3D Face Shape Analysis

Student: Adam Raine

Supervisor: Professor A.D. Marshall

Moderator: Dr X. Sun

Abstract

Statistical shape models can be used to effectively describe the structure and variety of the human face. Establishing a correspondence between 4747 3D face meshes allows for a detailed analysis of the principal components of variation in the face. Meshes are aligned using Procrustes Analysis, then warped into close proximity using thin-plate splines. Corresponding points are found, and the meshes are returned to their aligned positions. Principal Component Analysis is performed, and the end result is a shape model that reduces the dimensionality of the mesh data down to comparatively few modes of variation. A shape model with 138 modes of variation was created, which describe 95% of the variance. This model can be used for classification of facial structures, which could be a valuable tool for providing an objective analysis of a subject's face.

Acknowledgements

I would like to thank Prof. David Marshall for his continual support and encouragement throughout a consistently challenging project, and for providing the code required to perform PCA upon my data, as well as a machine to process that data.

I would also like to thank my friends and family for their help and understanding during the most difficult sections of the year.

Table of Contents

1. Introduction

Many rare genetic syndromes present themselves through congenital malformations of the structure of the face. Such syndromes include Miller-Dieker Syndrome, Noonan Syndrome, Velocardialfacial Syndrome (VCFS), and Williams syndrome (Winter, 1996). As a result, these syndromes may be diagnosed in individuals by searching for these abnormalities. However, this requires a training and experience. Inexperienced geneticists may have trouble due to limited experience with certain syndromes, or due to lack of exposure to a sufficient range of ages and ethnicities in their patients. This is all part of the training of a geneticist, but ideally we would minimise the amount of time and effort that is required to train an individual so that they can reliably diagnose patients. If the geneticist were to have a tool that could objectively analyse a face for malformations, then this would be of great assistance in the training process (Hammond *et al*, 2004). One way to achieve this would be to use Computer Vision techniques to perform a detailed analysis of a large set of human faces. A 3D face scanner is used to create a 3D model of each face, and then the faces are analysed as a group. From this group, we can form a shape model based off an average face. Variations in the model can describe general facial types and structures. A patient's face could be read in, compared to the model, and if certain types of variation are present, a classifier can make a diagnosis based off this. However, there are a number of challenges involved. Unlike a human, a computer is not trained to recognise faces. We can recognise a set of points in 3D space as a nose or a mouth, but computers require specific training to do so. This simply isn't feasible, and isn't necessary either. By performing an in depth mathematical analysis of the 3D points that make up a 3D model derived from a scan, we can analyse the face as a whole. To do this, each face must be aligned with the rest of the faces in the set, so that eyes match up with eyes, mouths match up with mouths, and so on. From there, warping each face onto an arbitrary face allows us to compare each face to that arbitrary face. By doing this, we can compare each face to the others in the set, since we now know that a vertex in one location on one face will be at roughly the same location on another. From this, we can perform Principal Component Analysis (PCA) to analyse where the principal modes of variation are, as opposed to measuring the variation of each individual vertex. Using this approach, Hammond *et al* (2004) successfully managed to distinguish between controls, individuals with Noonan Syndrome, and individuals with VCFS with an accuracy rate of upwards of 80%. Ideally, this project would achieve similar results. There are a number of questions about the approach that this report will seek to answer. How should faces be aligned to minimise distortion? What are the principal modes of variation in the human face? How can these modes be used?

2. Background

Polygon mesh

A polygon mesh is a method for representing 3D objects efficiently using computer graphics. Each mesh has a set of vertices that describe points in 3-dimensional euclidean space. It also contains a set of faces, where a face is a set of points that are connected by straight lines to form a polygon in the mesh. For example, in a triangle mesh, each item in the list of faces will contain a reference to 3 points, since 3 points are required to draw a triangle. Triangles and quadrilaterals are typically used, as they follow simple geometric rules that make rendering and processing the model easier. The models I have worked on are triangle meshes generated by a 3D face scanner, and contain several thousand vertices.

Landmarks

A landmark is a point on an object that always appears in the same place for all examples of that object. A landmark on a map is a place that isn't going to change location over time, even if the map changes, such as places like the Eiffel Tower or Big Ben. On a human face, the location of the tip of the nose is going to be in roughly the same place on each face when compared to the rest of the landmarks. The tip of the nose will always be between the edges of the nostrils, but protruded. The outer edges of the nostrils will always be roughly below the inner corners of the eye. The locations of these points will be different on every person, but they will always be present, and the positions of the landmarks will always be in the same rough arrangement. We can create a model of facial structure by analysing how these landmarks vary in a large set of example faces (Cootes & Taylor, 2004).

Good landmarks are typically points of high curvature, such as the inner and outer corners of the eyes, or points where object boundaries meet, such as where the ridges of the philtrum meet the nose. Other easily locatable biological landmarks are also useful, such as the tip of the nose or the bottom of the chin. However, some points are hard to place exactly; the precise end of each eyebrow is hard to position, and may have been altered by cosmetic enhancements such as plucking. Automatic placement of landmarks on models is possible (Cootes & Taylor, 2004), but using a set of trained individuals is generally simpler, if more time consuming.

Procrustes Analysis

Procrustes Analysis is a statistical shape analysis tool that finds the Procrustes Distance between two sets of points, where Procrustes Distance d can be defined as:

$$d = \sqrt{(u_1 - x_1)^2 + (v_1 - y_1)^2 + (w_1 - z_1)^2 + \cdots}$$

where d is a measure of the total distance between two corresponding sets of points, u, v and w are the horizontal, vertical and depth components of one set of points, and x, y and z are the coordinates of the other set (Bookstein, 1991).

Procrustes analysis is performed by first transforming each object's points so that their centroid lies at the origin. Then, the scale difference between the two sets is removed by rescaling each object so that their root mean square distances to the origin are both 1. Lastly,

the rotational difference between the two objects is removed by fixing one of the sets of points, so that it can act as a reference. The other set is then rotated around the origin, so that the optimal angle of rotation theta can be found. Angle θ is optimal if the sum of squared distances between the sets of points is minimised. These transformations combined give us the optimal superimposition, and the procrustes distance can then be calculated. A procrustes distance of 0 indicates that the two sets of points are identical in shape. If we skip the scaling step, Procrustes Analysis allows us to rigidly align two objects such that they are as closely aligned as possible, without altering the shape of each object.
Procrustes Analysis is for two objects, but can be extended to multiple objects in a process called Generalised Procrustes Analysis (Gower, 1975). GPA can optimally superimpose a set of objects, so that they are all aligned to the mean set of points. First, one object is arbitrarily picked to act as the reference. Then, using Procrustes Analysis, each object is optimally superimposed onto the reference object. With the new positions of each object, the mean position is calculated, and if the Procrustes distance between the reference object and the mean object is above a certain threshold, the mean object becomes the new reference, and the cycle repeats.
Procrustes Analysis is an important tool for the project, as it is key to the rigid alignment phase.

Thin Plate Spline Warping

Thin Plate Spline Warping is a method to warp a surface, such as the 3D face scans used in the project. The thin plate spline is analogous to a thin metal sheet that has a number of points where the plate can be pushed and pulled out of shape (Bookstein, 1989). If the points are manipulated, the plate bends and warps to fit. However, it does this with as little distortion to the plate as possible; it maintains the minimum bending energy. This kind of function is very useful for object registration, as images or meshes can be warped to fit a certain template, but will change as little as possible, just like the titular plate. The process used in my project requires a large number of meshes to be superimposed on a reference mesh. In order to find out which vertices correspond, the meshes need to be very closely matching, which is not possible with simple rigid transformations. By using thin plate spline warping, we can minimise any loss of information contained in the mesh, and find an accurate correspondence.

Principal Component Analysis

Principal Component Analysis is a statistical analysis technique used to reduce the dimensionality of data without losing any information contained in the data. For example, in my project, each mesh contains several thousand vertices, each of which is a dimension in the data. Analysing the position of a single vertex is unlikely to provide much insight; for effective analysis, the number of variables must be reduced. So, rather than analysing the position of individual vertices, PCA may instead be used to find the areas with the most variation, such as the height and width of the face, the width of the mouth, the position of the nose and eyes, etc. The original variables (the position of each vertex) have been transformed to a new set of variables (the width of the face within a shape model, etc), which are called the principal components (Jolliffe, 2002). Each principal component has a degree of correlation with the

original data, but is also independent from all the other principal components, i.e. the width of a face can be altered without changing its height. PCA does not alter the data in any way; it simply rearranges it so that the factors which affect the data are more visible.

The process to perform PCA is the following, as described by Cootes & Taylor (2004):

1) Calculate mean of data

$$\bar{x} = \frac{1}{s}\sum_{i=1}^{s} x_i$$

2) Calculate covariance of the data

$$S = \frac{1}{s-1}\sum_{i=1}^{s} (x_i - \bar{x})(x_i - \bar{x})^T$$

3) Calculate eigenvectors and eigenvalues λ_i of S (sorted so that $\lambda_i \geq \lambda_{i+1}$)

The eigenvectors are the principal components. Processing data along these axes will allow for a much simpler analysis of any data, compared to analysing it against a set of correlated variables.

Statistical Shape Model

The simplest way to check if one object is of a certain type is to compare that object to a 'golden' example of that object, and then use various correlation measures to check if the object matches (Cootes & Taylor, 2004). However, this is difficult to do for a face; we don't have an example of a 'golden' face because every face is different. Comparing to an average face might be possible, but that average face doesn't necessarily exist in reality. If we were classifying types of facial features, we would also need perfect examples of those. The variability of the human face makes simple comparisons like this difficult. Instead, a statistical shape model can be created.

The shape of any object may be represented by a set of *n* points in any number of dimensions, usually two or three. For example, a triangle is a set of 3 connected points. An equilateral triangle is a set of 3 connected points of equal distance apart. These constraints describe what any example of that object will look like. The same constraints can be applied to a face; a nose will always be below the eyes, and will extrude out of the face. However, a computer doesn't know what a "nose" is, it only understands mathematical information. So, if a set of faces can be positioned in the same frame of reference, it is possible to calculate the residual differences between them. By completing this process a number of times, a set of residuals can be obtained which describes the range of values that points on a face can occupy, compared to a reference. Performing PCA on these residuals highlights where the variation in the data is highest, and creates a shape model. This shape model can have a mesh fed into it that outputs data describing where it lies within the model, and that same data can be fed back in to output the original mesh. Entirely new meshes can be generated by feeding in new values for the modes of variation.

Data

The data I will be using is from the Avon Longitudinal Study of Parents and Children (ALSPAC), also known as Children of the 90s. It is a long term health research project involving the study of over 14000 children as they grow up. I have a selection of untextured, anonymous facial scans from this group, and each scan has been manually landmarked and classified by Caryl Wilson of the School of Dentistry. The landmark points and their location on the human face are as follows:

Glabella (1) – Most prominent midline point between eyebrows
Nasion (2) – Deepest point of nasal bridge
Endocanthion L/R (3/4) – Inner commissure of the left and right eye fissure
Exocanthion L/R (5/6) - Outer commissure of the left and right eye fissure
Palpebrale superius L/R (7/8) – Superior mid-portion of the free margin of upper left and right eyelids
Palpebrale inferius L/R (9/10) – Inferior mid-portion of the free margin of lower left and right eyelids
Pronasale (11) – Most protruded point of the apex nasi
Subnasale (12) – Mid-point of angle at columella base
Alare L/R (13/14) – Most lateral point on left and right alar contour
Labiale superius (15) – Mid-point of the upper vermillion line
Labiale inferius (16) – Mid-point of the lower vermillion line
Crista philtri L/R (17/18) – Point on left and right elevated margins of the philtrum just above VL
Cheilion L/R (19/20) – point located at left and right labial commissure
Pogonion (21) – Most anterior mid-point of the chin

The classifications of the data are detailed by Wilson et al (2012), and include different properties of the philtrum, cupid's bow, upper and lower vermillions, sub-lip and nasolabial angle. A diagram of the positions of the landmarks on the face is visible in Fig 3. of the interim report. Each mesh has a number assigned for each of the different mouth features, which denotes which category that feature falls under.

Classification of the data

Each mesh has a corresponding classification. This classification denotes the sex of the subject, and the presence of certain features of the mouth and lips as described by Wilson *et al* (2012). Certain locations within the shape model should correspond to the presence of these features. An effective classifier would be able to read in a facial mesh, find out where that mesh lies within the shape model, and classify the features present with a high degree of accuracy.

3. Implementation

Algorithm

The first step in the algorithm is to read in all the landmark data, and perform Generalised Procrustes Analysis to align each set of landmarks to the mean set. The landmarks are stored in a .csv file, so I created a script to read the data and convert it into a matrix format, where each column corresponds to a set of landmarks, and each row corresponds to the X, Y, or Z location of a specific landmark. In addition to this, a vector of ID numbers is created, which correspond to the columns in the matrix. By looking up the column number of the current landmark set in the ID table, it is possible to associate a set of landmarks with its corresponding .obj file. Once the landmarks have been aligned, the new landmark locations and the transformations that they underwent are stored, and the original landmark data is discarded. This completes the rigid alignment phase, which allows for smaller thin-plate spline warps to minimise distortion.

The next part of the process is non-rigid alignment. The first step is to load the reference mesh that the other meshes will be aligned to. By aligning all the meshes to this one, the points where each mesh corresponds with the others can be found. The base mesh is rigidly transformed onto the its newly transformed set of landmarks. Then, this mesh undergoes a thin-plate spline warp to fit it to the mean set of landmarks. This will be the mesh that every other mesh is compared to. The same steps are undertaken by every other mesh; it is first rigidly transformed onto the procrustes aligned set of landmarks, and it then undergoes a thin-plate spline warp onto the mean set of landmarks.

The next step is to find which vertices correspond. A k-nearest neighbour search is performed on the reference mesh and the current warped mesh to find which vertices on the warped mesh correspond with vertices on the base mesh. By creating a copy of the base mesh and assigning the location of each vertex to its corresponding vertex on the warped mesh, an approximate copy of the warped mesh can be made, with the same connectivity as the base mesh. This is important for the PCA process. Since the coordinates for each mesh before the thin-plate spline warp are already known, the new set of vertices can be assigned the same locations. This process is completed for every mesh, and the original mesh data is discarded. Since each of the new meshes has the same number of vertices, they may now be rigidly aligned using generalised procrustes analysis. This provides a more accurate mean set of vertices, and the residual differences between each set of vertices and the mean set will be what the shape model is based on. PCA is then performed on each set of vertices to obtain the model.

The model allows for a mesh to be ‘projected’ into it, where the output is the point that the mesh occupies within the model, i.e. the values of the principal components. It also allows for a set of points to be ‘unprojected’ into real space, i.e. to create a set of vertices that correspond to those particular principal component values. A mesh may then be rendered by using the faces of the reference mesh.

Initial Procrustes test

To ensure that Procrustes alignment would work as I expected it to, I experimented with the landmark data I had been given. I took two sets of landmarks, plotted them, and then performed the procrustes alignment. Plotting the two sets of landmarks again should show that the points were much closer together, which can be observed in Fig 3.1. Although the points towards the top of the plot (the eye landmarks) are actually much closer together to the pre-procrustes points, the points towards the lower end of the plot are much closer together post-procrustes. The average distance between each corresponding point pre-procrustes is 7.6491, and the average distance post-procrustes is 4.7876. Note that the Procrustes alignment in this test included a scaling component to better demonstrate the difference between the two sets of points. The procrustes alignment used in the program omits the scaling step. This test showed that the Procrustes Alignment algorithm would be useful for rigidly aligning landmarks, as the overall gap between the two sets was clearly reduced. Alignment is necessary to provide the best frame of reference possible; it is simpler to compare a set of points when differences in translation and rotation have been removed.

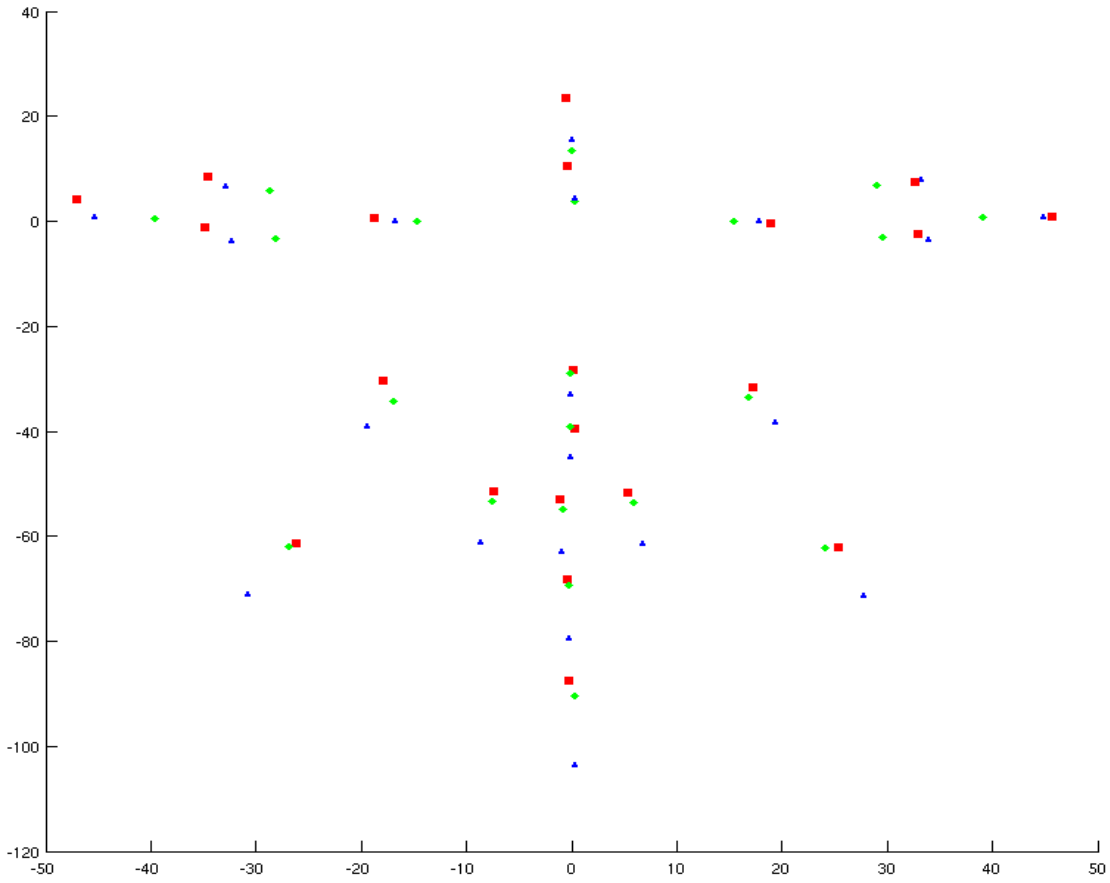

Fig 3.1: Initial Procruses alignment test. Reference landmarks are shown as red squares, pre-procrustes alignment landmarks are shown as blue triangles, and post-procrustes alignment points are shown as green diamonds. Procrustes alignment to the reference landmarks results in a much closer fit overall.

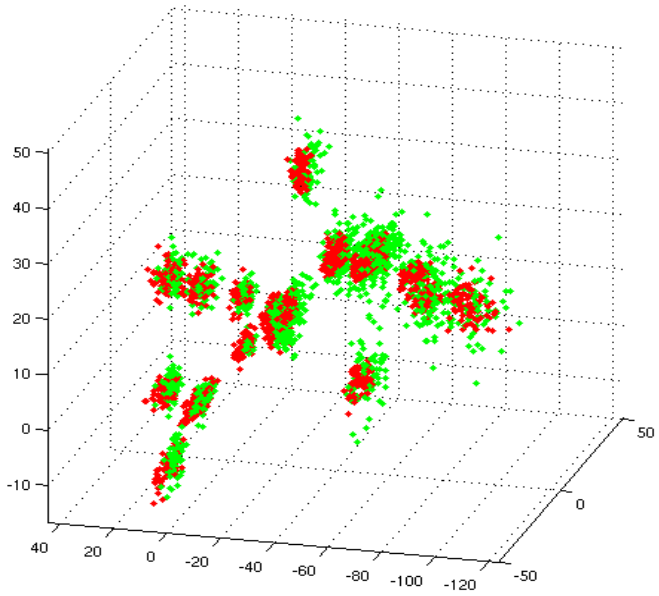

Fig 3.2: A plot of all 4747 landmark sets. The green points are the landmark sets before GPA, the red sets are the same points after GPA. The red points are much more tightly clustered.

GPA test

The next step with Procrustes alignment was to implement Generalised Procrustes Analysis. GPA is important because when creating the model, it is important to make sure that every set of landmarks is closely aligned to minimise any variance caused by differences in translation and rotation. Aligning to the mean set of landmarks is important, as the shape model will be calculated from residual differences from the mean. To achieve this, I used what I had learned about Procrustes analysis to create a function to align multiple sets of landmarks to a reference set. Once this had been achieved, it was a relatively simple to run the process again, except aligning to the mean set of landmarks resulting from the output of the previous execution. I executed this function on all the landmark data I had, and compared the post-GPA results to the pre-GPA data, visible in Fig 3.2. The post GPA points were clearly more tightly clustered. The average distance from the mean before was 3.6532, after was 2.7927. I tested the function out on all the landmark data I had, and plotted the range of values that were within 2 standard deviations from the mean for each landmark. This plot is visible in Fig 3.3. The plot looked very similar to a plot made by Toma *et al* (2011), in the process of creating a shape model using the same landmarks as me. The fact that the two plots were so similar meant that I could be confident that my GPA algorithm was working correctly, as the shape model simply describes the range of possible landmark positions. The range of positions is actually fairly small, but even a small difference in the position of a landmark can make a big difference to the overall shape of a face.

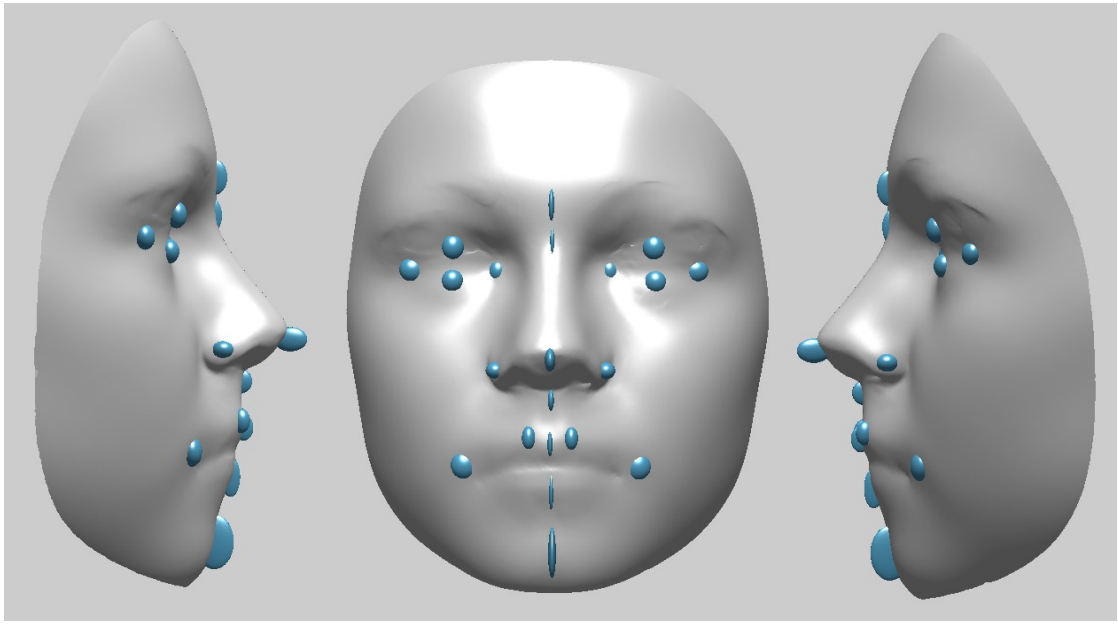

Fig 3.3: A plot of the range of positions that landmarks can take within +/- 2 standard deviations.

TPS Test

The thin-plate spline warping function needed testing to ensure that it was correctly warping one mesh onto a new set of landmarks, as I had sourced the function from Matlab Central, MathWorks' online repository of user-submitted files. Matlab comes with a thin-plate spline warp function, but this only worked in two dimensions, whereas I needed it to work in three dimensions. Obtaining this function online would save a great deal of time, as I would need to completely understand the maths behind the warp if I wanted to implement it in 3D, which would take a great deal of extra background reading that I had not allotted time for in my interim time plan. Since the function came with little in the way of guidance or comments, I had to spend some time figuring out how it worked, and how the input and output data needed to be arranged. I wrote some scripts to rearrange vertex data to the required format, and once I had successfully achieved complete execution of the function with no errors, I began a series of tests to check that the function was doing what I needed. One of the first tests I did was to take a 2-dimensional grid of points as my control points, and create a plane that was aligned to these points as my

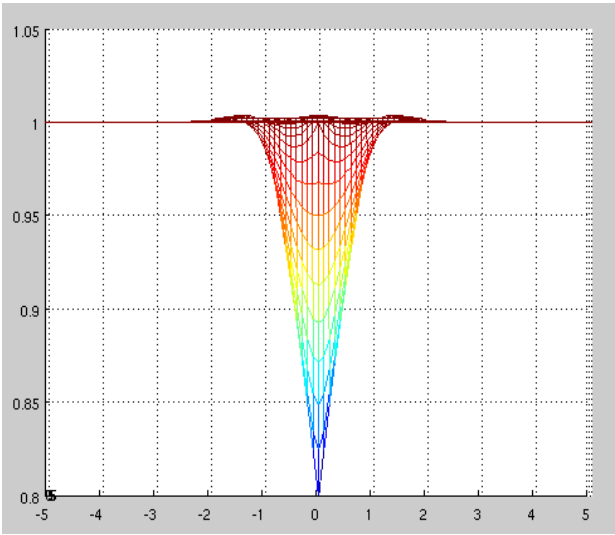

Fig 3.4: A plane with a single control point translated below the plane. A TPS warp i used to warp the plane to fit.

'mesh'. Next, I created a copy of the control points, and translated the central point so that it lay below the rest, and the plane was warped onto the altered control points. I did this test to see what effect the TPS warp would have on an existing smooth surface. By moving only one control point, it would be possible to observe how much of the surface was affected by the warp, and to what extent. The results of the test are observable in Fig 3.4. The plane appears to have deformed towards the altered point, without deforming the rest of the plane too much. After that, I varied a number of control points randomly in the z axis. To see how an inverse warp would work, I attempted to warp the randomly warped plane back to its original position (Fig 3.5). To the human eye, the result is nearly identical. However, there are minute differences in the positions of the vertices in the mesh. I also tried the same experiment with variation in all dimensions. The results were not as good (Fig 3.6). The plane is still very flat, but there is noticeable distortion in the mesh.

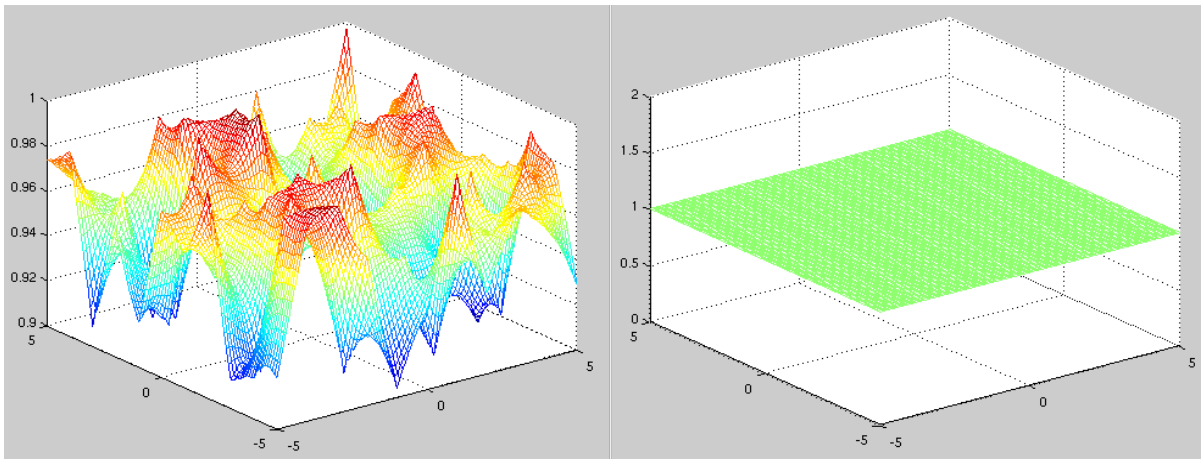

Fig 3.5: Left: A Plane's evenly spaced control points are randomly warped in the z axis. Right:The plane after warping the mesh back to the original control point locations. The mesh appears flat, but is not a perfect copy of the original.

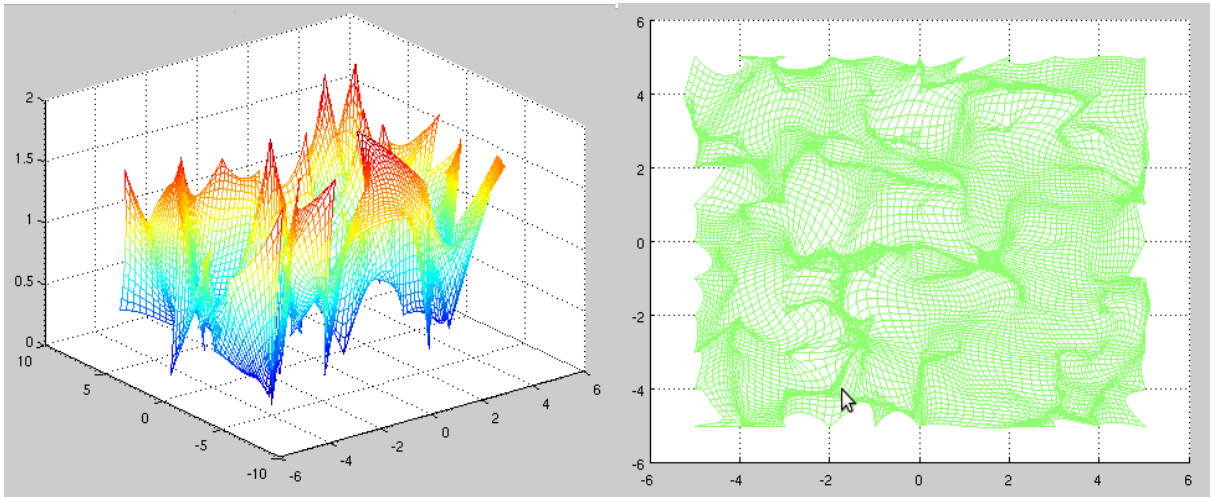

Fig 3.6: Left: a plane is randomly warped in all directions. Right: The plane is warped back to its original position, but there is noticeable distortion.

Implementing PCA

To save time, I was given pre-existing code to computer the shape model using PCA. Principal Component Analysis is computed using the follow process, as described by Cootes & Taylor (2004) and Smith (2002).

First, the data must be arranged in a simple manner for easier calculation. If the total number of observations in the data is n, then each observation must be arranged into a vector x_i, where each element of that vector is a variable, i.e. an x, y, or z axis value. Each vector is placed into an $m\,x\,n$ matrix X, where m is the number of variables in each observation.

The next step required is to calculate the mean of the data, as to calculate the variance in the data, a point from which it varies must be obtained. The mean is calculated for every variable as follows, to create a vector of mean values:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Once the mean has been calculated, the mean vector is subtracted from X to create a new $m\,x\,n$ matrix B of residuals, which has a mean of 0 for all rows.

Next, the covariance of the data must be computed. Covariance is a measure of the strength of correlation between two or more variables (Weisstein, 2013). Positive covariance values indicate that there is a positive linear correlation between the two variables, i.e. as one increases or decreases, so does the other. For negative values, as one variable increases or decreases, the other does the opposite. A value of 0 indicates that there is no correlation. The $m\,x\,m$ covariance matrix C is calculated as follows:

$$C = \frac{1}{N-1} B.B^T$$

The next step is to calculate the eigenvectors and eigenvalues of the covariance matrix C. An eigenvector of C is a non-zero vector v such that C multiplied by v yields λv, where λ is a scalar. λ is the eigenvalue corresponding to the eigenvector v. Each eigenvector is essentially a line of best fit for the data (Smith, 2002). It is simple to draw a line of best fit for data in 2 dimensions, since that can be easily visualised. Data in thousands of dimensions is a lot harder to represent, so eigenvectors are useful for characterising data with a high number of dimensions. These eigenvectors are our Principal Components. The eigenvectors are placed into a matrix V of size $m\,x\,m$, where each column is a single eigenvector. A vector of eigenvalues D is created for the eigenvalues, where the i^{th} eigenvector corresponds to the i^{th} eigenvalue. Finally, the eigenvectors and eigenvalues are sorted in order of decreasing eigenvalue. The first column of V is the most significant Principal Component, the m^{th} is the least significant.

Any of the data x from the original set X can be approximated as such (Cootes & Taylor, 2004):

$$x \approx \bar{x} + Vb$$

where $b = V^T(x - \bar{x})$

The vector b defines a set of parameters within the shape model, and each eigenvalue describes the variance of that principal component. By taking the square root of the eigenvalues, their standard deviations can be obtained. Creating a new vector b with values within +/-3 standard deviations, shapes generated will be similar to those in the original set.

If all the eigenvectors are retained, then it is possible to recreate examples from the training set exactly. However, by removing some eigenvectors, the dimensionality of the data can be further reduced with minimal loss to accuracy, so long as the least significant principal components are removed first.

Landmark Shape Model

To test my understanding of the PCA code, I attempted to make a shape model from a data set with much lower dimensionality; the landmark data. Whilst the goal of my project is to create a detailed model from every point in a mesh, it would also be possible to create a model using just the landmarks. This model would have much less detail than the full vertex model, but would be quick to compute and practice with, as well as providing an insight into what some of the more significant principal components could be.

First, I aligned the landmarks using GPA. Since no meshes were involved, the next step was simply to use Principal Component Analysis on the data to create a shape model. The model came out as expected; I could feed a set of landmarks into it to get a vector of values in the model. Feeding those values back in would result in the same landmarks that were supplied in the first place. I was able to compare my results to a previous study conducted by Toma *et al* (2011), and the results were similar. The first three modes of variation were identical.

Mode 1 was described as the "*vertical distance between the centroids of the upper and lower sets of landmarks"*. This description matched what my analysis showed. The upper sets of landmarks around the eyes and the very top of the nose could be observed moving up or down, and the lower sets around the mouth and chin could be seen moving up or down in the opposite direction, whilst the landmark positioned at the tip of the nose was roughly stationary.

Mode 2 was described as the "*horizontal distance between the centroids of the left and right sets of landmarks associated with the eyes"*, which was also observable in my model. Additionally, I noticed a small variation in the horizontal distance between landmarks associated with the mouth in my model, which appears to be represented in their model as well.

Mode 3 was described as "*z coordinate of the centroid of the landmarks associated with the nose"*, which is observable in my model as well. I also noticed movement of the chin, which did not seem to be present in their model.

There were also differences. Toma *et al* retained principal components according to the 'Kaiser – Guttman criterion' (Guttman, 1954 ; Cliff, 1988 ; Jackson, 1993). This rule states that the principal components with eigenvalues greater than the average eigenvalue should be the only components to be retained. According to this rule, they identified 14 PCs. However, I only found 11. Their 14 PCs explain 82.1% of the variation, whereas my 11 PCs explain 81.9%. My PC1 accounts for 38% of variance, whereas theirs accounts for 28.8%. My PC2 explains 10% of variance, whereas theirs explains 10.4%. My PC3 accounts for 7.4% of variance, whereas theirs accounts for 6.7%. The discrepancy was odd, given that we were working with the same data. It is possible that my Generalised Procrustes algorithm was not quite working correctly, but the results appeared similar enough that I felt confident that I had successfully created a shape model.

110 mesh test

The next step was to test the shape model process out on a selection of the meshes that I would be working on. Computing a shape model using just the landmarks was a simple process, but using the meshes would require use of all the procedures I had developed and tested during the rest of the implementation phase. I had developed each part in isolation; now I needed to put all the parts together.

There were two goals for this test. The first was to put together a script that could read a specified set of meshes and landmarks, perform the required registration, form the shape model, and output the results. As such, I had to refactor some of the functions I had written to minimise the number of calculations needed. Since the model would take a long time to calculate, any time that could be shaved off now would be beneficial later.

The second goal was to evaluate how effective the procedure would be with a reduced sample set. This reduced set may be enough to create a sufficiently detailed model, but it may also be entirely insufficient. The results of this test could be compared to the final results to see what is gained by having a larger set of training examples to work with. Either way, testing with a smaller set would be a useful dry run to find where possible points of failure might lie.

I had been given 110 meshes to experiment with, so I used these for my test. This took much longer than the landmark test; around half an hour, compared to seconds. This was to be expected, as there are 63 variables in each landmark set (x, y and z coordinates for each landmark) compared to around 15000 vertex coordinates. This long processing period made me aware of the fact that the program needed to regularly update its user as to what it was doing at any given time, to ensure that they were aware that the program hadn't crashed or frozen. The information that was output by the program could also help to debug any bugs, which would need to be fixed by the time the program was run on the full set of meshes, as the entire process would take many hours. Once the GPA had finished and the meshes had been aligned with their landmarks, thin-plate spline warping was performed on each one to form a correspondence with a base mesh. Meshes were returned to their original positions with the new mesh connectivity, and a shape model was formed. Loading the meshes, aligning them and forming the correspondence took the bulk of the time; around half an hour in total. In comparison, forming the model took a couple of minutes. The

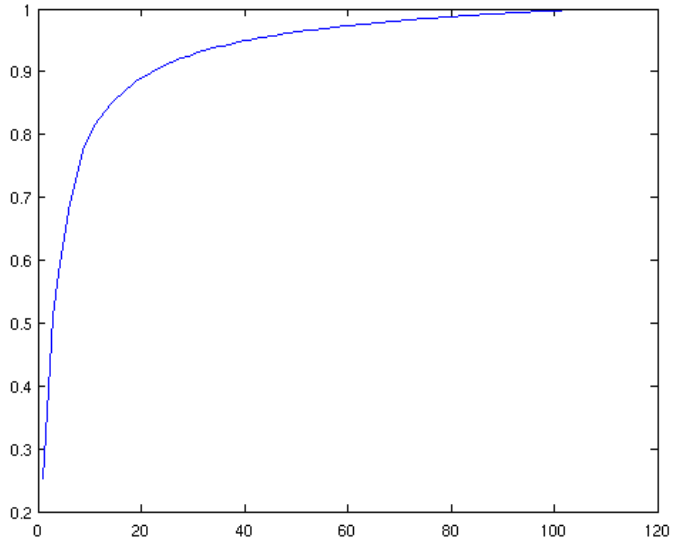

Fig 3.7: A cumulative frequency plot of the total variation represented by the modes of variation in the 110 mesh shape model. The horizontal axis shows the cumulative number of modes of variation, and the vertical axis shows the total proportion of the variation represented, out of 1.

results appeared promising; I had a shape model that allowed me to alter the modes of variation and output an entirely new mesh that resembled a human face. Varying the values supplied to the modes of variation produced changes that altered that face's appearance in a realistic way.

By applying the Kaiser-Guttman criterion, 14 modes of variation were detected, which describe 84.73% of the total variance. Fig 3.7 contains a cumulative frequency diagram of the significance of the modes. Fig 3.8 shows the effects that the first 5 modes of variation have on the face. The principal components seemed to resemble the components found in the landmark test; PC 1 here was the same as PC1 before. PC4 was similar to the landmark test's PC2, and PC5 is similar to the landmark test's PC3. However, if values supplied were outside of 2 standard deviations, then the mesh produced would be of a noticeably lower quality, with bumpy, grainier surface replacing one that is ordinarily smooth. This is likely due to the lack of data supplied to the model; without a sufficient number of examples of meshes at the extreme ends of certain principal components, the calculation may not be as accurate. The accuracy of the model and the quality of the meshes produced would be greatly improved with more sample meshes.

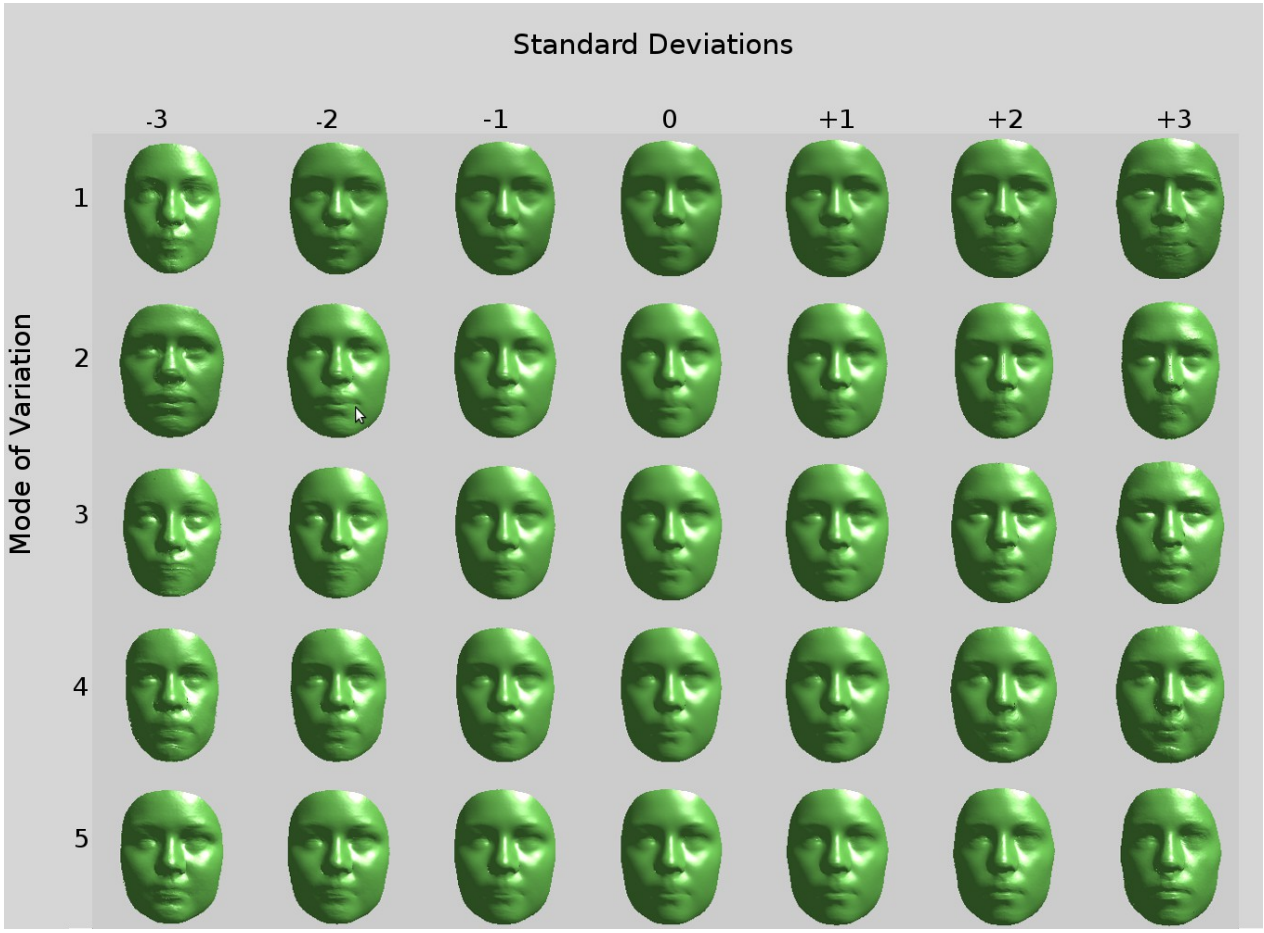

Fig 3.8: Faces in between -/+3 standard deviations of each mode of variation between 1 and 5.

Batches

To mitigate any losses from a failure in the program, I ensured that the model generation process would create backups of any data loaded so far. The process of loading in a set of meshes, rigidly aligning them to the reference set of average landmarks and forming a correspondence could take up to half an hour for around 100 meshes. Given that there are almost 5000 meshes, up to 25 hours could be lost if the data were to be improperly handled. To protect against this, I loaded up the data in batches instead of all at once, processed these batches, and then saved them so that they could be loaded again later. This meant that whilst a bug in the program would be a setback, it would not waste days of valuable processing time.

Problems Encountered

One of the issues that goes with working with a large set of data is the physical limits of computer hardware. For example, a typical mesh contains around 50000 vertices, each of which has an x, y and z value, which is stored in a double data type. On average, each .obj file takes up around 3.3MB of space, and when multiplied by 4747 (the number of scans used in total), the storage requirement comes to around 15GB. This is a great deal of space, which meant that transferring all the meshes at once would be impractical. This was not the biggest problem, however. To conduct PCA on the data, all the meshes needed to be loaded into memory at the same time. Most machines have around 1 to 4GB of memory, whereas four times that would be needed for the project. Thankfully, I was able to access a machine with 16GB, so I was able to run the program on all the data, rather than splitting it up. Computing the model in parts is possible, but reduces its overall accuracy, so processing all the data at the same time is preferable. The entire process took over 24 hours; this meant that if the program broke at any point, huge amounts of time and processing cycles would have been wasted. Rather than hope that I'd ironed out all the bugs in the program, I decided to ensure that the program saved its progress along the way. That way, if something went wrong that I hadn't anticipated, I could restart where the failure occurred. This proved to be prudent, as I did encounter a problem at the end of the process. For my 110 mesh test, the last step was to save the variable data in a file using Matlab's save command. I had written the program as a function rather than just a script, so the variables would not appear in the workspace. What I hadn't realised was that Matlab cannot save files that are over 2GB in size unless a specific flag is invoked. This meant that although the calculations had finished successfully, Matlab had failed to save the data, which would now need to be recalculated. Thankfully, only the shape model had to be rebuilt, as I had saved all the progress of the rigid registration.

The end result was a file 10GB in size. My lab machine had a single gigabyte of RAM, so loading the data on to that was not feasible. Loading the data onto a machine proved difficult; since the file was so big, it needed to be stored in a slightly different format. To access the data without loading it into memory, I used a tool called HDFView2.9, which allowed me to extract the model. The initial model turned out to be 5GB in size, which just about fit in the memory of a Mac in the lab with 6GB available.

4. Results

Despite the setbacks, the model was successfully created. Once a machine capable of analysing the results had been set up, I started to analyse the results using a number of plots and mathematical techniques.

Standard Deviation Plots

A simple way to observe how the face changes is to plot a range of values within the model. The average face can be plotted by supplying the model with a value of 0 for each mode of variation, and the effects of varying a specific mode can be observed by changing the value of that mode and setting every other mode's value to 0. By applying limits of +/- 3 standard deviations to that value, any models generated should be similar to the training set (Cootes & Taylor, 2004). To observe how the average face changes for each mode, I plotted the faces in a line at -3, -2, 0, 1, 2 and 3 standard deviations.

Heat Maps

The heat maps visualise the changes using colour. The meshes for -3, -2, -1, 0, +1, +2, +3 standard deviations are calculated, and for the x, y and z components of each vertex, the components of that vertex in the average mesh are subtracted, and a matrix of the differences is created. For each set, the average mesh is plotted and coloured according to the magnitude of the difference in a particular dimension. So, for a heat map that shows differences in the x axis, points that move in a positive direction, i.e. right, will be coloured red, and points that move in a negative direction will be coloured blue. Points that don't move very much will be coloured green. The higher the change relative to the rest of the changes, the deeper the colour. The same applies for plots in the y and z axis; red represents shifts upwards for y, and towards the screen for z. Blue represents shifts downwards for the y axis, and away from the screen for the z axis. All the heat maps are available in the supporting material.

Movies

I created animations of the modes of variation by rendering the meshes at various values between -3 and +3, with enough frames to make the transition between each frame look smooth. Each set of frames has its own reverse appended so that the animation loops cleanly. The frames are then converted into an .avi file and compressed. Viewing the movies is a good way to observe the general effect of varying different modes, although the effect can be subtle at times. Watching the movies with the heat maps on hand for reference can help to pick out the more subtle effects. All the movies are available in the supporting material.

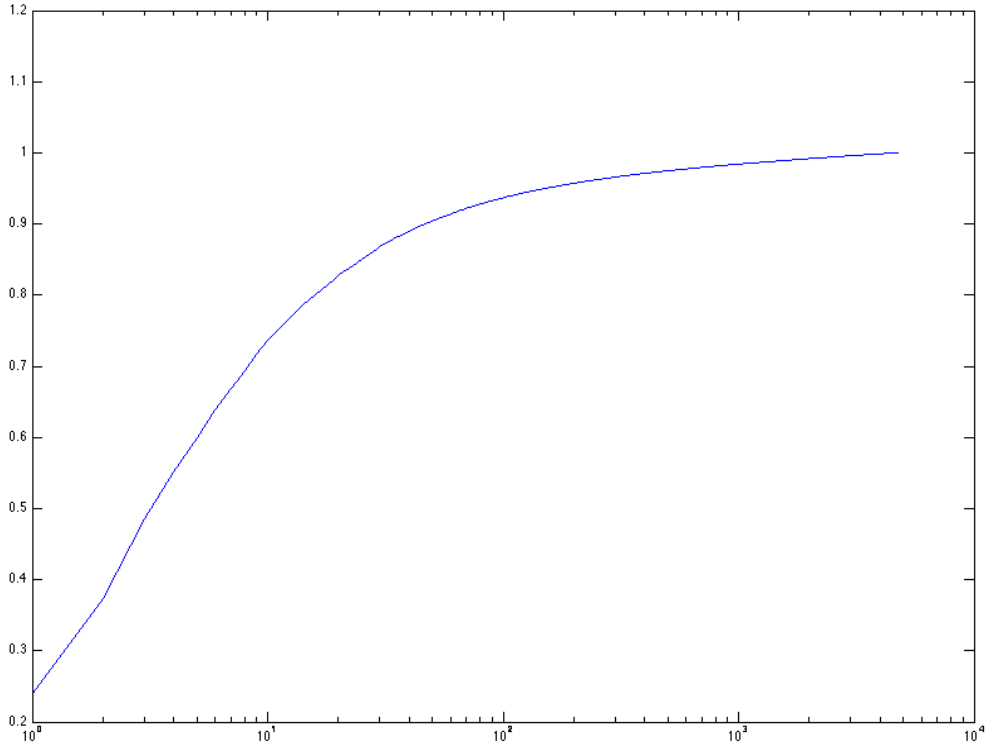

Fig 3.9: A cumulative frequency plot of the total variance occupied by the accumulation of modes of variation. The horizontal axis is logarithmic, and shows the total number of principal components. The vertical axis shows the proportion of the variation occupied by those modes, out of 1.

Modes Detected

The PCA process generated 4746 principal components in total. These are not all significant; the modes of variation at the lower end of the scale will only represent a small fraction of the total variation. Toma *et al* described using the Kaiser – Guttman criterion (Guttman, 1954 ; Cliff, 1988 ; Jackson, 1993) to retain principal components, where the only components retained are those that have a variance greater than the average variance. If I also apply this rule, then 138 modes of variation were found, which describe 94.73% of the variance altogether.

As can be seen in Fig 3.9, the significance of each mode decreases rapidly. All modes below mode 13 account for less than 1% of variance, with the first 10 representing 73.69% in total. I have detailed the effects of the first 10 modes, to show where the majority of the variation lies.

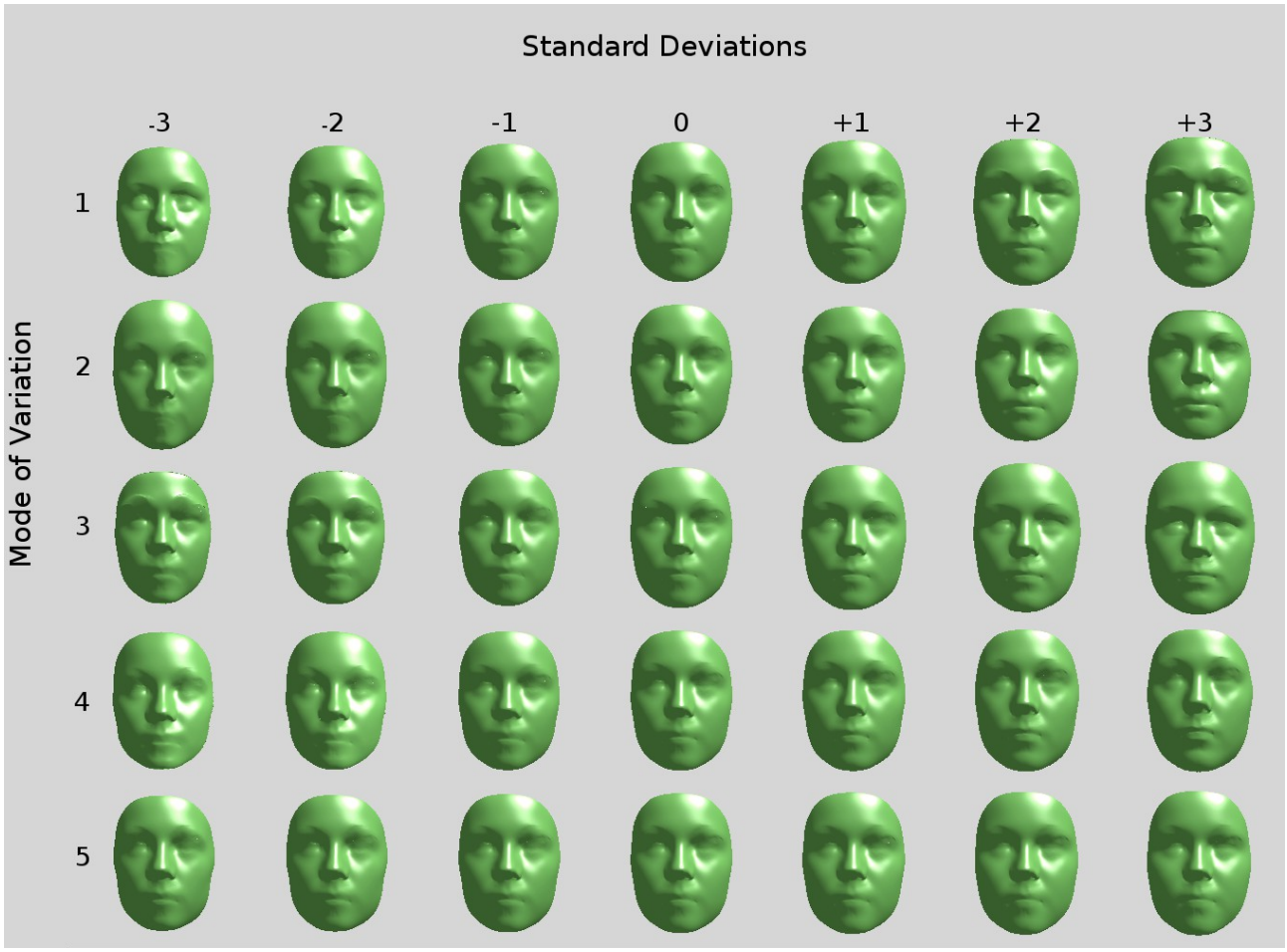

Fig 3.10: The first 5 modes of variation.

Mode 1

Mode 1 represents general growth all over the face, and accounts for 28.83% of the total variance. Mode 1's heat maps show substantial widening of the entire face, particularly at the edges of the face around the cheeks, eyes and chin. Lengthening of the face is observable all over, but is most extreme at the forehead, brow and chin. The depth of areas of the face is also affected, with the brow , nose and mouth noticeably extruding further from the face. When a negative value is supplied for this mode, contraction of the face is observable. Hutton *et al* (2003) have shown this mode to show a correlation with the age of the subject.

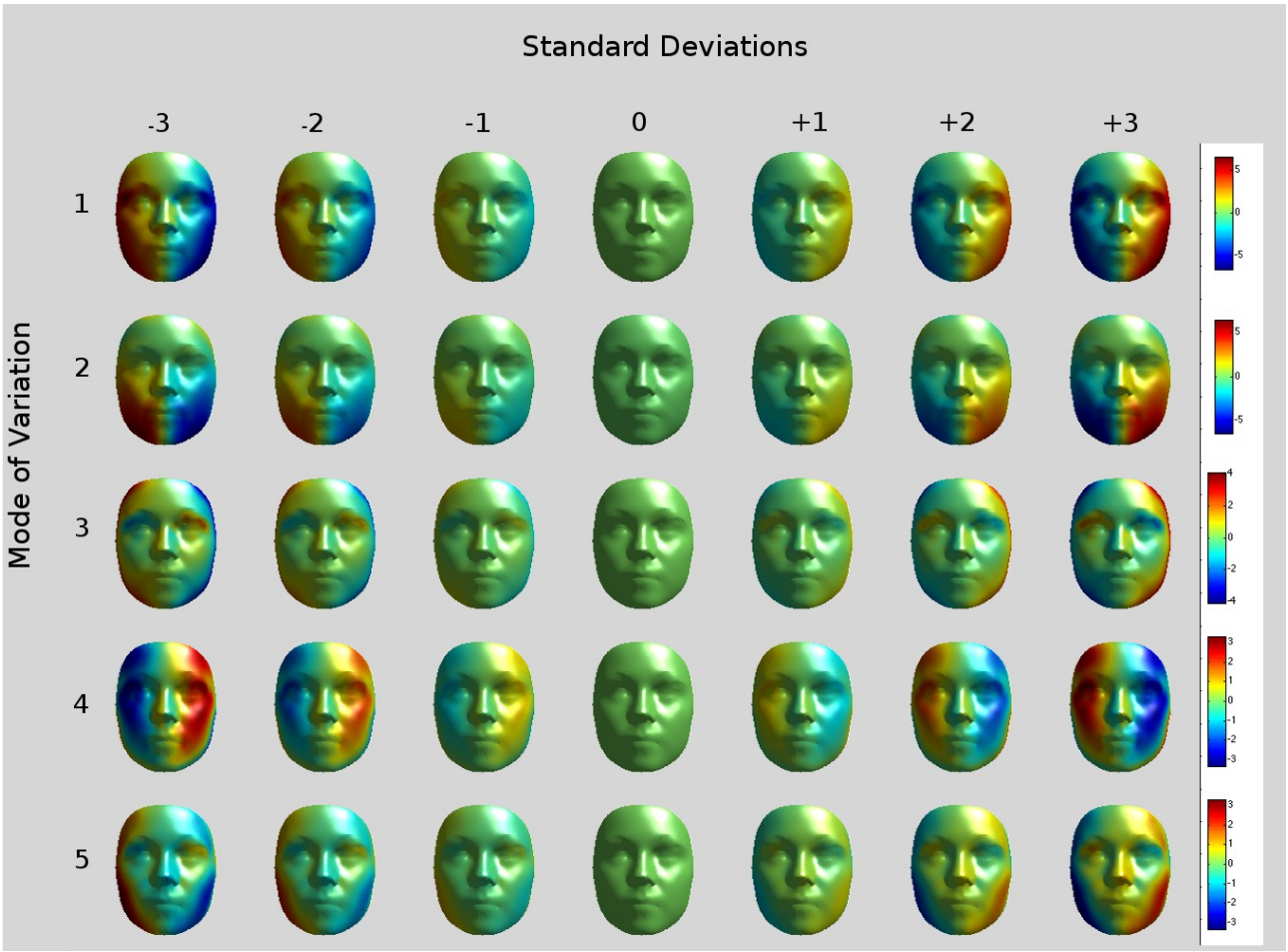

Fig 3.11: Heat maps for the modes of variation. This plot shows the heat maps for the horizontal axis: red shows movement towards the right, blue shows movement towards the left.

Mode 2

Mode 2 stretches and compresses the face along the y axis, and accounts for 13.52% of facial variance. For negative values, the face grows longer, but also pulls the facial features inwards so that the nose and mouth don't extrude as far from the face. As the chin is pulled further down, it also gets narrower, making it look pointy rather than round. Positive values for this mode create a rounder, smaller face. The chin and forehead are pushed towards the rest of the facial features, which pushes the nose and mouth further out , and makes the chin look rounder.

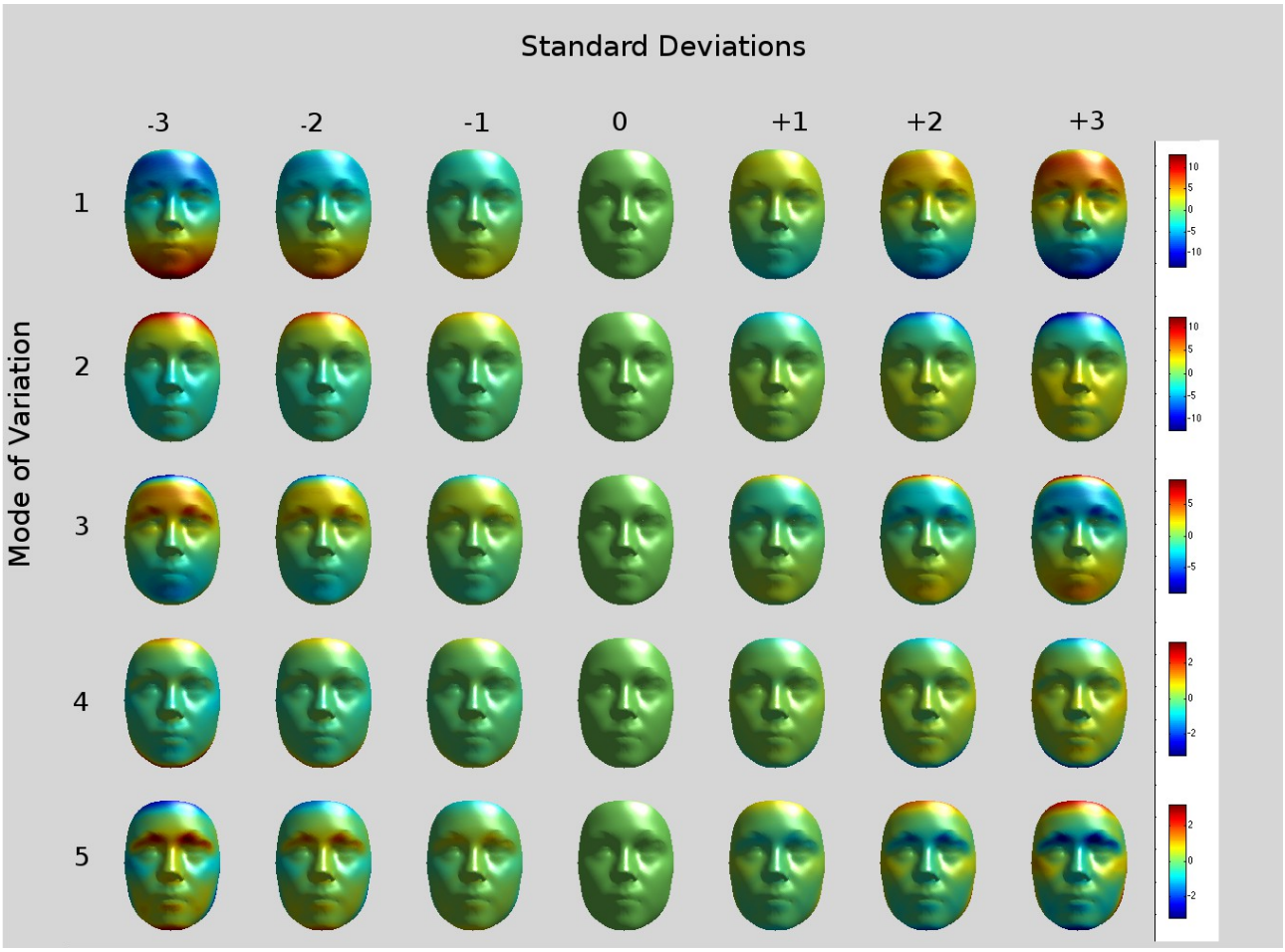

Fig 3.12: Heat maps for the modes of variation. This plot shows the heat maps for the vertical axis: red shows movements upwards, blue shows movement downwards.

Mode 3

Mode 3 represents the shape of the brow and the prominence of the chin, and accounts for 11.27% of variance. When supplied positive values, the brow moves further down the face, and expands outwards alongside the rest of the forehead. The chin also protrudes further from the face, but moves upwards towards the mouth. The eye sockets also become slightly more sunken in comparison to the rest of the face. For negative values, the chin retreats in towards the face, but the brow moves up.

Mode 4

Mode 4 correlates with the position of all features on the face, and the length and prominence of the chin. Positive values show eyes, nose and mouth migrating up the face as the chin gets larger, whilst the forehead contracts. Negative values show the chin contracting and the forehead growing. This mode represents 6.52% of variance.

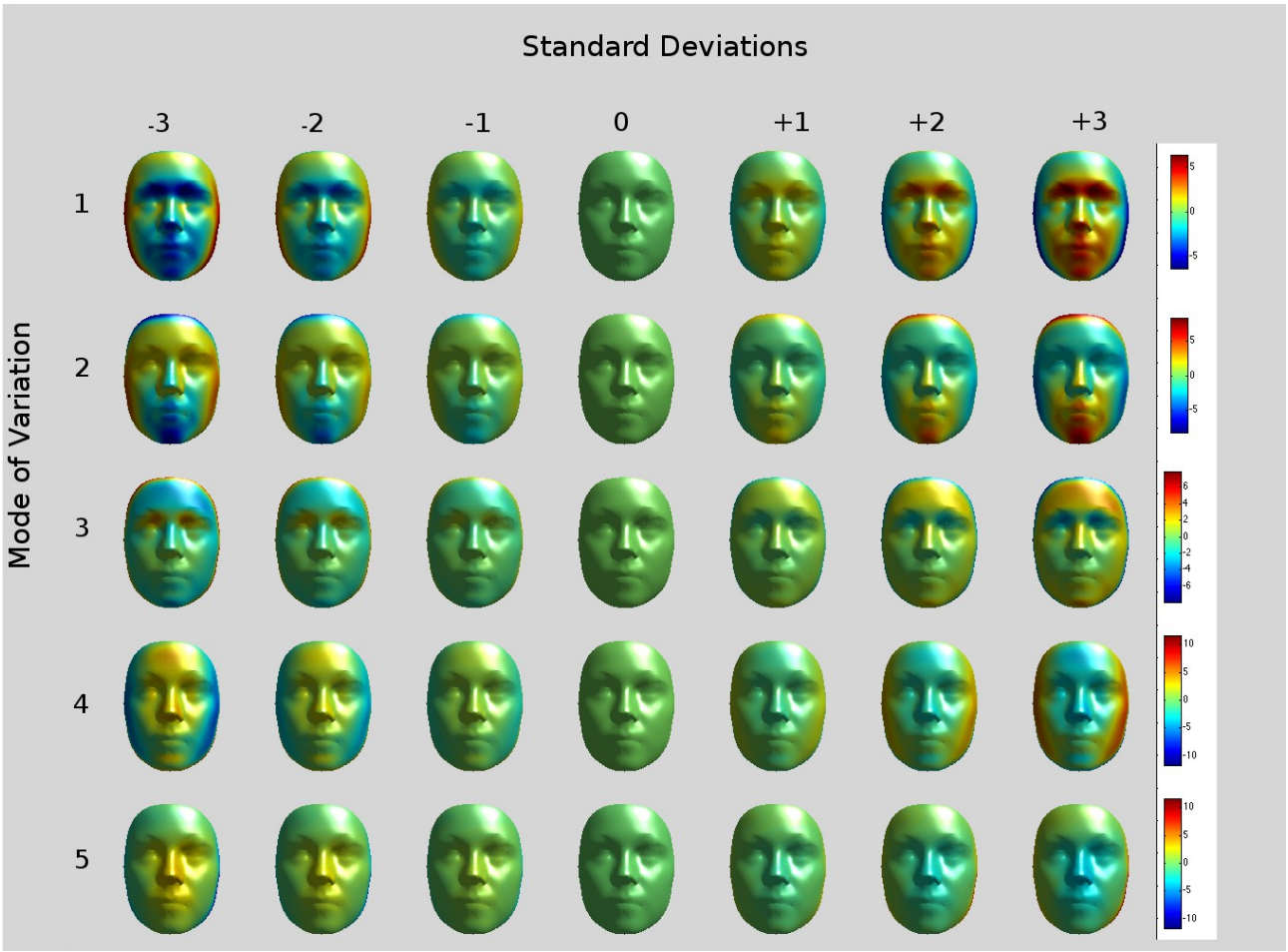

Fig 3.13: Heat maps for the modes of variation. This plot shows the heat maps for the depth axis: red shows movements towards the screen, blue shows movement away from the screen.

Mode 5

Mode 5 represents the curvature of the facial mesh, and accounts for 4.73% of the variance in the model. The nose, eyes and mouth can be observed to extend further out of the face for positive values, as if the face is a mask being squeezed from the sides. Negative values show the face flattening as if it's a mask with the sides being pulled.

Mode 6

Mode 6 represents the shape of the forehead, accounting for 4.01% of variance. Positive values increase the height and width of the forehead, and lower the points just above the eyes. The forehead also extends outward relative to the eye sockets and nose area. Negative values show the opposite effect, with the forehead becoming less prominent.

Mode 7

Mode 7 clearly represents the shape of the cheeks, and the shape of the lips. Positive values for this mode lead to a thinner, more gaunt face. The lips also appear to protrude further from the face. Negative values lead to a rounder face with much fuller cheeks, and the lips appear to be flatter. 3.05% of variance is represented in this mode.

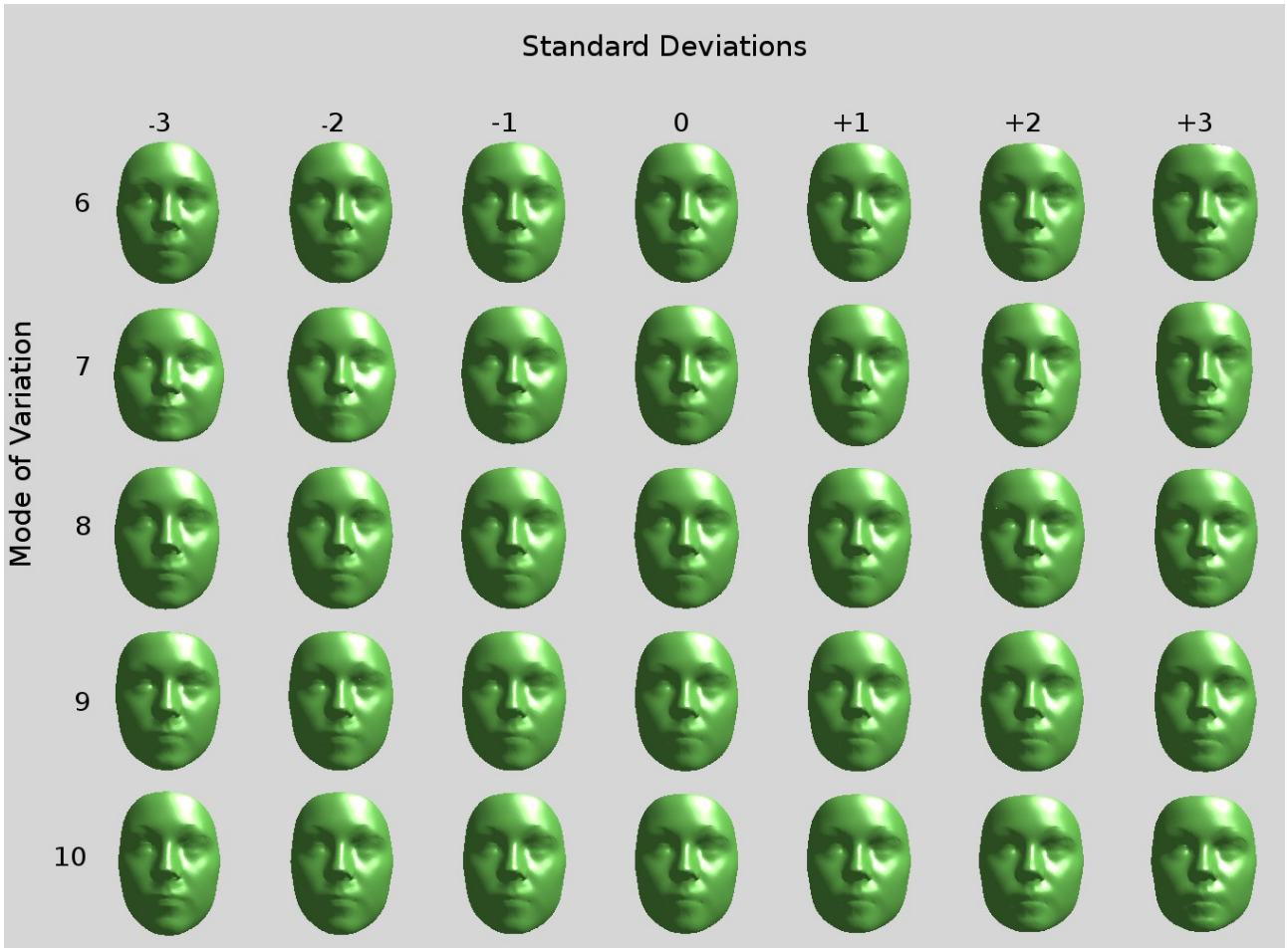

Fig 3.14: A plot showing PC6 to PC10.

Mode 8

Mode 8 is more subtle, but it appears to dictate the lateral position of the features on the face, and accounts for 2.38% of variance. The heat maps suggest that it also has an effect on the shape of the mouth, and also the point at which the nose ends and the forehead begins. Positive values appear to lower the bottom lip, and lower the top end of the nose. Negative values raise the lower lip, and raise the top end of the nose.

Mode 9

Mode 9 represents the protrusion of the nose and the angle of the nostrils. Positive values result in a flatter face with a more upturned nose, whereas negative values represent a face where the nose's tip is lower with more horizontal nostrils. Positive values also raise the outer eyebrows up, whereas negative values lower them. 2.33% of variance is represented in this mode.

Mode 10

Mode 10 represents the width of the eye sockets, the position of the the lower lip, and the squareness of the jaw. Positive values appear to make the eye sockets slightly more squashed,

and the lower lip takes a lower position. Squareness of the jaw also increases. Negative values stretch out the eye sockets, and raise the lower lip. 2.06% of the total variance is represented in this mode.

Comparison to 110 Mesh Test

When analysing the heat maps, PC1 and PC2 for both match up to their counterparts in the 110 mesh test, which isn't surprising given that in both cases, they account for general size of the face. PC 7 is similar to the 110 test's PC5, and PC9 is similar to the 110 model's PC3. Many modes have similar patterns, but are reversed, or are slightly different. The 110 test's modes seem less clear cut beyond PC10; this is likely because these modes represent much less variation than their counterparts in the full test.

5. Future Work

Unfortunately, I ran out of time to attempt any kind of classification. As such, creating a classifier using the model that I've made would be a good avenue for future analysis. The data I have used has already been classified, so the work would be based around training the classifier to recognise certain features based on that data. Obtaining the values of each face scan in the model would be the first step in creating any kind of classifier. Classifying the data based on thousands of variables is likely to be inefficient and difficult; having boiled the variation in the set of faces down to their key components makes classification much simpler. Having sets of values for each face within the model allows for analysis of those values. It may be the case that certain ranges values of a component or a set of components within the model correspond with a particular type of facial feature. If this is the case, then a simple check can be made to see if a face lies within that range. However, given the variability of the human face, it is unlikely that a clear-cut boundary exists. Hammond *et al* were able to identify Noonan Syndrome and VCFS in individuals with an accuracy of 95% (2004), and Suttie *et al* had success in identifying Fetal Alcohol Syndrome (2013), but neither could get 100% accuracy. Therefore, the rules for classification must be fine tuned so that accuracy as high as possible, and false positives are low.

Once the classifier had been made, constructing a UI around it would be another area to explore. Since the goal of classification is ultimately to assist humans, creating a simple UI to assist geneticists, dentists and other individuals who would be interested in the data would be of great benefit. Once the model had been created, it would be possible to port it into other languages, which would help to make the classifier more portable, as any data I have is currently reliant on Matlab for access. Implementing thin-plate spline warping, procrustes analysis and PCA would be a challenge, but are certainly achievable. Libraries for PCA already exist online in various languages, so they could be used to save time. Porting to another language could make use of the shape model faster, as well as opening up the possibility of use on tablets and smartphones.

In addition to this, another avenue could be investigating alternative functions for performing non-rigid registration. Thin-plate spline warping works well, but are there other methods that would work well? Models could be created for various functions and, in conjunction with a working classifier, the accuracy of each of these methods could be tested. Functions that successfully classify features at a higher rate of accuracy would be preferable to a lower rate.

Creation of a shape model using lower resolution meshes would be another interesting avenue

to explore. Zollhöfer *et al* describe a process to capture a 3D model from a Kinect camera (2011); much larger sample sets could be obtained if subjects could create scans in their own home, rather than visiting a professional. Shape models could be used to simulate the appearance of a subject's face in VoIP calls over the web, resulting in a face with more detailed expressions, and less bandwidth required for smooth movement.

6. Conclusions

In all, creating a model using facial meshes as well as landmarks is a great improvement over use of landmark data alone. The human face contains a great amount of detail that can be hard to account for when marking landmarks, which can be captured with 3D face scans. The model can effectively create examples of new human faces, as well describe existing faces, and in much greater detail; the mesh model has roughly 10 times the number of modes of variation compared to the landmark model. The model also represents several gigabytes of data in under 200MB, once 5% of the principal components have been culled. This is much cheaper computationally and in terms of space needed. Thin-plate spline warping is an effective technique for non-rigid registration, as meaningful correspondences were successfully found to create the model.

For these purposes, the model meets the goals set at the beginning of the project. Generating realistic 3D faces from a model could have a variety of applications where a detailed, computer generated face is required, such as in film or video games.

However, what remains to be seen is the model's usefulness in classification. It is possible that the model doesn't represent very subtle differences in facial shape especially well. Investigation is required to evaluate the model's usefulness in classification. Firstly, can a correlation be found between particular modes of variation and particular facial feature types? If so, then a classifier should be created, and its accuracy assessed. If the classifier is inaccurate, or a correlation cannot be found, then other warping functions should be investigated. Would other warping functions be better suited to preserving more subtle features? Answering these questions would be the next step in evaluating the usefulness of the model.

7. Reflection

In the original project plan, I assigned time to research the more difficult aspects of the overall approach, such as Procrustes Analysis and Thin-plate spline warps. I used the time to read about these techniques, and thought I more or less understood what they were. The problem was that when the time came to put them into practice, I didn't understand *how* to use them. This meant that I spent a great deal of time getting my head around exactly what these techniques would do to my data specifically. When the time came to write my interim report, I found myself spending even more time doing research, as I didn't fully understand how each process fit together to get the end result. The fact that the research time at the beginning had not been spent well meant that I would be using up even more time later on. If I could do the project over again, I would spend much more time reading about similar approaches, so that I had a better idea about context. At first, it is not a high priority to understand exactly what a technique does and how it does it; understanding what it does more generally is more important, and why it is needed in the greater context of the project. Any new techniques learned can then be practised with a specific idea of what kind of output is required.

Understanding each part in the context of the whole helps to make everything clearer. In future projects, I will endeavour to seek out examples of similar projects, and to first learn what the key issues are. Knowing what may cause problems, and generally how to solve those problems will assist greatly when doing more detailed research.

I also found that performing research without any way to put new knowledge into practice was unhelpful. I would often do my research at home, reading papers in the evening, only to find that the knowledge hadn't sunk in the next day. Matlab has a large number of visual examples of its various functions, and looking for example implementations online also provides an insight into how they work. In future, I plan to be testing while I read, so that I can associate ideas in writing with actions that I have undertaken myself.

Keeping regular backups became an issue during the implementation of the project. My lab computer became unusable when I needed to work on putting together the various functions I had created into a coherent whole. I had been taking backups throughout, but these were not frequent enough to allow me to continue working on another computer entirely unhindered. Thankfully my lab computer was up and running again within a week, with no loss of data. The incident taught me a valuable lesson in the dangers of keeping all my work in one place, so I immediately made copies of the data, and I made sure that any new work I had done would be saved in multiple locations. This is a practice I will continue for future projects, as I may not be as lucky next time my computer fails.

Overall, the project has been a challenging experience, but I have learned many valuable lessons in the process.

8. References

Bookstein, FL. 1991. *Morphometric tools for landmark data*, Cambridge University Press

Bookstein, FL. 1989. Principal Warps: Thin-Plate Splines and the Decomposition of Deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 11 (6), 567-585.

Bookstein, FL. 1997. Shape and information in medical images: A decade of the morphometric synthesis. Computer Vision and Image Understanding. 66, 97–118.

Cliff, N. 1988. The eigenvalues-greater-than-one rule and the reliability of components. *Psychological Bulletin.* 103, 276 – 279.

Cootes, TF & Taylor, CJ. 2004. Statistical models of appearance for computer vision. Technical report, Dept of Imaging Science and Biomedical Engineering, University of Manchester

Gower, JC. 1975. Generalized procrustes analysis. *Psychometrika.* 40 (1), 33-51.

Guttman, L. 1954. Some necessary conditions for common factor analysis. *Psychometrika.* 19, 149 – 161.

Hammond P, Hutton TJ, Allanson JE, Campbell LE, Hennekam RC, Holden S, Patton MA, Shaw A, Temple IK, Trotter M, Murphy KC & Winter RM. 2004. 3D Analysis of Facial Morphology. *American Journal of Medical Genetics*. 126 (A), 339-348.

Hutton, TJ, Buxton, BF, Hammond, P. 2001. Dense Surface Point Distribution Models of the Human Face. *Proceedings IEEE Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA 2001).* 153-160.

Hutton, TJ, Buxton, BF, Hammond, P, Potts, HWW. 2003. Estimating Average Growth Trajectories in Shape-Space Using Kernel Smoothing. *IEEE Transactions on Medical Imaging*. 22 (6), 747-753.

Jackson, DA. 1993. Stopping rules in principal components analysis: a comparison of heuristical and statistical approaches. *Ecology.* 74, 2204 – 2214

Jolliffe, IT, 2002. *Principal Component Analysis*. 2nd ed. New York: Springer.

Polly, PD. 2012. *Procrustes, PCA, and 3D coordinates*, Department of Geological Sciences, Indiana University

Salvi, J, Matabosch, C, Fofi, D, Forest, J. 2007. A review of recent range image registration methods with accuracy evaluation. *Image and Vision Computing*. 25, 578-596.

Smith, LI. 2002. *A tutorial on Principal Components Analysis*. University of Otago. www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

Suttie, M, Foroud, T, Wetherill, L, Jacobson, JL, Molteno, CD, Meintjes, EM, Hoyme, HE, Khaole, N, Robinson, LK, Riley, EP, Jacobson, SW, Hammond, P. 2013. Facial dysmorphism across the fetal alcohol spectrum. *Pediatrics*. 134 (3), 779-788

Toma, AM, Zhurov, AI, Playle, R, Marshall, D, Rosin, PL, Richmond, S. 2012. The assessment of facial variation in 4747 British school children. *European Journal of Orthodontics*. 34 (6), 655-664.

Weisstein, Eric W. 2013. "Covariance." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/Covariance.html

Weisstein, Eric W. 2013. "Eigenvector." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/Eigenvector.html

Wilson C, Playle R, Toma A, Zhurov A, Ness A, Richmond S. 2012. The prevalence of lip vermilion morphological traits in a 15-year-old population. American Journal of Medical Genetics Part A.

Winter RM. 1996. What's in a face?. *Nature Genetics*. 12, 124-129.

Zollhöfer, M., Martinek, M., Greiner, G., Stamminger, M. and Süßmuth, J. (2011), Automatic reconstruction of personalized avatars from 3D face scans. *Computer Animation and Virtual Worlds*, 22, 195–202