Studies on Facial Surface Reconstruction from Image Correspondence

Cheng Kin Shing Dominic

A dissertation submitted to
the University of Hong Kong
in partial fulfillment of the requirements
for the degree of Master of Philosophy

August, 2000

Abstract of thesis entitled
Studies on Facial Surface Reconstruction from Image Correspondence

submitted by
Cheng Kin Shing Dominic

for the degree of Master of Philosophy
at The University of Hong Kong
in August 2000

Nowadays, many computer applications involve the use of facial surfaces. Computer generated facial surfaces in movies, commercials and electronic games are examples. So, reconstruction of facial surfaces can easily find its application. However, either special equipment is used or a generic facial model is involved in many of the previous approaches to facial surface reconstruction. To get rid of the special equipment, my research was aimed to build a three-dimensional facial model only from the correspondence information between images. The whole research can be roughly divided into two parts: image correspondence and surface reconstruction.

For the image correspondence part, the inputs are the images with background. Facial regions are extracted from the images, followed by the extraction and matching of facial features. A new matching criteria function is derived to compute the matching score between the features. The matching criteria used include the color value, the gradient value as well as the Laplacian value. The weighting of each criterion for matching a point depends on the uniqueness of the criterion of that point. Besides the matching process, the accuracy and speed-up issues are also considered.

For the surface reconstruction part, a new approach to reconstructing a facial surface using gridlines is proposed. Grid points, horizontal and vertical gridlines are defined in the frontal view image. From the correspondence information between images, the x, y coordinates of these grid points in other views are observed. Two values, horizontal and vertical point proportions, are defined at each grid point. From the relationships between the point proportions and the gridline parameters, the z-values of all the grid points are determined. Finally, after triangulation, a facial surface can be obtained.

One advantage of this reconstruction approach is that only simple computation is involved. Furthermore, the only inputs of this reconstruction approach are three images (one of them is the frontal image) and the correspondence information between the images. However, when some grid points do not appear in the frontal image, they will not appear in the reconstructed model. In addition, one free parameter needs to be set interactively to obtain the final surface produced by this approach.

To My Family

Content

Chapter 1. Introduction

1.1 Project Objective

In this project, the objective is to reconstruct a facial surface using the correspondence information between the images taken by a digital camera or a DV. It should be an economic way for facial modeling that does not involve the use of expensive laser range finders. Moreover, a generic face model that will restrict the shape of the reconstructed model will not be used.

1.2 Problem definition

The problem can be divided into two subproblems as shown in Figure 1-1.

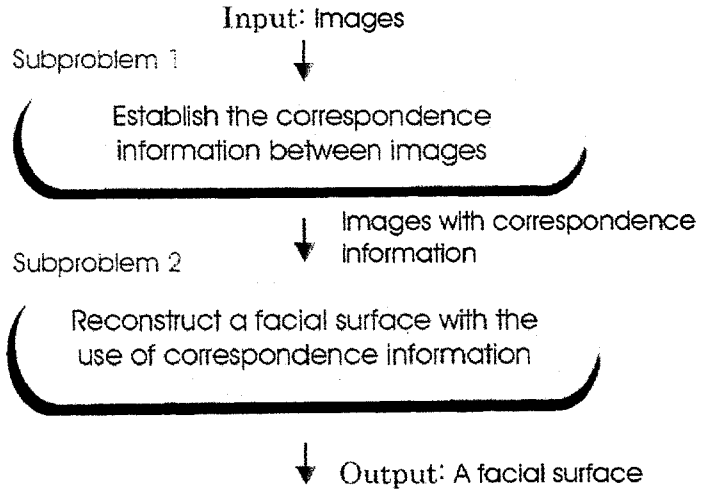

Figure 1-1: **Subproblem 1 & Subproblem 2**

1.2.1 Subproblem 1

The first subproblem is to obtain the correspondence information between images. For this subproblem, both the accuracy and the performance issues of the matching process are considered.

1.2.2 Subproblem 2

The second subproblem is to reconstruct the three-dimensional facial surface from the correspondence information between the images. The main essence of the proposed approach is to obtain the depth information from the positions of the defined grid points using the images from different views.

1.3 Thesis Organization

In this thesis, after this 'Introduction' chapter, there follows the 'Related Work' chapter. In the 'Related Work' chapter, related previous techniques used throughout the process of image matching are briefly mentioned. Also, different previous approaches to modeling a facial surface are described.

Then, it comes the 'Image Correspondence' chapter. The process of establishing the correspondence information is described in details. It includes:

- Extraction of the facial region from an image
- Extraction of feature points
- The matching criteria function
- Matching of feature points
- Matching of non-feature points
- Accuracy issues
- Speed-up issues

Next, it is the 'Surface Reconstruction from Image Correspondence' chapter. A proposed approach of reconstructing the facial surface is described. It includes:

- Definition of grid points and gridlines
- Depth hints from the side view images
- Definition of point proportions
- Relationship between point proportions
- Practical simplification with assumptions
- Result
- Cons and Pros

After that, the 'Conclusion' chapter gives a conclusion for the whole project, the 'Future Work' chapter includes some of the possible future extensions and a list of references is given in 'Bibliography'.

Figure 1-2 below shows the outline for the thesis:

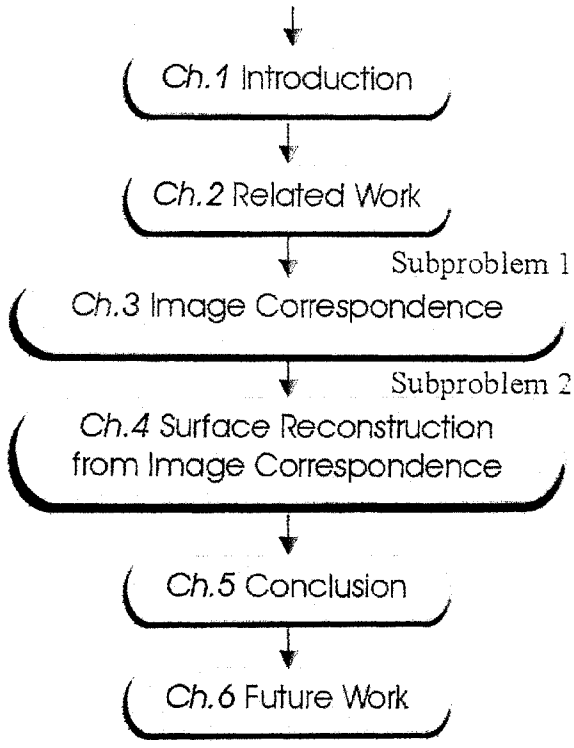

Figure 1-2: **The thesis organization**

Before the end of this chapter, I would like to give brief descriptions for the two subproblems here.

Subproblem 1 – Image Correspondence

The first subproblem is about the establishment of the image correspondence between images. The first step is to extract the face regions from the images and methods to achieve this goal are described. Then, feature points in the images are extracted using an algorithm from a literature. Next, the matching components used in the matching criteria functions for computing the matching score between two points are mentioned. Then, an approach of computing the weightings of different criteria in the matching criteria function for each point is introduced. After that, procedures for matching feature points and non-feature points are described.

Besides, both speedup and accuracy issues are addressed. Speedup issues include the ways for reducing the searching space for the corresponding point and multiresolution matching. Accuracy issues include the consideration of computing the color difference using the values in a uniform color system. Moreover, matching error detection methods, such as "double-direction checking", "pixel order preserving" and "quad preserving", are discussed.

Subproblem 2 – Surface Reconstruction from Image Correspondence

The second subproblem is about the reconstruction of a facial surface from image correspondence between the images. An approach to reconstructing a facial surface using the correspondence information is proposed. The minimum number of images required for the reconstruction is three and one of them needs to be a frontal view. The first step in the reconstruction process is to define grid points in a frontal view image. Horizontal and vertical gridlines are formed by connecting the grid points along the horizontal and vertical directions respectively. Then, two additional images are required. With the use of the correspondence information, the positions of the grid points in other two images are identified. From the positions of these grid points in different views, two values called point proportions (will be defined in Chapter 4) are obtained at each grid point. Equations are formed with these point proportions. Then, the z-values at all the grid points can be obtained by solving these equations. Under an assumption about the shape of the facial surface, the computations can be simplified. Reconstructed facial surfaces are shown in Chapter 4. Also, the advantages and limitations of this approach are discussed.

Chapter 2. Related Work

In this chapter, the related previous work is described. Since the objective of this project is to produce a facial surface, several approaches to modeling a human face will be described. On the other hand, as stated in the "Problem definition" in the "Introduction" chapter, the whole project is divided into two parts and the first part is about how to establish correspondence information between two images. So, in additional to different approaches to constructing a facial surface, the related work about how to establish the correspondence information between images is also included in this section.

2.1 Establishment of Image correspondence

In this section, related work about the process of establishing image correspondence between images is mentioned.

2.1.1 Image Preprocessing

In this project, the input images come from a video sequence of a moving head. However, the images cannot be used immediately due to the presence of noise. So, images need to be processed using the techniques described in [h3, h9, h10, h12] so as to minimize the influence of noise.

2.1.2 Facial Region Extraction

After the image has been processed to be free of noise, the facial region in the image needs to be extracted from the image that includes the background. Techniques used to detect and extract the face region in the images can be found in [a1, a2, a5]. A range of color values can be predefined as the face color. Alternatively, the face color can be estimated from the color histograms of the image. After that, the face region can be identified by locating the face color in the image. Another way to extract the face region can make use of a face template [a3, b1]. The region that gives the best matching result to that face template will be regarded as the face region.

2.1.3 Feature Extraction

Features need to be extracted from the image for the use in the image matching process. Edges and points are two example types of features. For locating edges, a gradient filter, a Laplacian filter or the Canny detector can be applied to the image and then the resultant image will provide hints of the edges [h3, h6, h8, h9, h10, h12]. Another way to detect edges in the image is to perform minimization on energy functions such as snakes [a6, f21]. For points (or sometimes called corners) identification, intersections of edges may yield the results. Sometimes, corners can be found at the points where the image edges have local maximum absolute curvatures

there [a4]. Furthermore, edge directions and edge strengths at a point can give some hints on whether the point is a special point or not [h12]. The process of obtaining edge directions and strengths involves the computations of the gradient values along both vertical and horizontal directions at each point. Besides detecting the features by applying filters or carrying out computations on the gradient values, templates can be used for feature extraction [a7]. In some cases, special points such as pupils can be found using a tracker that is specially designed for that particular organ [a1]. Indeed, the performances of different feature extraction methods depend on the properties of the images.

2.1.4 Feature Matching

Obviously, after the features have been extracted, the next step is the matching of the features. There are many classical descriptions on the matching of images [h3, h9, h10, h12]. Feature based methods, correlation based methods and template matching methods [b1] are common approaches to carrying out feature matching. Techniques such as singular value decomposition [b2] can also be applied in feature matching.
On the other hand, epipolar constraint holds when two images are taken for the same object. An epipolar line passing through the epipole on the first image will match to another line on the second image. This facilitates the process of feature matching [b3] because the epipolar constraint reduces the problem to one-directional searches along lines. In order to make use of the epipolar constraint, the camera parameters need to be known or to be estimated from the images.

The performance of the matching process can be improved if some techniques are applied. A multiresolution approach [c2, c5, c6] is a common way to achieve that. In multiresolution approaches, images need to have different levels of image representations (which are of different resolutions). For examples, a Laplacian pyramid is a multiresolution representation of an image in [c2] and the "min-max" images are used in [c6]. Also, wavelets are good multi-level representations of images [c3]. Basic knowledge of wavelets can be found in [c1, c4, c7, c8, c9, h11].
Indeed, an optimal image matching method is not easy to be found and the performance of different matching methods can vary with different sets of inputs.

2.2 Face modeling

After describing the related work about the establishment of the feature correspondence between images, different approaches of modeling a facial surface are briefly mentioned in this section.

2.2.1 Use of a 3-D range scanner

One of the most obvious approaches to construct facial surfaces involves the use of 3-D range scanners [f10, f21]. With the use of it, both the color values and the range data of the points can be obtained easily. A 3-D face model can be built by combining the color values and the range information. The advantage of using a 3-D range scanner is that the modeling process becomes trivial. But, the equipment itself is still expensive for it to become popular for home uses.

2.2.2 Use of a generic face model

Another common approach to building a facial model is to make use of a generic face model [f2, f3, f4, f6, f7, f8, f9, f23, f25]. Then, the generic face model is deformed by the images of a person to become an individual face model. The advantage of using a generic model is that the process becomes relative easy and only a deformation process is required for constructing a new face model. But, the generic face model will restrict the individual face model in terms of the shape and also the details. Moreover, some people get face shapes that are very different from others. In these cases, deformation to a generic model will not be able to yield acceptable individual face models.

2.2.3 Use of camera information

The 3D coordinates of the points on the face model can be computed by solving the projection equations provided that sufficient camera information is available. Relevant information can be found in the literatures that involve pose estimation [d1, d2, d3, d4] and stereo reconstruction [e1, e2, e3, f1, f7, f11, f15, f19, h5].

2.2.4 Lighting Switch Photometry

'Lighting Switch Photometry' [f18] is a method of building 3-D faces that involves the use of structured lightings. In this method, the lightings information (includes the positions and intensities of the lights) is assumed to be known. If the reflection properties of the surfaces are also assumed to be known, the normal vectors can be computed by solving the lighting equations. After the normal vectors have been obtained, 3-D surfaces can be fitted into the normal vectors to form a 3-D face model. The shortcomings of this method are the involvement of the special lighting equipment and the need of estimation for the reflection properties of the surfaces.

Indeed, the building of 3D face models can also involve the use of differential properties [f11, f12, f13, f15], minimization of other energy functions [f12, f14], 2D silhouettes [f6] or the use of singularities [f20]. Besides, more literatures or findings about the reconstructions of 3-D models from shading, motion, etc. can be found in [f1, f26, h1, h4, h5, h6, h7, h8, h10, h12].

Since one of the common applications of facial models is in the field of computer animations, some previous work [f3, f7, f9, f10, f21, f22, f25] has been done for making face models that can be adapted to animations. Some of them added facial tissues and muscle actuators in order to make the face models more realistic.

After briefly describing different approaches to modeling face models, I will state our main considerations regarding the facial surface reconstruction part of this project. We decided not to use a 3-D laser range finder, not to use a generic face model and not to use special structured lights.

In the next two chapters, the process that has been gone through in the project is described in details.

Chapter 3. Image Correspondence

In this chapter, the process for establishing image correspondence between images is described. This problem itself is a classical problem in the field of computer vision and image processing. In this project, image correspondence is the first subproblem that needs to be solved and the result is used in the reconstruction process described in Chapter 4.

The first task is to locate and extract the face regions from the images and two ways to achieve that are introduced. Then, feature points in the images are extracted using an algorithm in [h12]. Next, a matching criteria function is derived for computing the matching score between two points. The main idea is to adjust the weightings of different matching criteria according to the uniqueness of those criteria on the matching point. After that, the procedures of matching feature points and non-feature points are described. Finally, both speedup and accuracy issues are addressed. Speedup issues include the ways for reducing the searching area for a corresponding point and multiresolution matching. Accuracy issues include the consideration of computing color difference using values in a uniform color system. Also, matching error detection methods, such as "double-direction checking", "pixel order preserving" and "quad preserving", are discussed.

3.1 Facial Region Extraction

In this section, two approaches to extracting the face regions from the images are described. The objective is to remove the background scene from the images. One makes use of the gradient values and another makes use of the difference between consecutive images.

3.1.1 Video-taking

In this project, the input images come from a video file consisting of head movements. The set up is very simple. A DV is used to take the video. The person sits in front of the DV (the position of the DV is fixed) and makes up-to-down and left-to-right head movements. Figure 3-1 shows some images from a video file.

3.1.2 Video file manipulation

After taking a video, a video file consisting of up-to-down and left-to-right head movements is obtained. The video file is of *.mov* extension. To extract individual

images from the video file, several routines in a library called "Digital Media Library" in the SGI machines are called.

Besides extracting the images from the video file, the gradient values and the Laplacian values are computed. These values are used in the process of extracting the face regions as well as the process of features matching. The methods for computing the gradient and the Laplacian values by applying filters can be found in [h3, h6, h8, h9, h10, h12].

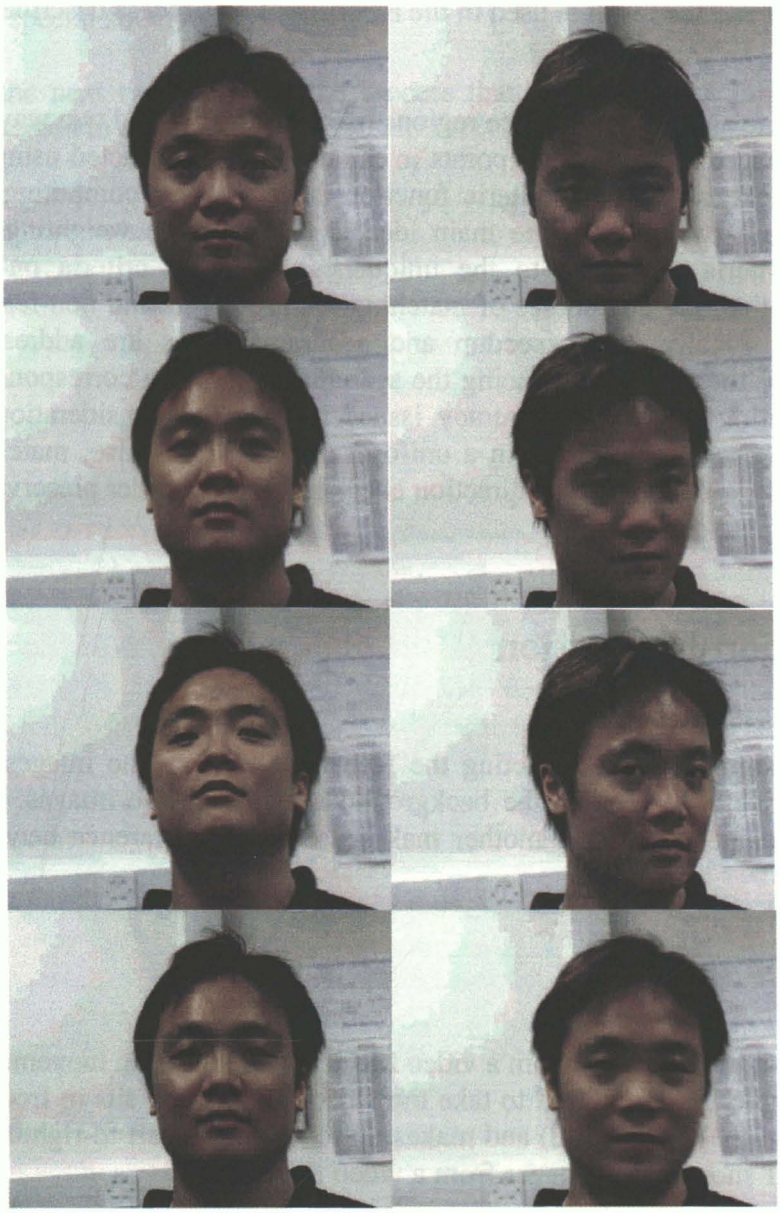

Figure 3-1: **Images from a video file**

In Figure 3-4, the edge points found along the specified scanlines are shown. In the right image, the three scanlines are close to each other. However, the position of the right edge point found along the scanline 2 is quite different from those of the right edge points found along the scanlines 1 and 3. It is because there exists a point that belongs to the background scene in the gradient image along the scanline 2. In this case, the edge position of the right edge point on the scanline 2 is computed using the edge positions on the scanlines 1 and 3.

After the edge points for each scanline are obtained, the points lie between the two edge points are considered to be the points within the facial region. The points outside the two edge points are considered to be the points on the background scene and can be removed. Figure 3-5 below shows the images before and after the facial region extraction.

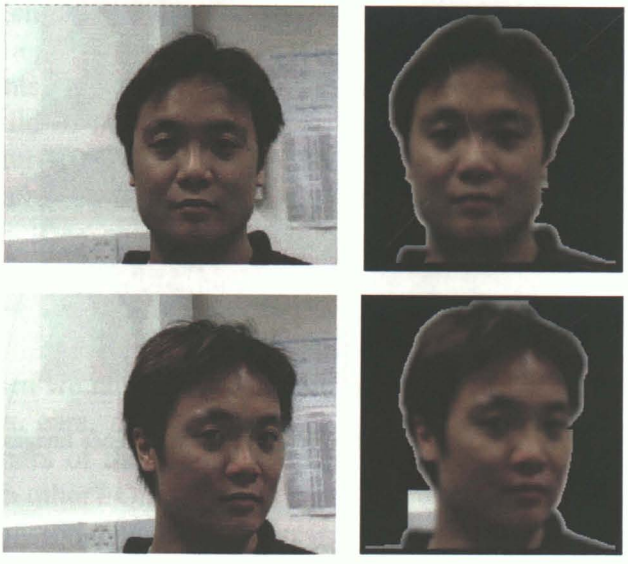

***Figure 3-5*: Images before the removal of the background scene and after the removal of the background scene**

It can be observed that the assumption that edges come from the facial part is not always valid. Indeed, some points on background objects that have the gradient values greater than the preset threshold will lead to the presence of the false edges points. This problem can be solved if other criteria (besides the gradient values) are used to detect the edge point. One obvious choice is the color of the face. A range of color can be set for the face region and then the edge points are identified along each scanline according to the preset color. But, even if the facial color is used to locate the edge points, there can still have some problems. Firstly, how can a color range be set for the facial region? People of different races will have different colors. Moreover, if the color value is the criterion, the background scene needs to avoid of that color range corresponding to the facial region. Otherwise, false edge points will be detected because a background point has a facial color.

Approach 2 - Difference between consecutive images

The second approach to extracting the face region assumes that the input images come from consecutive frames of a video file. It means that consecutive images are of very little difference. Also, it is assumed that the background objects do not have any large movement and only the head has movements between images. Due to the fact that only small movements of the head exist between images, the difference between two consecutive images will give roughly the face outline. So, by examining the difference values between images along each scanline, the face regions in the image can be identified. Figure 3-6 below shows the difference images between two pairs of consecutive images from a video file.

Figure 3-6: **Two sets of consecutive images (the two leftmost columns) and their difference images (the rightmost column)**

One limitation for this approach is that it works only if the images are the consecutive frames of a video file.

3.2 Features

In the last section, the facial regions are extracted from images. In this section, features need to be identified in the image. An approach to selecting features is described.

3.2.1 Why are feature points needed?

After images are extracted from the video file and the background objects are removed, points in different images need to be matched so as to combine the information in different images to reconstruct the three-dimensional model. But, before the points are matched, it is necessary to determine which points are going to be matched first. It is not a good idea to match all the points between images at one time. It is because some points can find their corresponding points accurately with a higher probability. So, it would be better to extract some feature points in each image first and then match the feature points found. Then, after matching the feature points, non-feature points can be matched with the use of the matching results of the feature points.

3.2.2 *What are feature points?*

When we talk about the features on the face, we will at the first glance think about the eyes, ears, nose and mouth. Is it a good idea to use the corners of the eyes or the corners of the mouth as features (see Figure 3-7) for two images to match between each other? On the other hand, can we choose the points that have special local image characteristics but not necessary being the tips of special organs?

Sometimes, image points that are not the extreme points of the major facial organs can also be appropriate points for performing matching between two images. If the lighting situations of two images do not differ for a large amount and there is no moving highlight in the images, the features selected according to local image characteristics from two images can contain more corresponding pairs than the number of extreme points of the facial organs. So, we have decided to extract the special points from the image according to local image characteristics.

***Figure 3-7*: Red points are the extreme points of facial organs.**

3.2.3 Feature points extraction

To extract the feature points (also called corners) according to the local image characteristics, we used an algorithm appeared on [h12, p.82].

The first step of the algorithm is to compute the gradient values along the x-axis and the y-axis at each point. The image color is denoted by E and the gradient value along the horizontal and the vertical direction are denoted by E_x and E_y respectively, where

$$E_x = \frac{\partial E}{\partial x} \quad \text{and} \quad E_y = \frac{\partial E}{\partial y}.$$

For each point in the image, its neighboring points (e.g. 5x5 pixels) are considered. A 2x2 matrix C is then formed using the gradient values of the points along the horizontal and the vertical directions within the neighborhood region.

$$C = \begin{bmatrix} \sum E_x{}^2 & \sum E_x E_y \\ \sum E_x E_y & \sum E_y{}^2 \end{bmatrix}$$

After a rotation of the coordinates axes, the matrix C can be transformed to a new matrix C'. Denote the eigenvalues associated with the matrix C' by λ_1 and λ_2. Then, for each eigenvalue λ, there is a corresponding eigenvector X satisfies the relation:

$$C'X = \lambda X,$$

$$C' = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, \text{where} \lambda_1, \lambda_2 \geq 0, \lambda_1 \geq \lambda_2$$

The two eigenvalues represent the strengths of the edges in the directions of the two eigenvectors. For a corner in the image, there should exist two edges (with edge strengths greater than a preset threshold) joining together. In other words, the eigenvalue λ_2 should be large enough at the corner. Thus, after the computations of the two eigenvalues for all the points in the image, the points are ranked according to λ_2. The points that have greater λ_2 are chosen.

One thing that needs to be taken care is that if a point has already been selected to be a corner, then its neighboring points should not be selected again. Figure 3-8 below shows the flow of the algorithm.

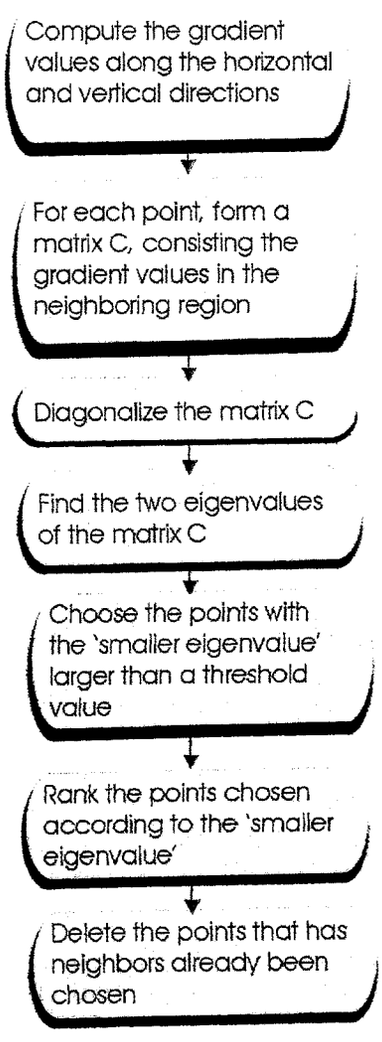

***Figure 3-8*: Steps for extracting feature points**

Laplacian values is that they are very sensitive to noise. So, the image needs to be pre-processed to get rid of noise if the Laplacian value is a component in the matching criteria function. Otherwise, false matching will be the consequence.

3.3.6 Laplacian of Neighbor points

Similar to the cases of the color value and gradient value, the fact that many points having more or less the same Laplacian values will lead to false matching. So, two blocks of the Laplacian values are matched instead of two Laplacian values. The benefit for doing this is just the same as that for the color value and the gradient value.

3.3.7 Size of neighboring region

Up to now, three components (the color value, the gradient value and the Laplacian value) are in the matching criteria function. It is also known that matching a block of color or gradient values can reduce the chance of having false matching. The next problem is to select a suitable neighborhood size. If the neighborhood size is too small, the effect of matching the color block or the gradient block will become insignificant. However, if the neighborhood size is too large, the time needs for computing the matching scores using the matching criteria function will be too long. Thus, the neighborhood size can neither be too small nor too large. The selection of the neighborhood size indeed requires experiments and experiences.

3.3.8 Combination of different matching components

From the previous subsections, it can be seen that the color value, the gradient value and the Laplacian value all have their distinguishing powers. But, which one has the best matching performance or what combination of these three kinds of values will yield the best matching result? This is an important problem that needs to be solved in order to have an accurate matching result. For some points, the matching of the color value can provide very accurate results. At the same time, for other points, the matching of the gradient value or the Laplacian value can locate their corresponding points more accurately. So, which one or which combination of matching components is the best depends on the points that are being matched. Furthermore, for a particular pair of images, a particular matching component or combination of matching components can provide a very high matching accuracy. But, if the same matching criteria function is applied to another pair of images, the matching accuracy may not be as high as before. Thus, which one or which combination of matching components is the best also depends on the property of the images that are being matched.

How to determine the weightings for different matching components?

If a point has a color value that many other points have, then the color value obviously cannot be the only matching criterion for this point. But, if a point has a color value

that is very different from the colors of other points, then the color value is certainly a suitable matching criterion for this point. Similarly, if a point has the gradient value that is very different from most of the other points, then the gradient value will be an obvious choice for the criterion used to match. Similarly, if a point has the Laplacian value that is very different from most of the other points, then the Laplacian value should take a large part in the matching criteria function. Hence, when performing matching for an image, the matching criteria function can vary from point to point and the matching criteria function at a point should depend on which matching component has a greater differentiating power at that point. The matching component that has a greater differentiating power at that point should have a greater weighting in the matching criteria function.

3.3.9 Procedures for obtaining the matching criteria function

Firstly, for each point in the image, it is necessary to collect the information related to the differentiating power for different matching components.
Assume that the point p is being matched.
The number of points having the color values that are different from p's color with an amount smaller than a preset threshold is counted. (Denote this value by *Nc*.)
The number of points having the gradient values that are different from p's gradient value with an amount smaller than a preset threshold is counted. (Denote this value by *Ng*.) Similarly, the number of points having the Laplacian values that are different from the p's Laplacian value with an amount smaller than a preset threshold is counted. (Denote this value by *Nl*.)

If *Nc* of a point is small, only a small number of points will be the candidates for its corresponding point according to the color value. Then, the weighting factor for the color component should be large because the color value has a greater differentiating power. Similarly, if *Ng* or *Nl* of a point is small, only a small number of points will be the candidates for its corresponding point according to the gradient or the Laplacian value. Then, the weighting factor for the gradient or the Laplacian component should be large because the value has a greater differentiating power.

Then, denote the matching criteria function of the point p by $Mp(x)$, where x is an arbitrary point on the image. From the formulae of the matching criteria function, it can be observed that the sum of the weightings of the three components is one. Component that has greater weighting will have greater influence to the matching score. For instances, if the point x has got a very unique color, then the weighting factor for $Diff\_c(x)$ will be large and the weighting factors for $Diff\_g(x)$ and $Diff\_l(x)$ will be small. Then, the matching score will mainly be determined by the value $Diff\_c(x)$.

$$M_p(x) = \frac{k}{Nc(p)} \cdot Diff\_c(x) + \frac{k}{Ng(p)} \cdot Diff\_g(x) + \frac{k}{Nl(p)} \cdot Diff\_l(x),$$

$$\text{where } k = \frac{Nc(p) \cdot Ng(p) \cdot Nl(p)}{Nc(p) \cdot Ng(p) + Ng(p) \cdot Nl(p) + Nl(p) \cdot Nc(p)},$$

Diff_c is the difference in color value between the point p and the point x,
Diff_g is the difference in gradient value between the point p and the point x,
Diff_l is the difference in Laplacian value between the point p and the point x.

After *Nc*, *Ng* and *Nl* have been counted, the matching criteria function of the point p can be formed. Then, the score for the point x on the other image can be computed. Points yielding lower scores mean that they have greater possibilities to be the corresponding points of the point p according to the computed matching criteria function. Figure 3-15 shows the process of locating corresponding points using the matching criteria function.

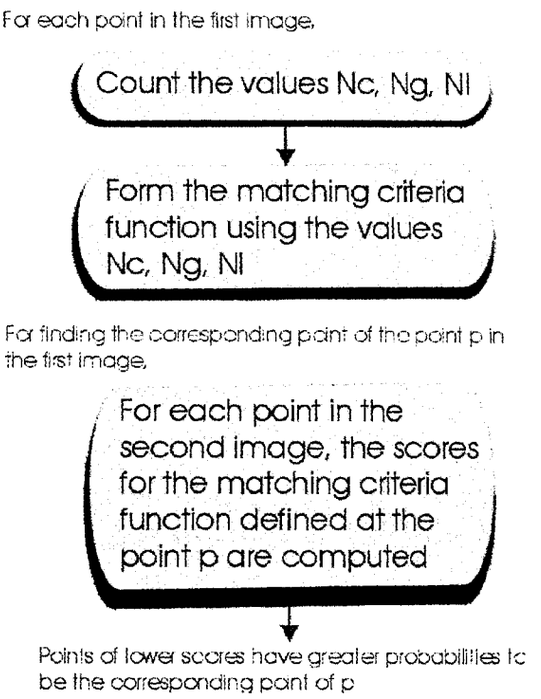

***Figure 3-15*: The process of finding corresponding points using the matching criteria function**

Speedup for the computation of the matching criteria function

In the process of forming a matching criteria function, the values *Nc*, *Ng* and *Nl* need to be counted. In other words, when counting the values *Nc*, *Ng*, *Nl* for a particular point p, the color values, the gradient values and the Laplacian values of all the other

3.4 Matching of feature points

After the investigation of the matching criteria function in the previous section, the matching process for the feature points is described in this section. Firstly, the feature points in the two images are found following the algorithm described in the subsection 3.2.2. Then, the feature points in the two images are being matched. The matching process makes use of the matching criteria function described in the previous section.

The matching process for a feature point begins with the computations of the weightings of different components in the matching criteria function. Then, the scores for the feature points in another image are computed using the matching criteria function and the point that gives the lowest score is selected to be the corresponding point.

Up to now, no positional information about the feature points is applied. In order to make the matching results as accurate as possible, a relative large neighboring region (10x10 pixels) is used. Figure 3-18 shows an example of the matching of the feature points between two images.

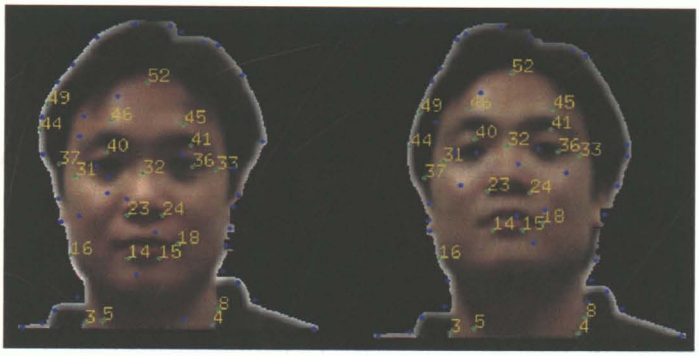

Figure 3-18: **The matching result for feature points**

However, no matter how large the neighboring region is considered, false matching is inevitable if no positional information is included in the matching process. Figure 3-19 shows the false matching due to the ignorance of the positional information.

Figure 3-19: **The matching of the feature point '25' is regarded to be wrong because the distances between '25' and '35', '48', '51' have great differences in the two images.**

Referring to Figure 3-19, the false matching of the point '25' can be avoided if it is known the point '25' is on the left of the point '31' and thus the corresponding point of the point '25' should also be on the left of the corresponding point of the point '31'. Another observation is that the relative distances between the feature points in the first image should be similar to the relative distances between the corresponding feature points in the second image. In Figure 3-19, the relative distances between the point '25' and '35', '48', '51' in the two images differ a lot. Thus, the matching of the point '25' can be detected as a wrong matching.

So, checking is required after the feature points are matched using the matching criteria function without considering the positional information. For each feature point, its distances between other feature points are computed. Then, after the matching process, the distances between its corresponding point and the corresponding points of other features are also computed. If some of the distances are very different from the original ones, the matching result for this point will have a high probability to be wrong. This checking is performed for all the feature points to identify any false matching. Figure 3-20 summarizes the steps for matching feature points.

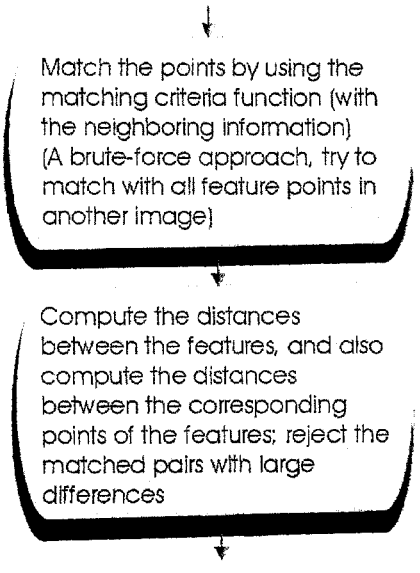

Figure 3-20: **The process of matching feature points**

In this section, the process for matching the feature points is described. In the next section, the steps for matching the non-feature points using the information of the matched feature points will be covered.

3.5 Matching of non-feature points

After the feature points are matched in the previous section, non-feature points are matched in this section. When the feature points are being matched, no positional information is considered. Checking is carried out after the matching to validate the matching results. However, for the matching of non-feature points, the positional information is considered in the matching process. The positional information comes from the positions of the matched feature points.

3.5.1 General steps for matching non-feature points

For matching non-feature points, the searching area for a corresponding point should not be the whole image. Instead, the searching area for a corresponding point should be bounded by the corresponding points of feature points or matched non-feature points. So, the first step for matching a non-feature point is to find its bounding points. After the bounding points for a non-feature point are found, the expected position of its corresponding point is estimated by simple computations.

Obviously, the estimated position of the corresponding point can hardly be an accurate one. So, the position of the corresponding point needs to be adjusted after the estimation. To achieve this, a local searching window centered at the estimated position is set up. The scores of the matching criteria function for the points in this local searching window are computed. (Note: The size of the local searching window depends on the average distance between the point and its bounding points. If its distances between the bounding points are greater, the size of the local searching window will be larger.) If the minimum score for the matching criteria function is not found at the center of the local searching window, it means that the estimated point is not the correct one. Then, the point having the minimum score will become the new estimated position. The process goes on until the point having the minimum score for the matching criteria function is found at the center of the local searching window. At that time, the estimated position will be the position of the corresponding point. Figure 3-21 summarizes the step for matching non-feature points.

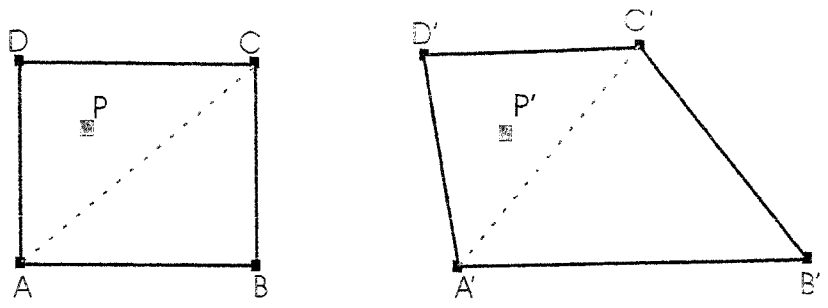

Figure 3-23: **Barycentric coordinates in a quad**

Referring to Figure 3-23, for a point P inside a quad $ABCD$, if the corresponding points for A, B, C, D are known to be A', B', C' and D' respectively, then the expected position of its corresponding point P' can be computed with the use of barycentric coordinates.

First, the barycentric coordinates (r_1, s_1, t_1) of P with respect to ΔABC are computed.

$$r_1 = \frac{S(\Delta PBC)}{S(\Delta ABC)},$$

$$s_1 = \frac{S(\Delta PCA)}{S(\Delta ABC)},$$

$$t_1 = \frac{S(\Delta PAB)}{S(\Delta ABC)}.$$

Then, the barycentric coordinates (r_2, s_2, t_2) of P with respect to ΔACD are also computed.

$$r_2 = \frac{S(\Delta PCD)}{S(\Delta ACD)},$$

$$s_2 = \frac{S(\Delta PDA)}{S(\Delta ACD)},$$

$$t_2 = \frac{S(\Delta PAC)}{S(\Delta ACD)}.$$

It is known that a point is inside a triangle if all the three values of the barycentric coordinates are greater than zero. So, if r_1, s_1 and t_1 are all greater than zero, the point P is inside ΔABC.

$$p' = \frac{S(\Delta PBC)}{S(\Delta ABC)}A' + \frac{S(\Delta PCA)}{S(\Delta ABC)}B' + \frac{S(\Delta PAB)}{S(\Delta ABC)}C'.$$

But, if r_2, s_2 and t_2 are all greater than zero, the point P is inside ΔACD.

$$p' = \frac{S(\Delta PCD)}{S(\Delta ACD)} A' + \frac{S(\Delta PDA)}{S(\Delta ACD)} C' + \frac{S(\Delta PAC)}{S(\Delta ACD)} D'.$$

But, if *s1* and *t2* equal to zero, it means that P lies on the line AC. In this case, the expected position of P' can be computed using either ΔABC or ΔACD. Figure 3-24 shows the computations of the expected positions of the corresponding points using the barycentric coordinates of the bounding points.

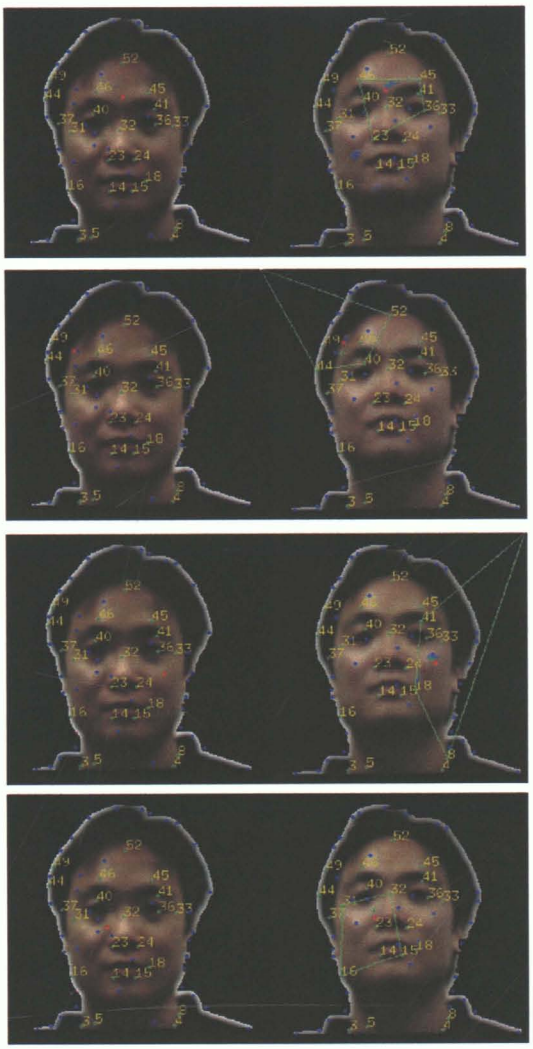

Figure 3-24: **The computation of the expected position of the corresponding point of a non-feature point by using the barycentric coordinates of the bounding points. The green quadrilaterals on the right images are formed by the bounding points of the red points on the left images. The green crosses on the right images are the computed expected positions of the corresponding points of the red points.**

Accuracy of the expected position

In this subsection and the previous subsection, the expected position of the corresponding point of a non-feature point is computed using the bounding quad and the bounding triangle respectively. It is obvious that if the bounding points are close to the original point and all these points are roughly lying on the same plane, then the computation of the expected position of the corresponding point will be more accurate. Otherwise, the computation of the expected position of the corresponding point is not accurate and further adjustment is needed. In the next subsection, the procedure for updating the bounding points is discussed.

3.5.4 Update of Bounding Points

Recall that in the subsection 3.5.1, the first step in the procedure of matching a non-feature point is to locate the bounding points. To do this, instead of searching all the matched points for the bounding points, the bounding points of all the unmatched points are updated (if necessary) whenever a new point has been matched.

Firstly, each point in the image is defined to have four bounding points, namely lower left bounding point, lower right bounding point, upper left bounding point and upper right bounding point. The bounding points of all the points in the image are initially set to be the image corners. Figure 3-25 shows the bounding points of a point.

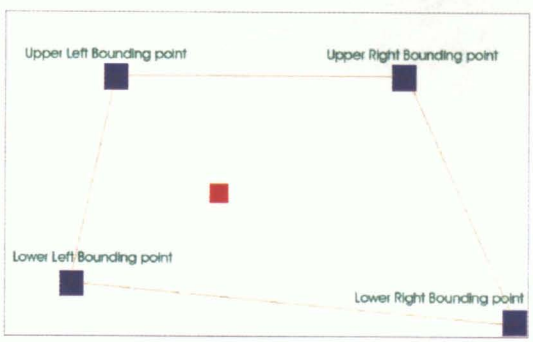

Figure 3-25: **The blue points are the bounding points of the red point.**

Then, after the matching of either a feature point or a non-feature point, the bounding points of the unmatched points are updated if necessary.

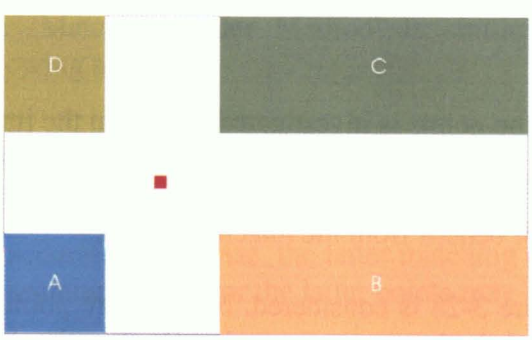

Figure 3-26: **Update of bounding points**

In Figure 3-26, the red point is the point that has just been matched. The bounding points of other unmatched points will be updated if necessary. For the points in the region *A*, if the distance from the red point is smaller than that from the existing upper right bounding point, then the red point will be the new upper right bounding point. For the points in the region *B*, if the distance from the red point is smaller than that from the existing upper left bounding point, then the red point will be the new upper left bounding point. For the points in the region *C*, if the distance from the red point is smaller than that from the existing lower left bounding point, then the red point will be the new lower left bounding point. Similarly, for the points in the region *D*, if the distance from the red point is smaller than that from the existing lower right bounding point, then the red point will be the new lower right bounding point.

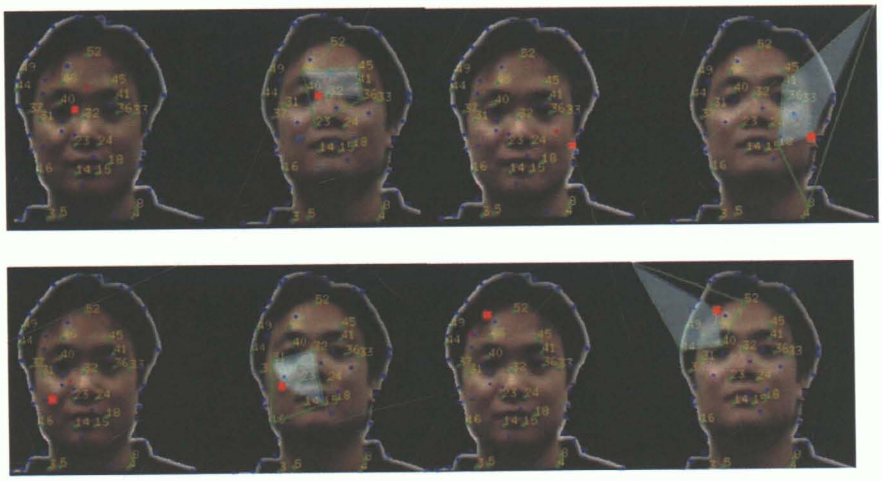

***Figure 3-27*: The orange points are the new matched points and the shaded blue quads are the new reduced searching areas for the small red points**

Figure 3-27 shows the examples for which the bounding points are updated when a matched pair of points is newly added. The original bounding areas (which are drawn by green lines) are reduced to the new bounding areas (the shaded blue areas) after a pair of matched orange points is inserted.

In this subsection, the procedure for updating the bounding points is described. Then, the matching order of the points will be investigated in the next subsection.

3.5.5 Matching order

In this subsection, the matching order of the points is investigated. Points in the image can be matched one by one following the row-major order or the column-major order. However, if the image is matched in row-major order or column-major order, the matching of the points cannot take the best benefit from the matched points.

In order to understand the situation, Figure 3-28 is considered. Non-feature points of the upper two images match with each other in the row-major order. On the other hand, non-feature points of the lower two images match with each other in the order that

points are matched sparsely and uniformly at the beginning. Then, the density of the matched points will increase gradually and uniformly.

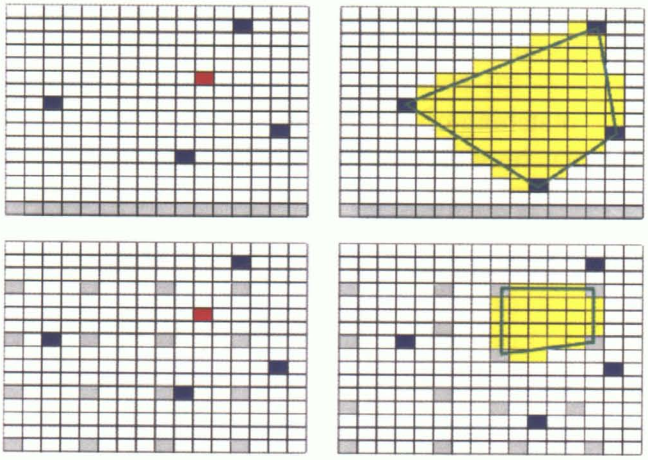

Figure 3-28: **The scenarios for the difference in the size of the search area for different matching order**

Referring to Figure 3-28, the pixels in blue represent the matched feature points. The pixels in gray represent the matched non-feature points. In both cases, 16 non-feature points have already been matched. The pixel in red is the non-feature point that is being matched at the current stage. The pixels in yellow are the ones that fall into the search area for the corresponding point. It can be observed that the search area is much smaller for the case that the points are matched from sparse to coarse uniformly. Let's call the matching method that follows this order be the 'Matching Order Method'.

In the next subsection, the 'Matching Order Method' and the multiresolution method are compared.

3.5.6 Comparison of the Matching Order Method and the multiresolution method

In this subsection, the 'Matching Order Method' is compared with the multiresolution method.

The 'Matching Order Method' is similar to the multiresolution method in the following sense:

- The main objective of these two methods is to reduce the computation time of the whole matching process by minimizing the searching area for a particular point.
- The matching results in the previous stages for these two methods need to be very accurate. Otherwise, the latter matching results will be greatly influenced since the searching areas for the latter points may have been modified wrongly.

Although the 'Matching Order Method' has some of the characteristics that the multiresolution method has, these two methods do have differences. The

multiresolution method relies on appropriate low-resolution image representations so that the matching result of the original resolution can be obtained from the matching results of the lower resolutions iteratively. For the 'Match Ordering Method', images of original resolution are used to perform the matching. It is more reliable and it does not depend on the existence of appropriate low-resolution image representations.

In this section, the issues about the matching of non-feature points are covered. In the next section, speedup issues of the matching process will be discussed.

3.6 Discussion - Speedup issues

In this section, techniques that are used for speeding up the matching process are discussed. When an image is matched with another image, all the points in the first image are trying to find their corresponding points in the second image. All the feature points go through the same process, from forming the matching criteria function to computing the matching scores for the points in another image. All the non-feature points go through another process, from locating the bounding points to local search at the computed expected position of the corresponding point. It is important to improve the performance of the algorithm so that the matching process can be done within a short time.

3.6.1 Consecutive Images from a video file

To improve the performance, a reasonable thinking direction is to reduce the search area for each point. If the input images come from a video file, the difference between consecutive image frames is small. Thus, the search area for a point in another image can be set to a small value. When two close images are being matched, points outside the small search area should not be selected as the corresponding points even if the scores obtained from the matching criteria functions are small enough.

3.6.2 Feature Tracking

This method can mainly be used for the images extracted from a video file. When two images that are not of consecutive pair from a video file are being matched, the difference can be large. Then, the search area for finding corresponding point will need to be large and also the matching result will be less reliable. To cope with this, one way to match two images of large difference is to match all the consecutive image pairs in-between the two original images. Then, the matching of the two original images can be obtained by traversing the in-between image pairs. In doing this, the number of matching performed will increase. But, for the matching of each individual point, the searching area can be much smaller because all the in-between matching is performed using two consecutive image frames. More importantly, the overall result of the matching will be more reliable since the matching between the consecutive frames is more reliable than the matching between two images of a large difference. Therefore, if the in-between images are available and the accuracy of the matching is the primary concern, this approach can be a suitable choice.

3.6.3 Pixel grouping

One way to speed up the matching process is to reduce the time for computing the matching criteria functions. The matching criteria function depends on the color value, the gradient value and the Laplacian value. For the points that have the same color

value, the same gradient value and the same Laplacian value, they should have the same matching criteria function and so this function needs to be computed once only. So, in order to avoid the re-computations of the same matching criteria functions, the pixels that have the same pixel characteristics should be grouped together. This pixel grouping process can improve the performance of the matching. The difficulty in applying this technique is that in real images, the pixels often have pixel characteristics of tiny differences but seldom have the identical pixel characteristics. This makes the pixel grouping not so practical.

3.6.4 Reduction of the searching area

Since after the computations of the matching criteria functions, the scores of the pixels within the search area need to be calculated. So, it is important to reduce the search area of a pixel as small as possible for speeding up the matching process.
There are two ways to reduce the searching area for a pixel.

One way to reduce the searching area for a pixel comes from an assumption that the difference between two images should not be large. With this assumption, all the corresponding points in the second image are not expected to be away from their original positions in the first image by a small threshold. For instances, if it is known that feature points and their corresponding points in another image differ at most by a value λ, the search area for a non-feature point can be set as $4 \times \lambda \times \lambda$. Obviously, a non-feature point will not be able to find its corresponding point correctly if its correct corresponding point is outside the search area. To be secure, a larger search area may still be needed but this will slow down the process. It needs a compromise.

The second way to reduce the searching area is to consider the positional relations between the points in the original image. Referring to Figure 3-29, consider five points in the first image. Denote these five points by *P1*, *P2*, *P3*, *P4*, *P5*. *P1* is on the left of *P5*, *P2* is on the right of *P5*, *P3* is below *P5* and *P4* is above *P5*. Assume that the corresponding points of *P1*, *P2*, *P3* and *P4* in the second image are already known and denote them by *Q1*, *Q2*, *Q3* and *Q4* respectively. Also, denote the unknown corresponding point of *P5* by *Q5*. Since *P1* is on the left of *P5*, *Q1* should also be on the left of *Q5*. Since *P2* is on the right of *P5*, *Q2* should also be on the right of *Q5*. Similar relations also hold for *Q3*, *Q5* and *Q4*, *Q5*. Therefore, the searching area of *P5* is bounded by the *x*-coordinates of *Q1*, *Q2* and the *y*-coordinates of *Q3*, *Q4*.

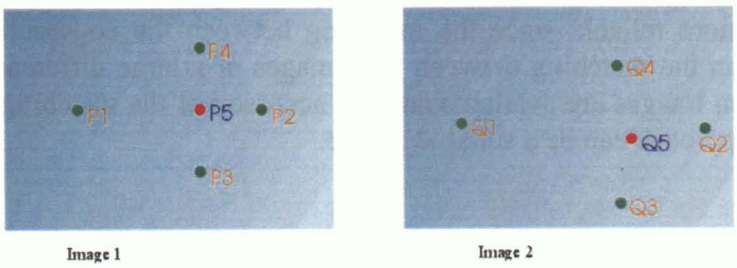

***Figure 3-29*: Locations of P1-5, Q1-5**

In general, as the corresponding points of more points are found, the searching areas of other points will be further reduced by the positions of the matched points. So, the matching order of the points will affect the time needed for matching all the points (as stated in the previous section). The search areas of the points are reduced in different rates for different matching orders of the points.

Shortcomings
One shortcoming for applying this approach to reduce the searching areas is that throughout the matching process, the matched pairs of points are assumed to be correct and used to reduce the searching area of other points in latter stages. If some errors exist in the matching, the searching area of other points will be modified wrongly and the consequence can be the correct corresponding point of a particular point has already been outside the search area. In other words, it is 'error accumulating'.

3.6.5 Multiresolution matching

Multiresolution matching is another method for speeding up the matching process. The main essence of multiresolution matching is to match the images of original resolution by first matching the images of lower resolutions.

How to form the coarse versions of the images?

Min-Max representation

This 'min-max' representation is used in [c6]. Four sub-images of lower resolution will be formed from one image of higher resolution. Each point in the image of lower resolution is formed from four points in the image of higher resolution.

Wavelets

Besides the "min-max" representation, the lower resolution images can also come from the wavelet representations. Basic knowledge about wavelets can be found in [c1, c4, c7, c8, c9, h11]. Figure 3-30 shows different resolutions of the image formed using Daubechies wavelets.

***Figure 3-30*: Different resolutions of the image formed using Daubechies wavelets**

Multiresolution Matching Process

The matching process starts from images of low resolution. From the matching results of the lower resolution images, the search areas of the points in the higher resolution images are restricted. After matching images of all the lower resolutions, the images of the original resolution will be matched based on the results of the lower resolution images. Figure 3-31 shows the flow for multiresolution matching.

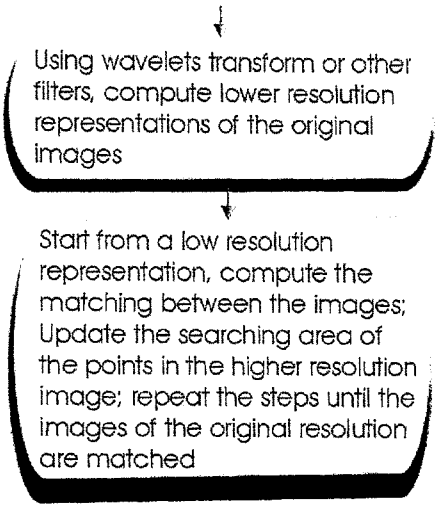

Figure 3-31: **The flow diagram for multiresolution matching**

Limitation

One of the disadvantage of the multiresolution matching is that if there are errors in the matching for the lower resolution images, the search area for the points in the higher resolution images will be wrongly set. Then, the corresponding points for the high-resolution images will be wrongly decided. In other words, the matching errors in the lower resolution will pass to the higher resolutions. Moreover, multiresolution matching depends on appropriate multiresolution image representations, only appropriate lower resolution image representations can provide accurate matching results for higher resolution images.

Indeed, the multiresolution matching will only be suitable to match the images that do not differ for a lot. It is because, if the images are of large difference, the low-resolution images will differ a lot from each other, then the matching in the low-resolution images will be unreliable. Consequently, the points in the high-resolution images can hardly find their corresponding points accurately. Thus, despite for the fact that the multiresolution matching can reduce the search areas of the points and hence can reduce the matching time, the reliability of the matching results may be affected by the quality of the lower resolution matching. It is again a tradeoff between the performance and the accuracy.

In this section, speedup issues about the matching process have been discussed. Besides speedup issues, accuracy issues are also important. Thus, accuracy issues about the matching process will be addressed in the next section.

3.7 Discussion - Accuracy Issues

In this section, several techniques are investigated for improving the accuracy of the matching process.

3.7.1 Color Perception

When two points in two images are being compared in term of their color values, one way is to compare the sum of the squared differences of their color components. (Assume the color is represented by the *R*, *G*, *B* values, the color differences of the two points will then be expressed in terms of their *R*, *G*, *B* values.) If the computed value is large, the two points are very different in terms of their colors. If the computed value is equal to zero or a very little value, the two points have a high probability to be a corresponding point pair if only the factor of color difference is considered. But, is it a fair approach to expressing the color differences in terms of the *R*, *G*, *B* values?

Color Model

When the colors of two points are compared, the *R*, *G*, *B* components of their colors are compared. Consider a situation, the color *(Ra, Ga, Ba)* of the point *A* in the first image is compared with the colors *(Rb, Gb, Bb)* and *(Rc, Gc, Bc)* of the points *B* and *C* in the second image. If the color difference values between the point *A* and the point *B* and between the point *A* and the point *C* are *D1* and *D2* respectively and *D1* equals *D2*, then the color differences between the point *A* and the point *B* is equal to that between the point *A* and the point *C* according to the computation based on the *RGB* values. However, a human may easily point out which point looks more like to be the corresponding point of the point *A* despite of the tie in the color difference computations. It means that the same amount of the *RGB* difference may not account for the same perceptual difference in the human eyes. It is due to the fact that the *RGB* color space is not a uniform color space. If it is intended to relate the difference in the color values to the perceptual difference in the human eyes, it is necessary to choose a color space that provides uniformly spaced human visual perception.

Besides *RGB* color model, other color models that follow the *CIE* standard (*CIE* stands for International Commission on Illumination) exist. *CIE* color models are not developed for displaying on any device and the main purpose is for measuring colors. *CIEXYZ* is the most basic *CIE* color model, but it is not a uniform color model. *CIELAB* and *CIELUV* are two uniform color *CIE* models. Thus, for computing the color difference that is intended to relate the visual perception, the *R*, *G*, *B* values are first converted to the *X*, *Y*, *Z* values of the *CIEXYZ* color model, and then further converted to the *L*, *A*, *B* values or the *L*, *U*, *V* values of the *CIELAB* color model and the *CIELUV* color model.

Literatures about color models and their uses can be found in [a5, g1, h2, h13]. In [a5], a uniform color system is applied for face detection. For comparing the matching

results of computing color difference using the *RGB* values and values in a uniform color system, a program downloaded from http://www.sys.wakayama-u.ac.jp/~chen/ucs.html has been used to convert the *R*, *G*, *B* values into the values of a uniform color system. Figure 3-32 shows the matching results of using values in a uniform color system to compute the color difference.

Points in blue on the right image are the points that the color differences (computed using values in a uniform color system) smaller than a preset threshold when compared with the red point on the left image.

Points in blue on the right image are the points that the color differences (computed using values in a uniform color system) smaller than a preset threshold when compared with the red point on the left image. (a neighboring block of 10x10 pixels considered)

Points in blue on the right image are the points that have the matching scores computed by the matching criteria function below a predefined threshold (with the color difference computed using values in a uniform color system).

Figure 3-34: **Pixel order preserving**

Referring to Figure 3-34, if it is known that *p2* lies between *p1* and *p3*, then their corresponding points should also be in the same order (i.e. *q2* should lie between *q1* and *q3*). That means, if it is known that *p2* lies between *p1* and *p3*, but *q2* does not lie between *q1* and *q3*, then there must be a matching error. Using this technique, matching errors can be detected whenever there is an ordering error when a straight line in the first image is matched on another straight line in the second image.

3.7.5 Quad preserving

The pixel order preserving described in the previous subsection only considers the order of the pixels in one dimension. It is also possible to consider the pixel order in two dimensions.

Referring to Figure 3-35, the four points (labeled *P1*, *P2*, *P3*, *P4*) in the first image are arranged in an anti-clockwise order. Then, in the second image, their corresponding points are labeled by *Q1*, *Q2*, *Q3* and *Q4* respectively. If the matching of the points is correct, *Q1*, *Q2*, *Q3* and *Q4* should also be in an anti-clockwise order. Thus, if after performing the matching process, the corresponding points *Q1*, *Q2*, *Q3* and *Q4* are not in an anti-clockwise order (e.g. the third image in Figure 3-35), there must be some matching errors. So, by checking the "quad preserving" characteristic, the correctness of the matching results can be checked.

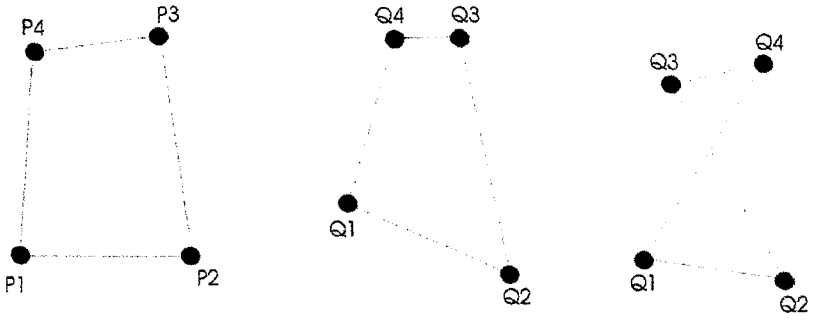

Figure 3-35: **The characteristics of "quad preserving". Points *Qi* are the corresponding points of points *Pi*. The second image satisfies "quad preserving" while the third image does not.**

Limitation

How can this checking be carried out? How to choose points to form quads in the image? Of course, it is not necessary to form quads for all the possible combinations

of four from all points and check the "quad preserving" characteristic. Instead, quads are formed from the grid points on the image. The separation between grid points can be set according to the time available for carrying out the test. If the grid points are denser, more quads are formed and more time is used for carrying out the tests. This again should be a tradeoff between accuracy and performance.

3.7.6 Quad area preserving

Besides the pixel order of the quads, the areas of the quads also provide some hints on the correctness of the matching. Given images from two different view points, the areas of the quads in the first image (formed by grid points) are clearly different from the areas of the quads in the second image (formed by the corresponding points of the grid points in the first image). But, the change of the areas of quads should not be very different from that of all of its neighboring quads. Therefore, if the change of the area of a quad is very different from that of its neighbors, there is a high possibility that there is a matching error. This property again can be used to detect the matching errors.

In this section, accuracy issues about the matching process have been discussed. In the next section, other issues will be addressed. These include discussions about the use of epipolar constraint and the matching of regions. Then, in the final section of this chapter, a summary will be given.

3.8 Discussion - Other Issues

In this section, two issues about the matching process will be discussed. One is about the use of epipolar constraint and another is about the matching of regions.

3.8.1 Use of epipolar constraint

The problem of image correspondence can also be solved with the help of the epipolar geometry [b3]. According to the epipolar constraint that holds for a stereo setup, a point lies on a line that passes through the epipole in the first image will lie on the corresponding line in the second image. This reduces the search region for the corresponding point from two dimensions to one dimension.
However, in some cases, the matching of the images cannot simply make use of the epipolar geometry. It is because the movement of the object between two images may be a non-rigid movement. For examples, consider the nodding action, the head may move up and down, but the neck will stay around its original position. Then, when the whole image is considered, the movement between images is not a rigid one. So, if the epipolar constraint needs to be applied, it is necessary to determine which part of the image follows a rigid movement and which part does not.

3.8.2 Matching of regions

When two images are being matched, points in the images can be grouped into regions and then the regions can be matched instead of the individual points. Matching the regions has an advantage over matching individual points that it is more efficient because there are often fewer regions than individual points.
However, in the case of matching faces, the method of matching regions cannot get much benefit since it is very difficult to group the points together to form regions due to the fact that most of the points have a similar color (i.e. the face color) and it is not easy to form distinctive regions of very different colors.

After describing the two issues in this section, a chapter summary will be given in the next section.

assumptions. In the next section, the assumptions made in the reconstruction approach will be stated.

4.2 Assumptions

The assumptions made in the reconstruction approach includes:

- The camera information is not used throughout the reconstruction process.
- The input images are formed with parallel projections.
- At least three images are provided as input images. One of them must be a frontal view image.
- The facial surface to be reconstructed is left-right symmetric.

In the next section, the grid points and gridlines used in the reconstruction part of the project will be defined.

4.3 Grid points and Gridlines

4.3.1 Definition of the gridlines

What are grid points?

Grid points are the points on the images between which are separated by a constant distance. This constant determines the density of the grid points. In our case, the grid points are defined with respect to the frontal view image

What are grid lines?

Two kinds of gridlines, the vertical gridlines and the horizontal gridlines, are defined. Vertical gridlines are formed by joining the grid points along the vertical direction on the frontal image. Similarly, horizontal gridlines are formed by joining the grid points along the horizontal direction on the frontal image. Figure 4-1 shows a frontal view image and the horizontal and vertical gridlines defined.

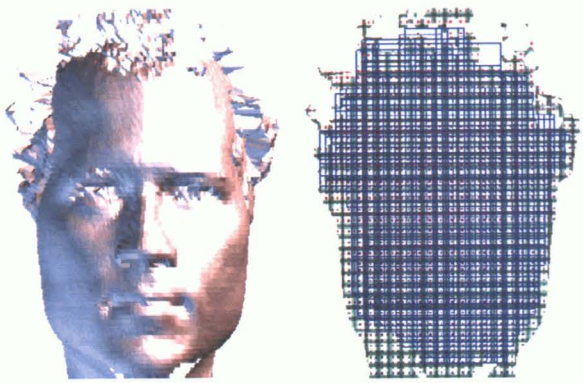

Figure 4-1: **Gridlines in the frontal view**

The grid points defined in the frontal view will be the point set for the reconstructed model. If the grid points are denser, the reconstructed model will also be denser. The x, y coordinates of points in the point set of the reconstructed model are set to the x, y coordinates of the grid points in the frontal view image. So, the task of the reconstruction is to find the z-coordinates of the points in the point set of the reconstructed model.

In the next section, the deformation of the gridlines from different views will be investigated.

4.4 Gridlines from different views

After defining the gridlines on the images in the last section, gridlines from different views will be studied in this section.

Planes

Obviously, horizontal gridlines and vertical gridlines on planes are still straight lines from different views (besides the frontal view). It implies that there is no change in the z-values of the grid points along the horizontal and the vertical gridlines.

Sphere

After considering the gridlines on planes from different views, the gridlines on a sphere are also studied. Figures 4-2 shows the horizontal gridlines and the vertical gridlines on a sphere from different views.

Facial surface

After observing the gridlines on the sphere from different views, the gridlines on a facial surface are studied. Figure 4-3 shows the horizontal gridlines and the vertical gridlines on a facial surface from different views.

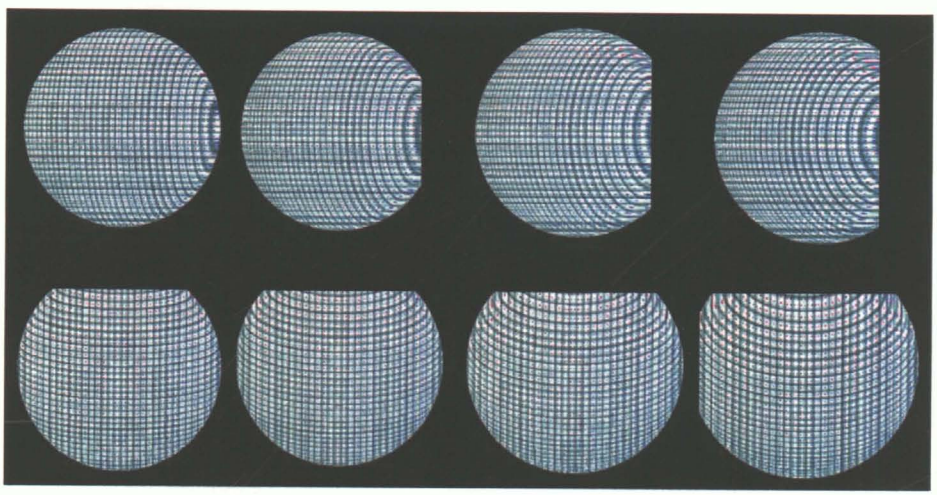

Figure 4-2: **Gridlines on the sphere from different views**

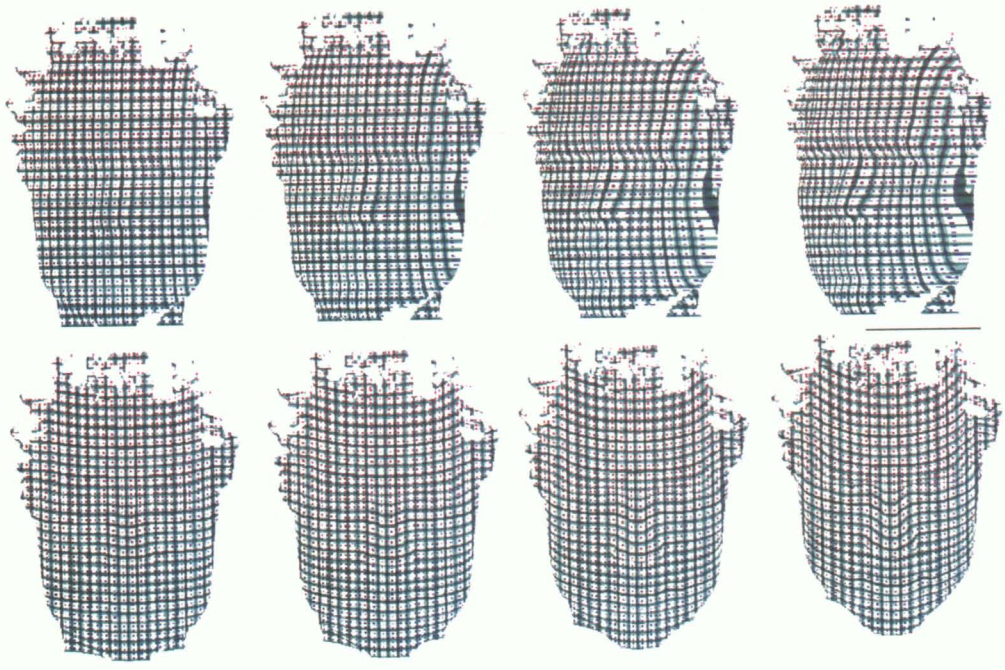

Figure 4-3: **Gridlines on the face from different views**

4.5 Depth Hints from the gridlines

In the last section, the grid points from different views are observed. Can any depth information obtained from the positions of grid points? Let's take a sphere as an example (Figure 4-4). Considering a side-view image, it can be observed that the vertical gridlines departs from straight lines. What can we talk about the depths of the point?

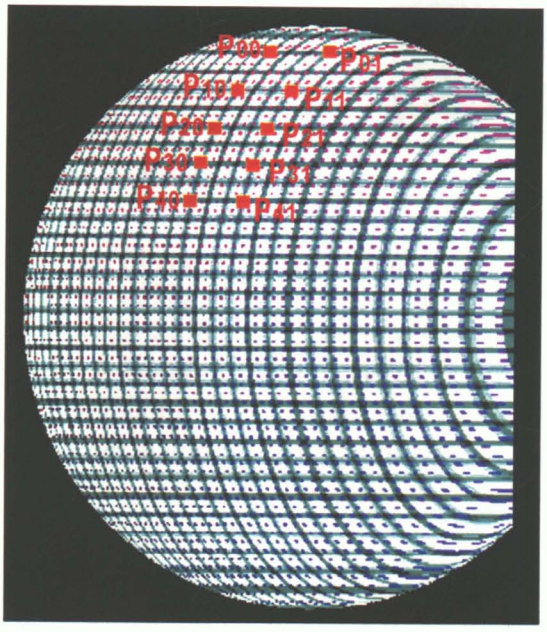

***Figure 4-4*: Grid points along the vertical gridlines**

Referring to Figure 4-4, from the x-coordinates of the grid points, P_{00} should have a smaller z-value than P_{10}, P_{10} should have a smaller z-value than P_{20}, P_{01} should have a smaller z-value than P_{11}, P_{11} should have a smaller z-value than P_{21}, etc.

However, since the image is not taken from exactly $90°$ from the side, the difference in x-coordinates between two grid points is not equal to the difference in z-values between two points. So, what obtained from x-coordinates of the grid points in a side-view image is the z-order of the grid points along their vertical gridlines. However, the information about the z-order of the grid points between different vertical gridlines or along horizontal gridlines cannot be obtained from a side-view image. Therefore, another image from the top front or the bottom front is required.

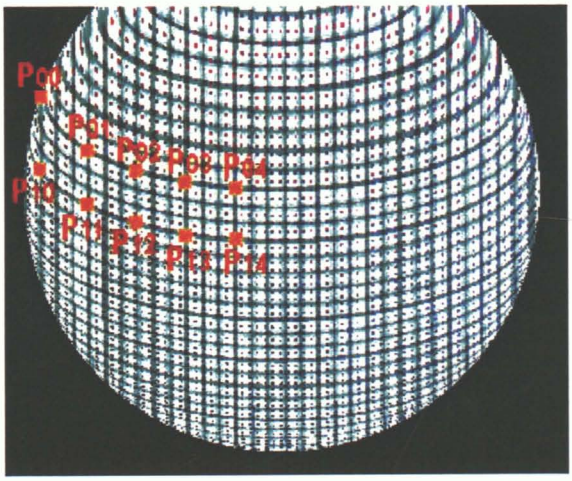

Figure 4-5: **Grid points along the horizontal gridlines**

Figure 4-5 shows the positions of the grid points from a top front view. It can be observed that P_{00} should have a smaller z-value than P_{01}, P_{01} should have a smaller z-value than P_{02}, P_{10} should have a smaller z-value than P_{11}, P_{11} should have a smaller z-value than P_{12}, etc. But, since the image is not taken exactly from the 'top', the difference in y-coordinates between two points is not equal to the difference in z-values between two points. So, what obtained from the y-coordinates of the grid points in a top front view is the z-order of the grid points along their horizontal gridlines.

Therefore, if a side view image as well as a top front view or a bottom front view image are available, the z-order of the grid points along the vertical gridlines and the horizontal gridlines can be obtained. So, the task is to obtain exact difference in the z-values from the z-order of the points. In order to facilitate this, two values called "point-proportions" are defined at each grid point in the next section.

4.6 Point proportions

In the previous section, the z-order of the grid points along the horizontal and the vertical gridlines can be obtained from the side view image and the top front image. In this section, two values called point proportions are defined at each point so as to facilitate the discussions.

4.6.1 Definition of Point proportions

Instead of applying the x-coordinates of the grid points in the side view image or the y-coordinates in the top front view image directly, two point proportions (horizontal point proportion and vertical point proportion) are defined at each grid point.

What is point proportion? Referring to Figure 4-6 and let's consider the diagram for the vertical gridline. Assume (without loss of generality) that the side view is from the right. Denote the rightmost point of the vertical gridline by V_{min}. Also, denote the leftmost point of the vertical gridline by V_{max}. If a grid point P has a horizontal distance of V from V_{min}, then the point proportion of P along this vertical gridline is

$$\frac{V - V_{min}}{V_{max} - V_{min}}$$

The situation is similar for the case of a side image from the left.

Next, the diagram for top front view is considered. Denote the bottommost point of the horizontal gridline by H_{min}. Also, denote the topmost point of the horizontal gridline by H_{max}. If a grid point P has a vertical distance of H from H_{min}, then the point proportion of P along this horizontal gridline is

$$\frac{H - H_{min}}{H_{max} - H_{min}}$$

The situation is similar for the case of a bottom front view image. It should be noted that both vertical and horizontal point proportions fall into the range of *0-1*.

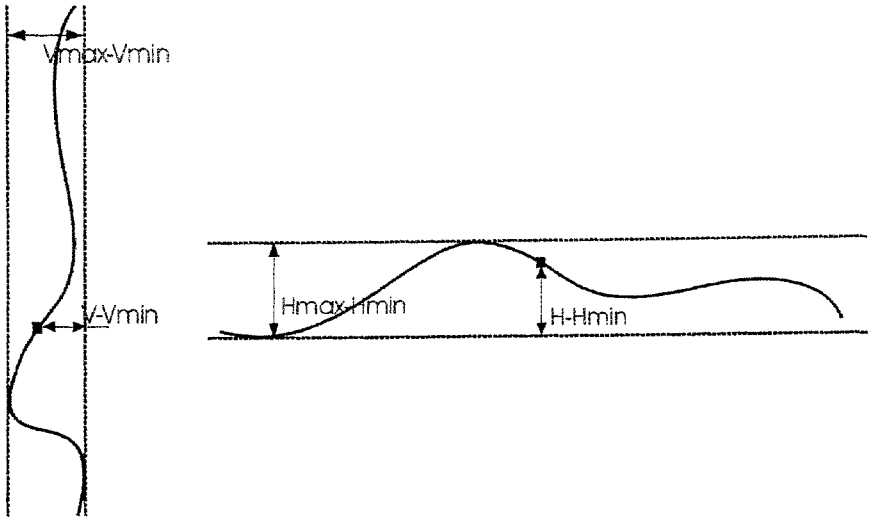

Figure 4-6: **Point proportions**

Since each visible grid point in the image lies on one vertical gridline and one horizontal gridline, each point has one vertical point proportion and one horizontal point proportion.

4.6.2 Invariance information of different views

Since the images used for reconstruction are assumed to be formed with parallel projection, the point proportions obtained from views of different view angles should be the same and the difference between the point proportions should reflect the difference between the z-values. In other words, the points should have the same horizontal and vertical point proportions when the point proportions are computed using the z-values instead of the pixel coordinates.

Despite of the fact that the point proportions are not affected by the view angle, the number of visible grid points will be affected by the view angle. Point proportions can be obtained only from the visible grid points. Besides, if the side view image used is too close to the frontal view image, point proportions obtained will be unreliable. In extreme situation, V_{max}-V_{min} may equal to *1* and V-V_{min} of the points equal to either *0* or *1*. Then, the point proportions of these points will be unreliable and useless.

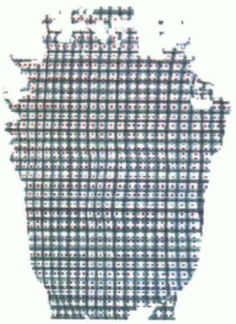

Figure 4-7: **A side view image with too small view angle**

In Figure 4-7, since the angle that the side view image makes with the frontal image is too small, the vertical point proportions obtained are not reliable and cannot used for reconstructing a facial surface.

4.6.3 Left-Right Symmetric

In this project, the reconstructed facial surface is assumed to be left-right symmetric. (It is reasonable since most of the real facial surfaces are approximately left-right symmetric.) For processing convenience, either a side view from the left or a side view from the right is used. Assume that a side view (Figure 4-8) of the object from the right is given. Considering the vertical gridlines, only the ones from the right are usable because of the occlusion due to the nose. Then, the point proportions of the invisible grid points are set to be the same as their counterparts on the right.

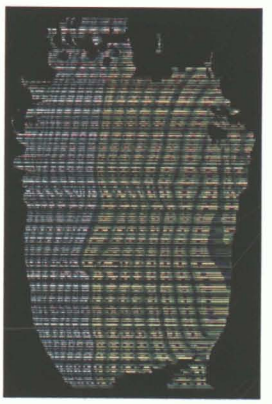

Figure 4-8: **Gridlines on the right hand side of the central vertical gridline are sufficient for reconstructing a facial surface if the facial surface is assumed to be left-right symmetric**

In this next section, the relationship between the point proportions is studied.

4.7 Relationships between gridlines

In this section, relationship between gridlines is investigated. The point proportions of grid points on the vertical and horizontal gridlines are the information retrieved from the input images

4.7.1 Relationships between the gridlines (part 1)

If there are N visible grid points appeared in the frontal image, the side view image and also the top or bottom frontal image, then the reconstructed model will have N grid points. The x, y coordinates of the points on the reconstructed model are equal to the x, y coordinates of the grid points on the frontal view image. The z coordinates of these N grid points are the values that need to be found.

Denote the point proportion of the grid point *(j, i)* along its vertical gridline by $v(i,j)$.
Denote the point proportion of the grid point *(j, i)* along its horizontal gridline by $h(i,j)$.
Denote the function of z-value along the j^{th} vertical gridline by P_j, and denote the z-value of the i^{th} point on it by $P_j(i)$.
Denote the function of z-value along the i^{th} horizontal gridline by Q_i, and denote the z-value of the j^{th} point on it by $Q_i(j)$.

By equating the z-value at the grid point *(j, i)*, $P_j(i)$ equals to $Q_i(j)$. Since the input images are formed using parallel projection, the z-value of a grid point on a vertical gridline should be equal to a constant plus the point proportion multiplied by another constant. The idea is shown in Figure 4-9.

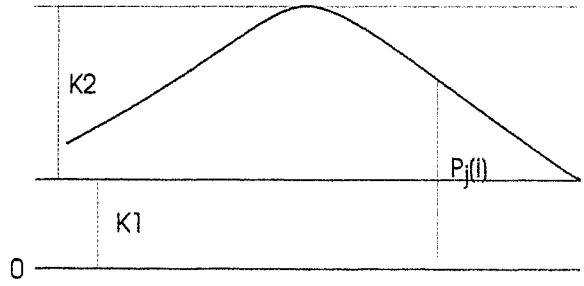

***Figure 4-9*: Relationships between $P_j(i)$, k1, k2**

$P_j(i) = k1_j + k2_j \cdot v(i, j)$, where $k1_j$ and $k2_j$ are unknowns

Since the input images are formed using parallel projection, the z-value of a grid point on a horizontal gridline should be equal to a constant plus the point proportion multiplied by another constant.

$Q_i(j) = h1_i + h2_i \cdot h(i, j)$, where $h1_i$ and $h2_i$ are unknowns

Since $P_j(i) = Q_i(j)$,
$k1_j + k2_j \cdot v(i, j) = h1_i + h2_i \cdot h(i, j)$
So, for each grid point (j, i), an equation of the same form can be formed as the one appeared on the last line.

$$\begin{pmatrix} 1 & v(0,0) & \dots & \dots & \dots & -1 & -h(0,0) & \dots & \dots & \dots \\ 1 & v(1,0) & \dots & \dots & \dots & \dots & \dots & -1 & -h(1,0) & \dots \\ \vdots & & & & & & & & & \end{pmatrix} \cdot \begin{pmatrix} k1_0 \\ k2_0 \\ \vdots \\ h1_0 \\ h2_0 \\ \vdots \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ \vdots \end{pmatrix}$$

An equation is only formed if the grid point is visible in the images. So, if there are m vertical gridlines and n horizontal gridlines, then the number of equations in the above system should be less than $m \times n$ and the number of variables is less than or equal to $2 \times (m + n)$.

Then, the task of finding the z-values of the grid points becomes the task to solve the system of equations to find the unknowns $k1_0, k2_0, \dots, h1_0, h2_0, \dots$. If the unknowns h, k can be found, the z-values of the grid points can be computed.

4.7.2 Relationships between the gridlines (part 2)

However, the variables $k1$, $k2$, $h1$, $h2$ in the previous system of equations are not independent, there exists other relationship between them.

Consider four neighboring grid points at (j, i), $(j+1, i)$, $(j, i+1)$, $(j+1, i+1)$ in Figure 4-10.

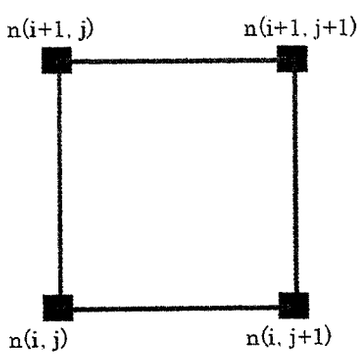

***Figure 4-10*: Four neighboring grid points**

Along the i^{th} horizontal gridline,

$Q_i(j+1) - Q_i(j) = h1_i + h2_i \cdot h(i, j+1) - h1_i - h2_i \cdot h(i,j)$
$Q_i(j+1) - Q_i(j) = h2_i \cdot (h(i, j+1) - h(i, j))$

Similarly, along the $i+1^{th}$ horizontal gridline,

$Q_{i+1}(j+1) - Q_{i+1}(j) = h1_{i+1} + h2_{i+1} \cdot h(i+1, j+1) - h1_{i+1} - h2_{i+1} \cdot h(i+1,j)$
$Q_{i+1}(j+1) - Q_{i+1}(j) = h2_{i+1} \cdot (h(i+1, j+1) - h(i+1, j))$

Along the j^{th} vertical gridline,

$P_j(i+1) - P_j(i) = k1_j + k2_j \cdot v(i+1, j) - k1_j - k2_j \cdot v(i, j)$
$P_j(i+1) - P_j(i) = k2_j \cdot (v(i+1, j) - v(i, j))$

Similarly, along the $j+1^{th}$ vertical gridline,

$P_{j+1}(i+1) - P_{j+1}(i) = k1_{j+1} + k2_{j+1} \cdot v(i+1, j+1) - k1_{j+1} - k2_{j+1} \cdot v(i, j+1)$
$P_{j+1}(i+1) - P_{j+1}(i) = k2_{j+1} \cdot (v(i+1, j+1) - v(i, j+1))$

Denote the z-value at the grid point (j, i) by $n(i, j)$.
$n(i, j) = P_j(i) = Q_i(j)$

Considering the grid points (j, i), $(j+1, i+1)$,
$n(i+1, j+1) - n(i, j) = n(i+1, j+1) - n(i+1, j) + n(i+1, j) - n(i, j)$
$n(i+1, j+1) - n(i, j) = h2_{i+1} \cdot (h(i+1, j+1) - h(i+1, j)) + k2_j \cdot (v(i+1, j) - v(i,j))$

$n(i+1, j+1) - n(i, j) = n(i+1, j+1) - n(i, j+1) + n(i, j+1) - n(i, j)$
$n(i+1, j+1) - n(i, j) = k2_{j+1} \cdot (v(i+1, j+1) - v(i, j+1)) + h2_i \cdot (h(i, j+1) - h(i, j))$

$\therefore h2_{i+1} \cdot (h(i+1, j+1) - h(i+1, j)) + k2_j \cdot (v(i+1, j) - v(i,j)) = k2_{j+1} \cdot (v(i+1, j+1) - v(i, j+1))$
$+ h2_i \cdot (h(i, j+1) - h(i, j))$

Now, considering the grid points $(j+1, i)$, $(j, i+1)$,
$n(i, j+1) - n(i+1, j) = n(i, j+1) - n(i, j) + n(i, j) - n(i+1, j)$
$n(i, j+1) - n(i+1, j) = h2_i \cdot (h(i, j+1) - h(i, j)) - k2_j \cdot (v(i+1, j) - v(i, j))$

$n(i, j+1) - n(i+1, j) = n(i, j+1) - n(i+1, j+1) + n(i+1, j+1) - n(i+1, j)$
$n(i, j+1) - n(i+1, j) = -k2_{j+1} \cdot (v(i+1, j+1) - v(i, j+1)) + h2_{i+1} \cdot (h(i+1, j+1) - h(i+1, j))$

$\therefore h2_i \cdot (h(i, j+1) - h(i, j)) - k2_j \cdot (v(i+1, j) - v(i, j)) = -k2_{j+1} \cdot (v(i+1, j+1) - v(i, j+1)) +$
$h2_{i+1} \cdot (h(i+1, j+1) - h(i+1, j))$

From the above two cases, it can be seen that an algebraic relationship holds between the unknowns $h2_i$ and $k2_j$ for every four neighboring grid points.

After reordering, the algebraic relationship looks like this:
$(v(i+1, j) - v(i, j)) \cdot \mathbf{k2_j} - (v(i+1, j+1) - v(i, j+1)) \cdot \mathbf{k2_{j+1}} - (h(i, j+1) - h(i, j)) \cdot \mathbf{h2_i} + (h(i+1,$
$j+1) - h(i+1, j)) \cdot \mathbf{h2_{i+1}} = 0$

By listing out the algebraic relationships, a system of equations can be formed.

$$\begin{pmatrix} v(1,0)-v(0,0) & -v(1,1)+v(0,1) & \cdots & -h(0,1)+h(0,0) & h(1,1)-h(1,0) & \cdots \\ & & & & & \end{pmatrix} \cdot \begin{pmatrix} k2_0 \\ k2_1 \\ \vdots \\ h2_0 \\ h2_1 \\ \vdots \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ \vdots \\ \vdots \end{pmatrix}$$

Therefore, besides the system of equations is formed by relating the unknowns *k1*, *k2*, *h1*, *h2*, a new system of equations involving the unknowns *k2*, *h2* can be obtained.

For obtaining the *z*-values of the grid points, the unknowns *k1*, *k2*, *h1*, *h2* require to be solved. The latter system of equations involving only *k2*, *h2* can be solved first. Then, the result can be substituted into the first system of equations to solve the unknowns *k1*, *h1*.

However, if some of the equations are wrongly formed, then the solution for the whole system of equations will not be reliable. The quality of the reconstructed model will not be good. Therefore, another approach that involving further assumption to tackling the problem is proposed in the next section.

4.8 Problem Simplification

In this section, a simplification under an assumption is proposed instead of solving the systems of equations derived in the last section.

4.8.1 Further Assumption

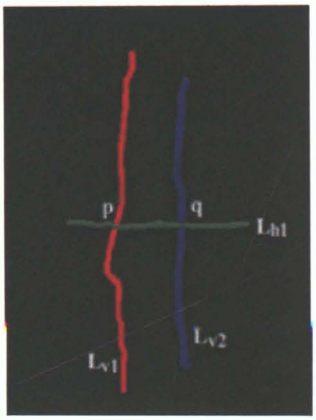

***Figure 4-11*: Show the configuration**

Referring to Figure 4-11, L_{v1} is the central vertical gridline (central with respect to the frontal view). L_{v2} is another vertical gridline. L_{h1} is the horizontal gridline that cross L_{v1} and L_{v2} at the point p and q respectively. Assume the scaling factor of L_{v1} be *Scale1*. Then, the objective in this section is to find scaling factors of other vertical gridlines in terms of Scale1. Denote the minimum and the maximum z-value of the horizontal gridline L_{h1} by a_{min} and a_{max} respectively. Denote the minimum and the maximum z-value of the vertical gridline L_{v1} by p_{min} and p_{max} respectively. Denote the minimum and the maximum z-value of the vertical gridline L_{v2} by q_{min} and q_{max} respectively.
Although the point proportions are defined using the pixel coordinates in section 4.6, they also apply to the z-values of the points since the input images are formed with parallel projection. In the following equations which involve the point proportions, z-values are used.

Since p lies on the vertical gridline L_{v1},

$$p_{.z} = v(p) \cdot Scale1 + p_{min}$$

Since p and q lie on the same horizontal gridline L_{h1},

$$\frac{p_{.z} - a_{min}}{a_{max} - a_{min}} = h2(p),$$

$$\frac{q_{.z} - a_{min}}{a_{max} - a_{min}} = h2(q),$$

$$\frac{p_{.z} - a_{min}}{q_{.z} - a_{min}} = \frac{h2(p)}{h2(q)},$$

$$p_{.z} \cdot h2(q) - a_{min} \cdot h2(q) = q_{.z} \cdot h2(p) - a_{min} \cdot h2(p),$$

$$q_{.z} = \frac{p_{.z} \cdot h2(q) + (h2(p) - h2(q)) \cdot a_{min}}{h2(p)}. \tag{4.1}$$

Since q lies on the vertical gridline L_{v2},

$$\frac{q_{.z} - q_{min}}{q_{max} - q_{min}} = v2(q),$$

$$q_{.z} - q_{min} = v2(q) \cdot q_{max} - v2(q) \cdot q_{min},$$

$$q_{max} = \frac{q_{.z} + (v2(q) - 1) \cdot q_{min}}{v2(q)}. \tag{4.2}$$

Assumption

From above mathematical expressions, $q_{.z}$ and q_{max} depends on the values a_{min} and q_{min}. In order to get rid of these two unknowns and make $q_{.z}$ and q_{max} only depends on p_{min} (the value associated with the central vertical gridline, an further assumption is made:
$p_{min} = q_{min} = a_{min}$

This assumption means that the minimum points of the z-values for all the vertical gridlines and horizontal gridlines are of the same value. If the surface to be reconstructed is a cylinder, then this assumption is invalid because the vertical gridlines on the cylinder will not have the same minimum z-value. However, if the surface to be reconstructed is a cone or a sphere, then this assumption is valid since all minimum z-values for all the vertical and horizontal gridlines are of the same value.

For the case of a human face, the assumption is roughly valid. It can be observed that most of the vertical and horizontal gridlines have their minimum points near the boundary

of the face and also points on the boundary of the face are roughly of the same z-values. In other words, the facial surface to be reconstructed has roughly the shape of ellipsoid. Thus, this assumption is taken to simplify the computation in the reconstruction process. Since the reconstructed surface does not change with the translation along the z-axis, p_{min} can be assumed to be an arbitrary value. In our implementation, p_{min} is set to be *500*. Based on the values *Scale1* and p_{min}, z-values of grid points along the central vertical gridlines can be computed.
For other vertical gridlines, one point q is chosen on the vertical gridline and then p is located along the same horizontal gridline. (The condition for the points q and p will be stated later.)

Using the assumption that a_{min} equals to p_{min}, the equation 4.1 becomes

$$q_{.z} = \frac{p_{.z} \cdot h2(q) + (h2(p) - h2(q)) \cdot p_{min}}{h2(p)} \tag{4.3}$$

After q has been calculated and the assumption that q_{min} equals to p_{min} is used, the equation 4.2 becomes

$$q_{max} = \frac{q_{.z} + (v2(q) - 1) \cdot p_{min}}{v2(q)} \tag{4.4}$$

Then, the z-value of any grid point q_a along the q's vertical gridline can be computed.

$$q_{a.z} = q_{min} + v2(q_a) \cdot (q_{max} - q_{min})$$

This process is repeated for every vertical gridline. Consequently, the z-values of all the grid points are computed.

4.8.2 How to choose q and p?

Consider the equation 4.3, *h2(p)* cannot be equal to zero. So, p cannot be the minimum point along its horizontal gridline.
Consider the equation 4.4, *v2(q)* cannot be equal to zero. So, q cannot be the minimum point along its vertical gridline.

4.8.3 How to find the value 'Scale1'?

Up to now, what has been obtained is a point set of the reconstructed model that the z-coordinates are up to a scale. When the z-value of p is computed, there is an unknown *'Scale1'* that has not been decided. This value is indeed the difference between the minimum and the maximum z-values of the central vertical gridline. At this stage, this value is changed interactively. It will be a challenging future work to obtain the value of

'Scale1' without any manual intervention. Figure 4-12 shows the flow of the reconstruction process with the use of simplification.

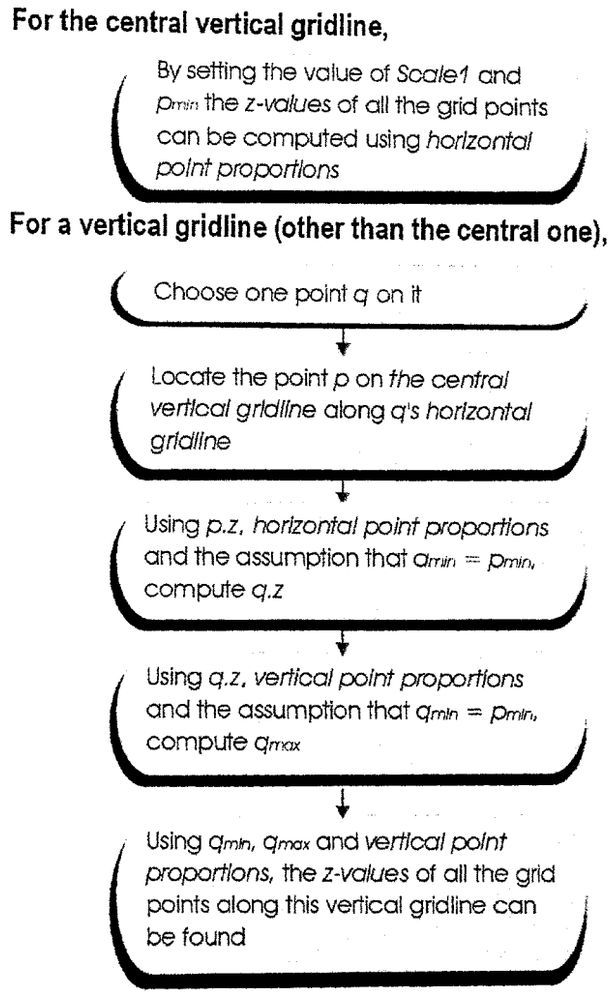

Figure 4-12: **The flow of the reconstruction process with the use of simplification**

4.8.4 Triangulation

After the z-values of all the grid points are computed, a *3D* model is rendered followed by triangulation. In the implementation, the triangulation is completed by using a library downloaded from the Internet, called 'qhull'. The URL for this library is http://www.geom.umn.edu/software/qhull/.

In the next subsection, the result of the reconstruction is shown.

4.9 Result

Figure 4-13 shows the facial surfaces reconstructed using different value of *'Scale1'*.

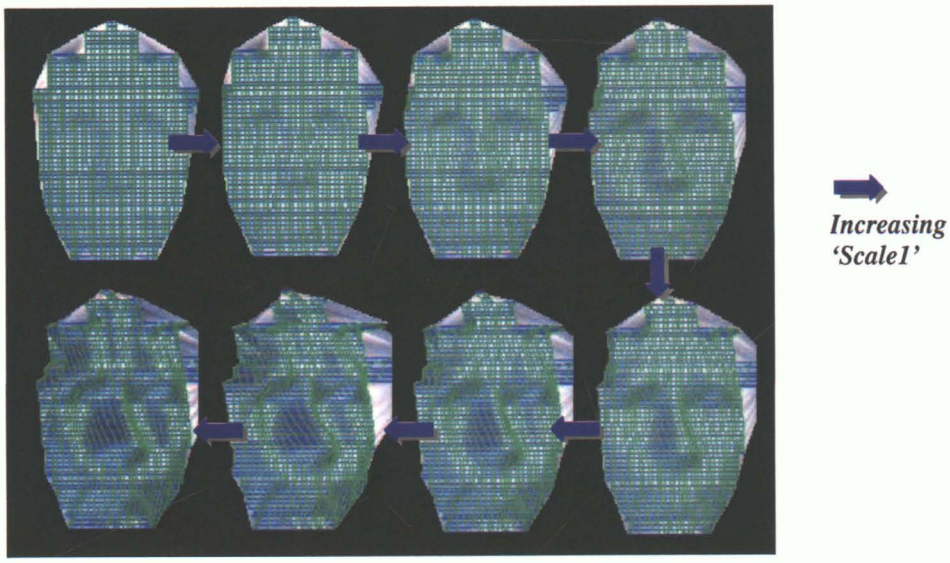

Figure 4-13: **Facial surfaces reconstructed for different values of 'Scale1'**

It can be observed that the surface reconstructed looks like a plane surface if *'Scale1'* is too small. Also, if *'Scale1'* is too large, the reconstructed surface does not look like a human facial surface. So, the value of *'Scale1'* needs to be set carefully.

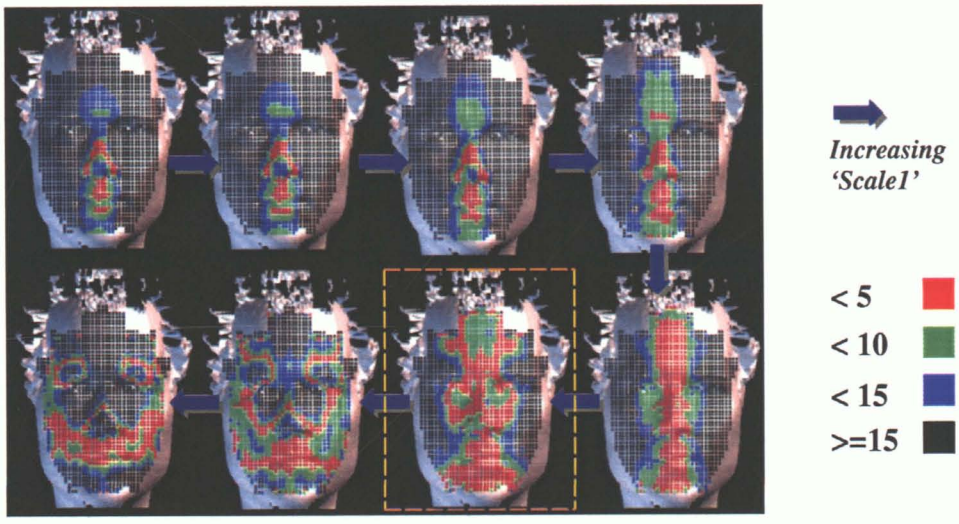

Figure 4-14: **Errors in facial surfaces reconstructed for different values of 'Scale1' (the *z* range of the original model is about *500*) (The one in the yellow dashed box has the minimum Hausdorff distance with the original model)**

4.10 Pros and Cons

4.10.1 Advantages

- The computation involved is simple.
- No special equipment is used in this reconstruction approach. Also, no generic model is needed.
- Only the corresponding information between the images is used.

4.10.2 Limitations

- If some parts of the surface do not appear in the frontal view, those parts will not appear in the final reconstructed model.
- The value *'Scale1'* needs to be set interactively.

4.11 Summary

Figure 4-17 shows the flow for the reconstruction process of a facial surface from images.

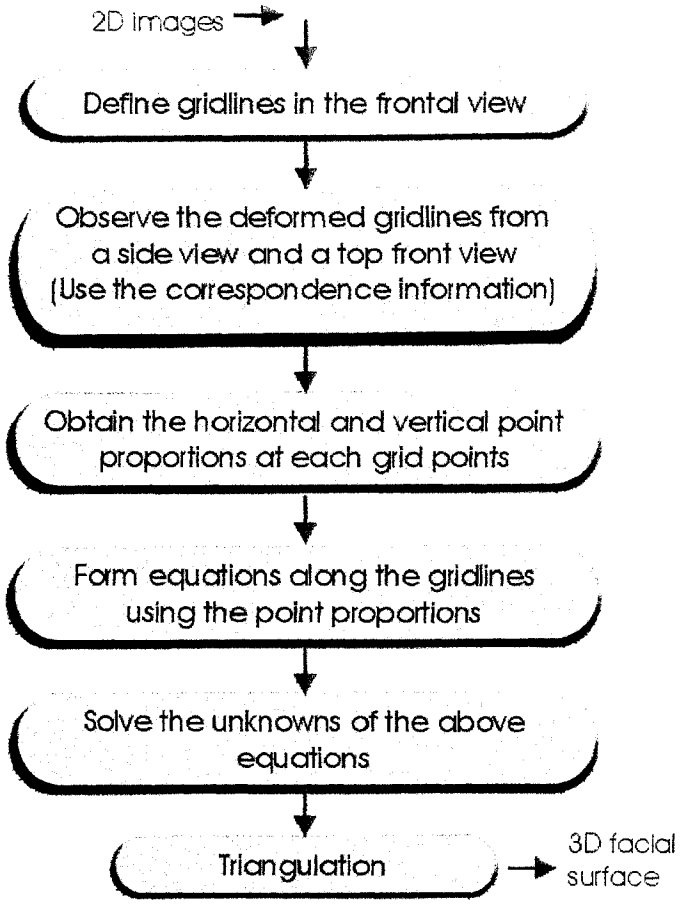

Figure 4-17: **Steps for reconstructing a 3D surface from 2D images**

In this chapter, an approach to reconstructing a facial surface has been described. The changes of positions of the grid points from different views are observed since the image correspondence between images is available. A facial surface can be reconstructed with simple computations.

Chapter 5. Conclusion

In this thesis, the steps that we have followed throughout the project are described. This project is divided into two parts (the feature matching part and the reconstruction part).

In the chapter for the 'Related Work', the previous related work about the feature matching part is briefly described. Also, several approaches to modeling facial surfaces are mentioned.

In the 'Image Correspondence' chapter, the process has been gone through in the project is described. In the feature matching part, the thing that differs most from other previous work is the formation of the matching criteria function. The weighting of a component in the matching criteria function is determined by its uniqueness with respect to the matching point. Moreover, speedup and accuracy issues have been addressed in that chapter.

For the reconstruction part, we have decided not to use a laser range finder, not to use a generic model and also not to estimate the camera parameters from the images. Instead, only the correspondence information is used to reconstruct facial surfaces. The changes of the positions of the grid points in different views are used to compute z-values of the grid points. Although the quality of the output at this stage still has much room for improvement, we believe that our reconstruction approach is a way that can have much extension to reconstruct a facial surface only from the correspondence information between images.

Chapter 6. Future Work

There are still some aspects in this project that are worth to be future extensions.

For the image matching part, it is believed that the use of epipolar constraint should be a method to improve the performance. The problem is that it works only when the movement between two images is a rigid one. If the movement is a non-rigid one (such as the movement of neck or some facial expressions), then the epipolar constraint will not apply. But, there should be still some regions on the image that can be assumed to be following a rigid movement and these regions should be able to make use of the epipolar constraint. So, it is worth to investigate how to identify the regions of an image that follow rigid movement and the regions of an image that does not follow rigid movement.

For the reconstruction approach, the density of the gridlines used at this stage is the same over all the parts of the model. It is obvious that more gridlines should be used in the regions of greater geometric changes and fewer gridlines are required in the regions of smaller geometric changes. So, one of the future investigations can be focused on the adaptive use of the gridlines for reconstruction. Another possible extension is to reconstruct a surface that can be used for computer animations. It implies the need for an underlying structure for the face model instead of just a collection of meshes or triangles. Furthermore, effort can be made on getting rid of some of the assumptions made at the current stage.

Bibliography

Papers / Articles

<Face Detection / Feature Extraction>

[a1] Lars-Peter Bala, Kay Talmi and Jin Lin, "Automatic Detection and Tracking of Faces and Facial Features in Video Sequences, " in *Picture Coding Symposium 1997.*

[a2] R. Chellappa, S. Sirohey, C.L. Wilson, C.S. Barnes, "Human and Machine Recognition of Faces: A Survey, " *CAR-TR-731, CS-TR-3339, DACA76-92-C-0009, Computer Vision Laboratory, Center for Automation Research, University of Maryland; National Institute of Standards and Technology*, August 1994.

[a3] Roel Hoogenboom & Michael Lew, "Face Detection Using Local Maxima, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1996*, p.334-339.

[a4] Farzin Mokhtarian and Riku Suomela, "Robust Image Corner Detection Through Curvature Scale Space, " *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.20, No.12, December, 1998, p.1376-1381.

[a5] Haiyuan Wu, Qian Chen and Masahiko Yachida, "Face Detection From Color Images Using a Fuzzy Pattern Matching Method, " *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No.6, June 1999, p.557-563.

[a6] Haiyuan Wu, Taro Yokoyama, Dadet Pramadihanto, and Masahiko Yachida, "Face and Facial Feature Extraction from Color Image, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1996*, p.345-350.

[a7] Alan L. Yuille, David S. Cohen and Peter W. Hallinan, "Feature extraction from faces using deformable templates, " in *Proc. 1989 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, p.104-109.

<Feature Matching>

[b1] Liyanage C. DE SILVA, Kiyoharu AIZAWA, and Mitsutoshi HATORI, "Detection and Tracking of Facial Features by Using Edge Pixel Counting and Deformable Circular Template Matching, " *IEICE Trans. Information and Systems*, Vol. E7-D No.9 SEPT., 1995, p.1195 – 1207.

[b2] Maurizio Pilu, "A direct method for stereo correspondence based on singular value decomposition, " in *IEEE CVPR'97*, Puerto Rico, June 1997.

[b3] Zhengyou ZHANG, Rachid DERICHE, Olivier FAUGERAS, Quang-Tuan LUONG, "A Robust Technique for Matching Two Uncalibrated Images through the Recovery of the unknown Epipolar Geometry, " *INRIA Research Report No.2273*, May 1994.

<Multiresolution / Wavelets>

[c1] G. Beylkin, R. Coifman, V. Rokhlin, "Fast Wavelet Transforms and Numerical Algorithms I".

[c2] Peter J. Burt, Edward H. Adelson, "The Laplacian Pyramid as a Compact Image Code, " *IEEE Transactions on Communications*, Vol. COM-31, No.4, April 1983, p.532-540.

[c3] Jean-Pierre Djamdji, Albert Bijaoui, Roger Maniere, "Geometrical Registration of Images: The Multiresolution Approach, " *Photogrammetric Engineering & Remote Sensing*, Vol.59, No.5, May 1993, p.645-653.

[c4] Bjorn Jawerth and Wim Sweldens, "An Overview of Wavelet Based Multiresolution Analyses, " *SIAM Rev.*, 36(3), 1994, p377-412.

[c5] Michael S. Lew, Thomas S. Huang, "Optimal Multi-Scale Matching, " in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 1, 1999, p.88-93.

[c6] Yoshihisa Shinagawa and Tosiyasu L. Kunii, "Unconstrainted Automatic Image Matching Using Multiresolutional Critical-Point Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.20, No.9, Sept. 1998, p.994-1010.

[c7] Eric J. Stollnitz, Tony D. DeRose, David H. Salesin, "Wavelets for Computer Graphics: A Primer Part 1, " *IEEE Computer Graphics and Applications*, 15(3), May 1995, p.76-84.

[c8] Eric J. Stollnitz, Tony D. DeRose, David H. Salesin, "Wavelets for Computer Graphics: A Primer Part 2, " *IEEE Computer Graphics and Applications*, 15(4), July 1995, p.75-85.

[c9] Wim Sweldens and Peter Schroder, "Building Your Own Wavelets at Home, " in *"Wavelets in Computer Graphics", ACM SIGGRAPH Course Notes*, 1996.

<Pose Estimation>

[d1] Qian Chen, Haiyuan Wu, Takeshi Fukumoto and Masahiko Yachida, "3D Head Pose Estimation without Feature Tracking, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1998*, p.88 – 93.

[d2] Gaile G. Gordon, "3D Pose Estimation of the Face from Video".

[d3] Thanarat Horprasert, Yaser Yacoob and Larry S. Davis, "Computing 3-D Head Orientation from a Monocular Image Sequence, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1996*, p.242 – 247.

[d4] Ming XU and Takao AKATSUKA, "Detecting Head Pose from Stereo Image Sequence for Active Face Recognition, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1998*, p.82 – 87.

<Stereo matching / Stereo reconstruction>

[e1] Pascal Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features, " *Machine Vision and Applications*, Vol. 6, Springer-Verlag 1993, p.35-49.

[e2] P. Fua and Y.G. Leclerc, "Object-Centered Surface Reconstruction: Combing Multi-Image Stereo and Shading, " *IJCV(16)*, No.1, September 1995, p.35-56.

[e3] ZengFu Wang and Noboru Ohnishi, "3-D Reconstruction Based on Projective Invariants, " in *ACCV'95 (Second Asian Conference on Computer Vision).*

<Modeling / Reconstruction>

[f1] J. K. Aggarwal and C.H Chien, "3-D Structures from 2-D Images".

[f2] Takaaki Akimoto, Yasuhito Suenaga, Richard S. Wallace, "Automatic Creation of 3D Facial Models, " *IEEE Computer Graphics and Applications 13*, 5, September 1993, p.16-22.

[f3] Volker Blanz, Thomas Vetter, "A Morphable Model For the Synthesis of 3D Faces, " in *SIGGRAPH'99*, p.187-194.

[f4] Douglas DeCarlo and Dimitris Metaxas, "Deformable Model-Based Face Shape and Motion Estimation, " in *Proceeding of the 2nd International Conference on Automatic Face & Gesture Recognition*, 1996, p.146-150.

[f5] Olivier Faugeras, "Stratification of 3-D vision: projective, affine and metric representations, " *INRIA*.

[f6] P. Fua and Y.G. Leclerc, "Using 3-Dimensional Meshes To Combine Image-Based and Geometry-Based Constraints, " in *European Conference on Computer Vision*, May 1994, p.281-291.

[f7] P. Fua and C. Miccio, "Animated Heads from Ordinary Images: A Least Squares Approach, " *Computer Vision and Image Understanding*, September, 1998.

[f8] Horace H.S. Ip, Lijun Yin, "Constructing a 3D individualized head model from two orthogonal views, " *The Visual Computer (1996) 12*, p.254-266.

[f9] Tsungeya Kurihara and Kiyoshi Arai, "A transformation method for Modeling and Animation of the Human Face from Photographs, " in *Computer Animation '91*, N. Thalmann & D. Thalmann.

[f10] Yuencheng Lee, Demetri Terzopoulos, and Keith Waters, "Realistic Modeling for Facial Animation, " in *SIGGRAPH'95*, p.55-62.

[f11] Richard Lengagne, Pascal Fua and Olivier Monga, "3D Stereo Reconstruction of Human Faces driven by Differential Constraints, " 19 March 99.

[f12] Richard Lengagne, Pascal Fua, Olivier Monga, "3D Face Modeling from Stereo and Differential Constraints, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1998*, p.148-153.

[f13] Richard Lengagne, Olivier Monga, and Pascal Fua, "Using Differential Constraints to Generate a 3D Face Model from Stereo, " *Face Recognition.*

[f14] Richard Lengagne, Olivier Monga, and Pascal Fua, "Using Differential Constraints to Reconstruct Complex Surfaces from Stereo, " in *CVPR'97*, p.1081-1086.

[f15] Richard Lengage, Jean-Philippe Tarel, Olivier Monga, "From 2D Images to 3D Face Geometry, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1996*, p.301-306.

[f16] Roger Mohr, Boubakeur Boufama, Pascal Brand, "Understanding positioning from multiple images, "*Artificial Intelligence 78 (1995)*, p213-238.

[f17] Roger Mohr and Bill Triggs, "Projective Geometry for Image Analysis, " *A Tutorial given at ISPRS*, Vienna, July 1996.

[f18] Hrroshi Saji, Hirohisa Hioki, Yoshihisa Shinagawa, Kensyu Yoshida, Tosiyasu L. Kunii, "Extraction of 3D Shapes from the Moving Human Face Using Lighting Switch Photometry, " *Creating and Animating the Virtual World*, Springer-Verlag 1992, p.69-86.

[f19] Kuntal Sengupta and Jun Ohya, "Human Face Structure Estimation from Multiple Images Using the 2D Affine Space, " in *IEEE Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition 1998*, p.106-111.

[f20] Yoshihisa Shinagawa, Tosiyasu L. Kynii and Alexander G. Belyaw, Taketo Tsukioka, "Shape Modeling and shape analysis based on singularities, " *International Journal of Shape Modeling*, Vol. 1, No.3 (1995).

[f21] Demetri Terzopoulos and Keith Waters, "Techniques for realistic Facial Modeling and Animation, " in *Computer Animation '91*, Springer-Verlag.

[f22] Nadia Magnenat Thalmann, Prem Kalra, Marc Escher, "Face to Virtual Face, " *IEEE Proceedings Journal*, Vol. 86, No.5, May 1998.

[f23] Thomas Vetter and Volker Blanz, "Estimating Coloured 3D Face Models from Single Images: An Example Based Approach, " in *ECCV'98 Vol II*, p.499-513.

[f24] Thomas Vetter and Volker Blanz, "Generalization to Novel Views from a Single Face Image, " in *Proceedings: 3D Image Analysis and Synthesis'97*, p.43-50.

[f25] Lijun Yin, Anup Basu, "MPEG4 Face Modeling Using Fiducial Points, " in *ICIP97*, Volume I, Human Machine Interfaces, p.109.

[f26] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah, "Shape from Shading: A survey, " *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No.8, August 1999, p.690-706.

[f27] Zhengyu Zhang, Gang Xu, "A General Expression of the Fundamental Matrix for Both Perspective and Affine Cameras, " in *Proc. Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, p.1502-1510.

<Color Model>
[g1] Adobe Systems Incorporated, "Color Models - Adobe Technical Guides, "
, http://www.adobe.com/support/techguides/color/colormodels/main.html, 2000.

Books

[h1] Olivier Faugeras, *Three-Dimensional Computer Vision, A Geometric Viewpoint*, MIT Press.

[h2] Foley, van Dam, Feiner, Huges, *Computer Graphics Principles and Practice, Second Edition in C*, Addison-Wesley Publishing Company.

[h3] Rafael C. Gonzalez, Richard E. Woods, *Digital Image Processing*, Addison-Wesley Publishing Company.

[h4] Robert M. Haralick, Linda G. Shapiro, *Computer and Robot Vision Volume II*, Addison-Wesley Publishing Company.

[h5] Berthold K. P. Horn and Michael J. Brooks, *Shape from Shading*, MIT Press.

[h6] Ramesh Jain, Rangachar Kasturi, Brian G. Schunck, *Machine Vision*, McGraw-Hill Inc.

[h7] Reinhard Klette, Karsten Schluns, Andreas Koschan, *Computer Vision Three-Dimensional Data from Images*, Springer.

[h8] Vishvjit S. Nalwa, *A Guided Tour of Computer Vision*, Addison Wesley.

[h9] John C. Russ, *The Image Processing Handbook, Second Edition*, CRC Press.

[h10] Milan Sonka, Vaclav Hlavac, Roger Boyle, *Image Processing, Analysis, and Machine Vision*, PWS Publishing.

[h11] Eric J. Stollnitz, Tony D. Derose, David H. Salesin, *Wavelets for Computer Graphics Theory and Applications*, Morgan Kaufmann Publishers, Inc.

[h12] Emanuele Trucco, Alessandro Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice Hall.

[h13] Gunter Wyszecki & W. S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulas*, John Wiley & Sons, Inc.