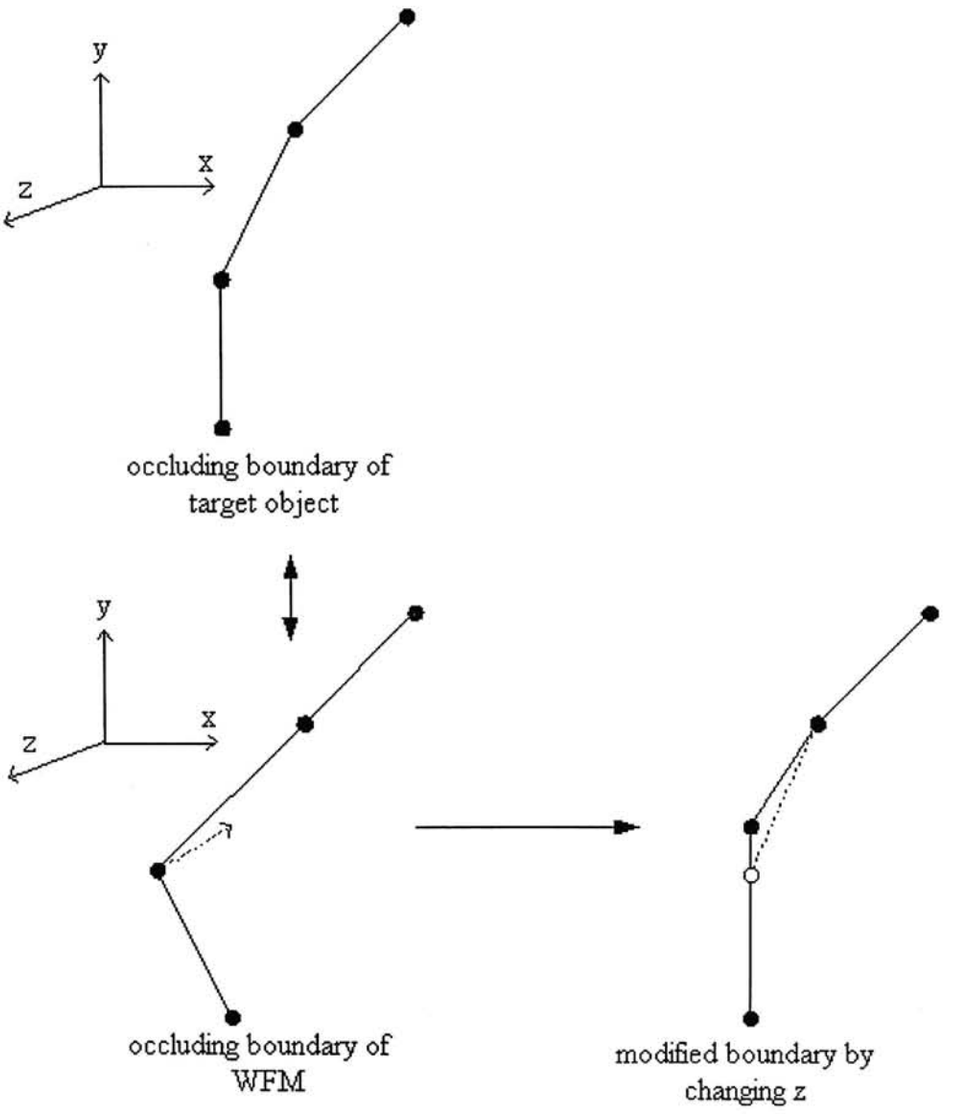


Figure 6.1: Modified boundary by varying z

However, in the case of the nodes laying on the boundary in the frontal view, x and y coordinates of the nodes will be more important. It is because the x and y coordinates of these boundary nodes will greatly affect the appearance of the face. For instance, larger range on x-direction will make the head bigger whereas larger range on y-direction will make the face longer. These kinds of nodes are called outlying nodes in this thesis. Therefore, the adjusted components of those outlying nodes are 3 parameters, x, y and z. The same procedures can be applied in the case of non outer boundary except that one more step is needed: those nodes should vary in the x- and y- direction on the frontal view first. As such, the WFM will enclose the image object in the image sequences.

6.4.2. Pose Estimation

Undoubtedly, the rigid transformation should obey the constraints on rotation matrix, which are

$$\boldsymbol{R} \cdot \boldsymbol{R}^{\mathrm{T}} = \boldsymbol{I} \qquad \textit{eq.6.1}$$

$$\left| \boldsymbol{R} \right| = 1 \qquad \textit{eq.6.2}$$

However, because the scale of transformation matrix is changed according to the distance between the target object and the camera by using Two Stage Algorithm, ***eq.6.1*** and ***eq.6.2*** should be modified. They become,

$$\boldsymbol{R}' \cdot \boldsymbol{R}'^{\mathrm{T}} = s^2 \boldsymbol{I} \qquad \textit{eq.6.3}$$

$$\left| \boldsymbol{R}' \right| = s \left| \boldsymbol{R} \right| = s \qquad \textit{eq.6.4}$$

If these constraints are input into Stage 2 process (non-linear optimization), the obtained transformation will not deform the WFM by affine transformation, which does not have constraint on rotation matrix. Although the obtained affine

transformation is quite close to the Euclidean case, the constraints on rotation matrix may further improve the performance of Two Stage Algorithm.

6.4.3. Facial Motion Estimation and Synthesis

In order to get higher quality of the images, the structure of *MU*s should be slightly adjusted. For example, some *MU*s should redefine their effective regions and some *MU*s may be divided into several more *MU*s for having higher freedom of motion. By doing so, the expressions can be animated more naturally and look more similar to the real one. Also, there is no consideration about the movement of eyeballs and jaw. While eyelids and mouth should be redefined, the definitions of these circular muscles are not enough to model all the facial motion of the organs which give many different motions. Therefore, the motion should be estimated according to other features like contour, furrow and wrinkle. Only feature points available cannot represent the motion well.

Up to now, the feature used in this approach is only based on points. If the *MU*s are located in the homogenous region, e.g. cheek and forehead, no feature point can be detected for estimating the motion of these *MU*s. Therefore, if we want to know the motion of these *MU*s, other features should be introduced in this system for correct estimation of all *MU*s.

Chapter 7

Appendix

7.1. Newton's Method or Newton-Raphson Method

Consider a given function $f(x)$ of one real variable x. The stationary points of the function $f = f(x)$ are to be found, that is, the values of x which satisfy

$$\frac{df}{dx} = g(x) = 0 \qquad \textit{eq.7.1}$$

are desired.

Newton-Raphson search is an iterative technique which can be used to find zeros of $g(x)$. The search consists of a series of trials, and each trial consists of the following steps:

1. Take the value x_k of x obtained from kth trial (x_0 is picked by a judicious guess) and evaluated $g(x_k)$ and $g'(x_k)$.
2. Fit a straight line to the data obtained in step 1.
3. Determine the point x_{k+1} at which the straight-line approximation intersects the x axis. This point x_{k+1} is an approximation to a stationary point and is used as the initial point for a new trial.

These steps are now expressed analytically for convenience computation. In Figure 8.1, it is to be observed that

$$g'(x_k) = \frac{g(x_k)}{x_k - x_{k+1}} \qquad \textit{eq.7.2}$$

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)}$$

$$= x_k - \frac{f'(x_k)}{f''(x_k)} \qquad \textit{eq.7.3}$$

An important property of this technique is that it yields quadratic convergence. Quadratic convergence is that property of a search technique which enables a minimum (maximum) of a quadratic function to be located exactly in a finite number (usually a number equal to the number of variables) of steps, subject to round-off errors.

To prove quadratic convergence in this one-dimensional case, assume that f is of the quadratic form

$$f = ax^2 + bx + c \qquad \textit{eq.7.4}$$

By employing ***eq.7.4***, it follows that

$$x_1 = x_0 - \frac{2ax_0 + b}{2a}$$

$$= \frac{-b}{2a} \qquad \textit{eq.7.5}$$

which can readily verify to be the exact minimum point (if $a>0$).

In general, the property of quadratic convergence is an important attribute for a search technique. To substantiate this statement, recall that any differentiable function $f(x)$ can be expanded in a Taylor's series about a given relative extreme point. The first three terms of such a series, the terms which form a quadratic, are dominant for any x in the near vicinity of the extreme point. Thus, using a quadratic-convergence search technique insures rapid convergence once the near vicinity of the extreme point is reached.

Note that in the vicinity of an extreme point, the second derivative $g'(x)$ of $f(x)$ is constant in sign, whereas the first derivative $g(x)$ of $f(x)$ changes in sign from one side of the extreme point to the other. The first derivative $f'(x)$ is a one-dimensional gradient of $f(x)$ and, depending on the sign of the gradient $f'(x_k)$, the k+1th value x_{k+1} of x is either smaller or larger than the kth value. Thus, the gradient of f determines the direction of change in x. This method is therefore related to n-dimensional gradient methods of search.

7.2. H.261

Recognizing the need for providing ubiquitous video services using the Integrated Services Digital Network (ISDN), CCITT (International Telegraph and Telephone Consultative Committee) Study Group XV established a Specialist Group on Coding for Visual Telephony in 1984 with the objective of recommending a video coding standard for transmission at m × 384 kbit/s (m=1,2,..., 5). Later in the study period after new discoveries in video coding techniques, it became clear that a single standard, p × 64 kbit/s (p = 1,2,..., 30), can cover the entire ISDN channel capacity. After more than five years of intensive deliberation, CCITT Recommendation H.261, Video Codec for Audiovisual Services at p × 64 kbit/s, was completed and approved in December 1990. A slightly modified version of this Recommendation was also adopted for use in North America.

The intended applications of this international standard are for videophone and videoconferencing. Therefore, the recommended video coding algorithm has to be able to operate in real time with minimum delay. For p = 1 or 2, due to severely limited available bit rate, only desktop face-to-face visual communication (often referred to as videophone) is appropriate. For p>=6, due to the additional available

bit rate, more complex pictures can be transmitted with better quality. This is, therefore, more suitable for videoconferencing.

7.3. 3D Measurement

If we want to construct a 3D WFM, 3D coordinates of the points or nodes should be measured first. For a highly accurate measurement, laser scanning [9] or three-dimensional digitizers [31] should be used to get the 3D information of points.

7.3.1. Three-Dimensional Digitizers

Three-dimensional digitizers are special hardware devices that rely on mechanical, electromagnetic, or acoustic measurements to locate positions in space. The electromagnetically based *Polhemus 3Space digitizer* is probably the most widely used device for this purpose.

These digitizers require physically positioning a three-dimensional probe or locator at each surface point to be measured. The digitizing process requires considerable time if a large number of points are to measure. The surface points measured may be the vertices of a polygon network or the control points for a parametric surface. Since real faces tend to move and change shape over time, digitizers work best with sculptures or physical models that do not change shape during the measuring process.

7.3.2. Laser Scanning System

Laser-based surface scanning devices, such as those developed by *Cyberware* [13], can be used to measure faces. These devices typically produce a very large regular mesh of data values in a cylindrical coordinate system. They can digitize

real faces as well as physical models and facial sculptures. Facial poses must be held constant during the short time needed to complete the scanning process. Newer versions of these scanning systems can simultaneously capture surface color information in addition to surface shape data.

Postprocessing of the scanned cylindrical data mesh is often required to extract the surface data needed for a particular face model. This postprocessing may include data thinning, spatial filtering, surface interpolation, and transformations from cylindrical to Cartesian coordinate systems.

Bibliography

[1] K. Aizawa, T. Saito, and H. Harashima, "Model-based analysis synthesis image coding (MBASIC) system for a person's face", Signal Process. : Image Commun. , 1989, 1, (2), p.139 - 152.

[2] T. Akimoto, Y. Suenaga, R.S. Wallace, "Automatic creation of 3D facial models", IEEE Computer Graphics and Applications, Volume: 13 5, Sept. 1993, p.16 –22.

[3] P.M. Antoszczyszyn, J.M. Hannah, P.M. Grant, "A comparison of Detailed Automatic Wire-frame Fitting Methods", Proceedings, International Conference on Image Processing, 1997, p.468 -471 vol.1.

[4] G. Ashraf, M.N. Chong, "Performance analysis of H.261 and H.263 video coding algorithms", Proceedings, IEEE International Symposium on Consumer Electronics, 1997, p.153 –156.

[5] G. Bozdagi, A.M. Tekalp, L.Onural, "Simultaneous 3-D motion estimation and wire-frame model adaptation including photometric effects for knowledge-based video coding", Processing, IEEE International Conference on Acoustics, Speech, and Signal ICASSP-94, Volume: v, 1994, p.V/413 -V/416 vol.5.

[6] Feng Chen, J.D. Villasenor; Dong Seek Park, "A low-complexity rate-distortion model for motion estimation in H.263", Proceedings, International Conference on Image Processing, 1996, Volume: 1, 1996, p. 517 -520 vol.2.

[7] L.S. Chen, T.S. Huang, J. Ostermann, "Animated talking head with personalized 3D head model", IEEE First Workshop on Multimedia Signal Processing, 1997, p.274 –279.

[8] M.K. Cheung, H.T. Tsui, "Image Motion Estimation for 3D Model Based Video

Conferencing", Proceedings of the 15th International Conference on Pattern Recognition, 2000.

[9] Cyberware Laboratory Inc, 4020/RGB 3D Scanner with Color Digitizer, Monterey, CA, 1990.

[10] E.A.Da Silva, M. Ghanbari, "A DCT-based aliasing cancellation method in subband coding", IEEE Transactions on Circuits and Systems for Video Technology, Volume: 3 5, Oct. 1993, p.384 –387.

[11] Li Ding, K. Takaya, "H.263 based facial image compression for low bitrate communications", Proceedings, Conference on Communications, Power and Computing WESCANEX 97, p.30 –34.

[12] G. Donato, M.S. Bartlett, J.C. Hager, P. Ekman; T.J. Sejnowski, "Classifying Facial Action", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: 21 10, Oct. 1999, p.974 –989.

[13] P. Ekman, W.V. Friesen, "Facial Action Coding System", Consulting Psychologists Press Inc., 577 College Avenue, Palo Alto, California 94306, 1978.

[14] I.A. Essa, A.P. Pentland, "Facial expression recognition using a dynamic model and motion energy", Proceedings, Fifth International Conference on Computer Vision, 1995, p.360 –367.

[15] I.A. Essa, A.P. Pentland, "Coding, analysis, interpretation, and recognition of facial expressions", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: 19 7, July 1997, p.757 –763.

[16] O. Faugeras, "Three-Dimension Computer Vision, a Geometric Viewpoint", The MIT Press, Cambridge, 1993.

[17] T. Fukuhara, A Umahashi, T. Murakami, "3-D motion estimation for model-based image coding", International Conference on Image Processing and its

Applications, 1992, p.69 –72.

[18] Van De Graaff, Kent M, "Human anatomy", Dubuque, IA: Wm. C. Brown, 1995.

[19] Richard I. Hartley, "Projective reconstruction and Invariants from Multiple Images", IEEE Trans. Pattern Anal. Machine Intell., vol.16, no. 10, 1994.

[20] Richard I. Hartley, "In Defense of Eight-Point Algorithm", IEEE Trans. Pattern Anal. Machine Intell., vol. 19, no. 6, 1997.

[21] A. Heyden, K. Astrom, "Euclidean reconstruction from image sequences with varying and unknown focal length and principal point", Proceedings, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, p.438 –443.

[22] A. Heyden, "Reconstruction from image sequences by means of relative depths", Proceedings, Fifth International Conference on Computer Vision, 1995, p.1058 –1063.

[23] A. Heyden, K. Astrom, "Euclidean reconstruction from constant intrinsic parameters", Proceedings of the 13th International Conference on Pattern Recognition, 1996, Volume: 1, p.339 -343 vol.1.

[24] T.S. Jebara, A. Pentland, "Parameterized structure from motion for 3D adaptive feedback tracking of faces", Proceedings, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, p.144 –150.

[25] H.C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections", Nature, vol. 293, p.133 – 135, 1981.

[26] R. Mech, M. Wollborn, "A noise robust method for segmentation of moving objects in video sequences", ICASSP-97, IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume: 4, 1997, p.2657 -2660 vol.4.

[27] B. Nagel, J. Wingbermuhle, S. Weik, C.-E. Liedtke, "Automated modelling of

real human faces for 3D animation", Proceedings, Fourteenth International Conference on Pattern Recognition, 1998, p.693 -696 vol.1.

[28] Frederic I. Parke and Keith Waters, "Computer facial animation", Wellesley Massachusetts: A K Peters, 1996.

[29] Donald A. Pierre, "Optimization Theory with Applications", John Wiley & Sons, inc., New York, 1969.

[30] C.J. Poelman, T. Kanade, "A paraperspective factorization method for shape and motion recovery", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: 19 3, March 1997, p.206 –218.

[31] Polhemus Navigations Sciences, 3Space Lsotrack Users Manual, Colchester, VT, 1987.

[32] M. Pollefeys, R. Koch, L. Van Gool, "Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters", Sixth International Conference on Computer Vision, 1998, p.90 –95.

[33] M. Pollefeys, R. Koch; M. Vergauwen, L. Van Gool, "Flexible 3D acquisition with a monocular camera", Proceedings, IEEE International Conference on Robotics and Automation, Volume: 4, 1998, p.2771 -2776 vol.4.

[34] R. Rajagopalan, M.T. Orchard, R.D. Brandt, "Motion field modeling for video sequences", Image Processing, IEEE Transactions on Volume: 6 11, Nov. 1997, p.1503 –1516.

[35] V. Ruiz, V. Fotopoulos, A.N. Skodras, A.G. Constantinides, "An 8x8-block based motion estimation using Kalman filter", Proceedings, International Conference on Image Processing 1997, p.140-143 vol.2.

[36] I. Shimizu, Z. Zhang; S. Akamatsu; K. Deguchi, "Head pose determination from one image using a generic model", Proceedings, Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998, p.100-105.

[37] Ru-Shang Wang, Yao Wang, "Facial feature extraction and tracking in video sequences", IEEE First Workshop on Multimedia Signal Processing, 1997, p.233 –238.

[38] Gang Xu, Zhengyou Zhang, "Epipolar Geometry in Stereo, Motion and Object Recognition: A Unified Approach", Kluwer Academic Publishers, 1996.

[39] Jiajun Zhang, M.O. Ahmad, M.N.S. Swamy, "A new variable size block motion compensation", Proceedings, International Conference on Image Processing, 1997, p.164 -167 vol.2.

Contribution

1. Two Stage Algorithm

We proposed a novel of Two Stage Algorithm [8]. The Stage 1 Method has better results than the linear method of projection matrix. Also, the required correspondences of the Stage 1 Method are only four while those of linear method are six. After the Stage 2 Method, very good rigid motion parameters can be obtained.

2. Motion Unit (Muscle Based) Approach

As far as we know, the motion unit approach in the past is mainly used for animation. There is no report on how to use it to estimate the facial motion directly. In this thesis, we propose to use the constrained motion direction of each non-rigid control point to estimate the action of each *MU* (muscle). In this way, recognition of expression is not necessary in the facial animation.