Fast 3D Reconstruction and Recognition

MARCOS A. RODRIGUES, ALAN ROBINSON and WILLIE BRINK
Geometric Modelling and Pattern Recognition Group
Sheffield Hallam University, Sheffield UK,

Abstract: In this paper we discuss methods for 3D reconstruction from a single 2D image using multiple stripe line projection. The method allows 3D reconstruction in 40 milliseconds, which renders it suitable for on-line reconstruction with applications into security, manufacturing, medical engineering and entertainment industries. We start by discussing the mathematical fundamentals of 3D reconstruction and the required post-processing operations in 3D to render the models suitable for biometric applications such as noise removal, hole filling, smoothing and mesh subdivision. The incorporation of data acquired as 3D surface scans of human faces into such applications present particular challenges concerning identification and modelling of features of interest. The challenge is to accurately and consistently find predefined features in 3D such as the position of the eyes and the tip of the nose for instance. A method is presented with recognition rates up to 97% and a preliminary sensitivity analysis is carried out concerning reconstructed and subdivided models.

Key–Words: 3D reconstruction, 3D modelling, 3D measurement, 3D face recognition, 3D animation

1 Introduction

Within our research group we have developed methods for fast 3D reconstruction using line projection (e.g. [9], [2]). The method is based on projecting a pattern of lines on the target surface and processing the captured 2D image from a single shot into a point cloud of vertices in 3D space. Once a point cloud is obtained, it is then triangulated and relevant feature points on the surface model can be detected for recognition. It is a conceptually simple process but one that offers a number of interesting challenges. First, a method to determine the exact correspondences between projected space and camera space needs to be determined. Second, a number of post-processing operations are required such as noise removal, hole filling, smoothing and mesh subdivision. Third, for object recognition it is necessary to determine consistent characteristics from the 3D surface; this is normally defined in terms of feature points but can also be defined in terms of areas or regions. Fourth, 3D models must be aligned to a standard pose which also takes care of scale for robust recognition. Fifth, if the method is to be applied to other areas such as animation it is necessary to define methods to achieve continuity when the 3D model is recorded over the fourth dimension of time. These challenges are faced variously in fields such as biometric face recognition, industrial inspection, reverse engineering and media applications among others.

This paper by no means describes the depth and breadth of our research; instead it is our intention only to highlight some of the approaches taken to those challenges. We argue that fast 3D reconstruction has the intrinsic advantage of speed and present the mathematical foundations to achieve this. Some recognition algorithms may depend on the density of the mesh model and we present methods to increase density allowing a wider range of algorithms to be used that depend on sampling regions of the mesh. Equally, higher density meshes can improve recognition methods that are based on a small set of feature points as these can be more precisely determined. We also present a method to align a mesh to a standard pose such that recognition can proceed from a robust and consistent set of feature points. We present some preliminary sensitivity analysis on recognition results and highlight their application into a biometric context.

The paper is structured as follows. In Section 2 we introduce our multiple stripe method for fast 3D reconstruction. In Section 3 we highlight methods for mesh post-processing and pose alignment. In Section 4 we describe our approach to 3D face recognition. Finally, Section 5 presents a conclusion and highlights areas of future work.

2 Fast 3D Reconstruction

Our research into 3D scanning has developed a novel uncoded structured light method [9], which projects a pattern of evenly-spaced white stripes onto the sub-

ject, and records the deformation of the stripes in a video camera placed in a fixed geometric relationship to the stripe projector. A camera and projector configuration is depicted in Fig 1.

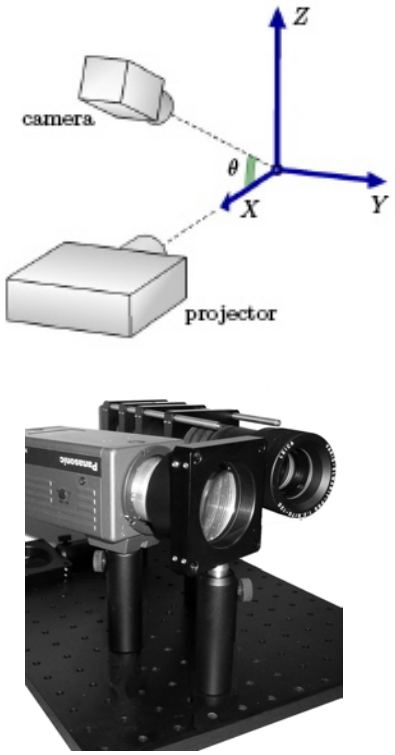

Fig 1: Top: The projector and camera axes meet at the calibration plane which defines the origin of the coordinate system. Bottom: a physical realisation of the design.

A detail from a video frame is depicted in Fig 2 (top) clearly showing the deformed stripes. Our research has successfully tackled the *indexing problem* which is to find the corresponding stripe indices in both image and projector spaces. Even for continuous surfaces the problem can be severe as small discontinuities in the object can give rise to un-resolvable ambiguities in 3D reconstruction. When there are large discontinuities over the object as shown in Fig 2 (bottom, points a, b and c belong to the same stripe) and these are distributed at many places the problem is particularly severe.

Despite such difficulties, the advantage of this over stereo vision methods [7] is that the stripe pattern provides an explicitly connected mesh of vertices, so that the polyhedral surface can be rendered without the need for surface reconstruction algorithms. Also, a smoothly undulating and featureless surface can be more easily measured by structured light than by stereo vision methods. These advantages for single frame scanning are even more important for 4D applications such as animation and immersive game playing.

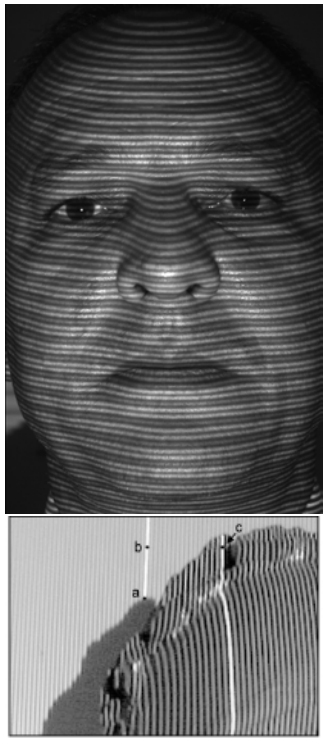

Fig 2: Top, detail from a bitmap, showing the stripes deforming across the face. Bottom, the indexing problem due to large discontinuities.

2.1 Mapping image space to system space

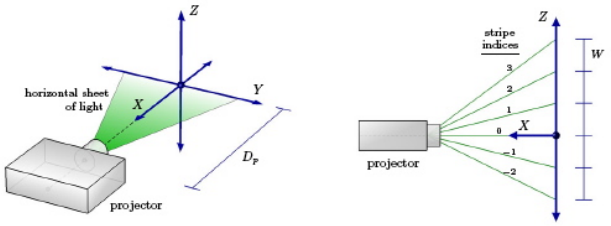

Fig 3:The coordinate system is defined in relation to the projector.

We define a Cartesian coordinate system here called system space in relation to the projector. In Fig 3 left, the axes are chosen such that: the X-axis coincides with the central projector axis; the XY plane coincides with the horizontal sheet of light cast by the projector; and the system origin is at a known distance D_p from the centre of the projector lens. Each projected stripe lies in a specific plane that originates

from the projector, and the position and shape of a certain stripe in such a plane depend on the surface it hits. Fig 3 right shows the arrangement of these planes as viewed down the Y-axis. They are all parallel to the Y-axis and their intersections with the YZ plane are evenly spaced. To discriminate between the planes we assign successive indices as shown. The horizontal plane containing the projection axis has an index of 0. The distance W between two consecutive stripes on the YZ plane can be measured and enables us, for example, to write the point of intersection between the Z-axis and a plane with index n as $(0, 0, W_n)$ in system coordinates. The position of the camera is fixed in system space with the following constraints: (a) the central camera axis points to the origin of system space; (b) the centre of the camera lens is at point $(D_p, 0, D_s)$ in system space, with $Ds > 0$, where D_s is the distance between camera and projector axes.

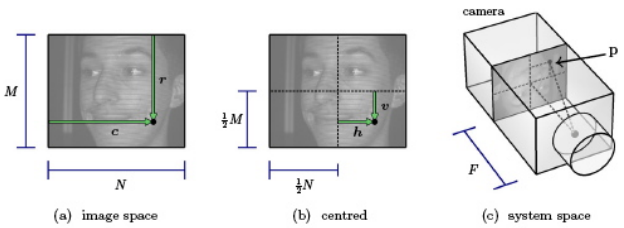

Fig 4: the position of a pixel in the image at row r and column c is transformed to coordinates (v, h) and then to a point $\mathbf{p}$ in system space.

An image is recorded in the sensor plane of the camera located behind the lens perpendicular to the camera axis as depicted in Fig 4. The transformation is given as

$$v = r - \frac{1}{2}(M + 1) \quad \text{and} \quad h = c - \frac{1}{2}(N + 1) \quad (1)$$

Since the focal point of the camera is located at $(D_p, 0, D_s)$ the centre of the sensor plane is located at $c = (D_p + F, 0, D_s)$ in system space. Here F is the focal length of the camera, i.e. the distance from the focal point of the camera to the sensor plane, as shown in Fig 4. Assuming that each pixel on the sensor plane is a square of size $PF \times PF$, we can write the coordinates of point $\mathbf{p}$ as

$$\mathbf{p} = c + (0, -hPF, vPF). \quad (2)$$

We have shown [9] that mapping a pixel (v, h) on stripe n to its corresponding surface point (x, y, z) can be written as:

$$x = \frac{D_c vP + Wn(\cos\theta - vP\sin\theta)}{vP\cos\theta + \sin\theta + \frac{W_n}{D_p}(\cos\theta - vP\sin\theta)} \quad (3)$$

$$z = W_n(1 - \frac{x}{D_p}) \quad (4)$$

$$y = hP(D_c - x\cos\theta - z\sin\theta). \quad (5)$$

Here D_c measures the distance from the focal point of the camera to the origin. A considerable challenge is to map the stripe index n from projector to camera space as pictured in Fig 5 below.

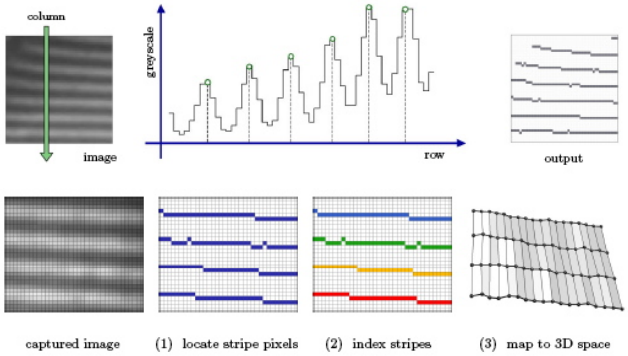

Fig 5: Detecting and mapping stripe indices from projector to camera space. Different colours mean different indices.

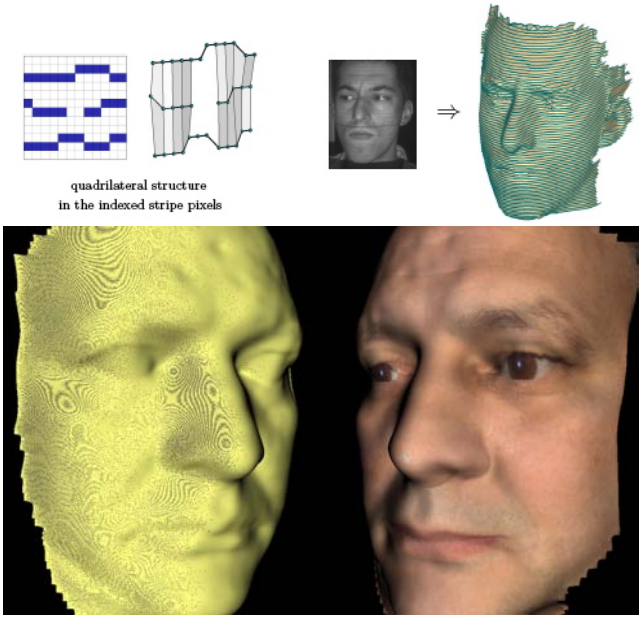

Fig 6: Top: estimating the point cloud from equations 3–5 and building a triangulated mesh with texture mapping from the stripe colour scheme. Bottom: texturing with a colour bitmap.

We have developed a number of successful algorithms to deal with the mapping as described in [9] and [2]. Once this mapping is achieved, a 3D point cloud is

calculated from equations 3–5 and the output is triangulated using the connectivity of the vertices as depicted in Fig 6. Once the surface shape has been modelled as a polygonal mesh, a texture image can be overlaid over the mesh either by using the same striped image or an identical image possibly taken by a second camera. The final model therefore contains the (*x, y, z*) coordinates and their corresponding RGB (red, green, blue) values for each vertex, and the face can be visualised as in Fig 6. This shows the model bitmapped with a sequence of stripe colours, and with a colour bitmap.

We can process point clouds at different resolutions by processing white and dark stripes followed by a sub-division scheme to increase mesh density without undue loss of accuracy. The resolution depends on how close together we can pack stripes in the vertical direction, while in the horizontal direction we can process one vertex per pixel.

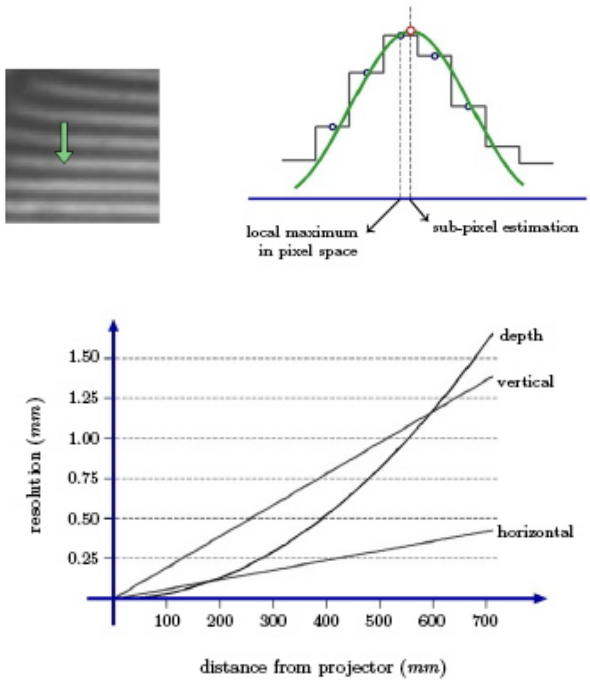

Fig 7: Top: sub-pixel estimator to determine the "true" position of the centre of the stripe. Bottom: resolution as a function of the distance to the projector.

In an effort to improve the depth resolution we have incorporated a sub-pixel estimator. Following [4] and others we assume that the spread of luminance values across a recorded stripe in one column conforms approximately to a Gaussian distribution. We fit such a profile to a neighbourhood around an identified stripe pixel, and the "true" position of the stripe in the continuous pixel space is then chosen as the maximum. Fig 7 illustrates the basic principle on a small region of a typical recorded image and the achieved resolution in depth as a function of the distance to the projector.

3 Post-Processing & Pose Alignment

We now turn our attention to some of the required post-processsing operations on the point cloud. These are appropriate for face processing and relate to the removal of noise, hole filling, mesh sub-division, and mesh alignment for recognition. The first operation on the mesh is to deal with holes in the structure. We use a bilinear interpolation by navigating the data structure through following each stripe index in the model. Once holes are filled in, we apply a Gaussian smoothing algorithm which has the effect of removing most noise – given that the face is a smooth ondulated structure. The area of the eyes represent a particular problem and a good solution for this is to punch an elliptical hole on the position of the eyes (which can be recovered from texture mapping). Once the holes are in place, we then apply again a bilinear interpolation and the final result is the model at the bottom of Fig 7.

The remaining aspect that we would like to comment on relates to sub-division to increase mesh density. First, a point cloud is generated by processing the available stripes. Due to the stripe patterns, the data has a convenient structure to perform all operations of hole filling, smoothing, and sub-division. The sub-division algorithms operate along both directions across and along stripes by using a 4th order polynomial. This process can be repeated indefinitely up to computer memory constraints, so in principle a mesh can have any desired density.

Figure 8 highlights some aspects of subdivision. On the left we processed only the white stripes leading to a sparse mesh density. On the right, first we have doubled the mesh density by processing both white and dark stripes then double again by applying one step of subdivision across the stripes. Effectively the model on the right has three extra stripes in between each original stripe of the left model. It can be noted that it captures more details on the face which can lead to a more robust recognition.

Our method to extract features from a face model for recognition is based on cutting oriented planes from defined feature points and detecting all points on the mesh that intercept those planes. We thus, require pose alignment but this may not be necessary if alter-

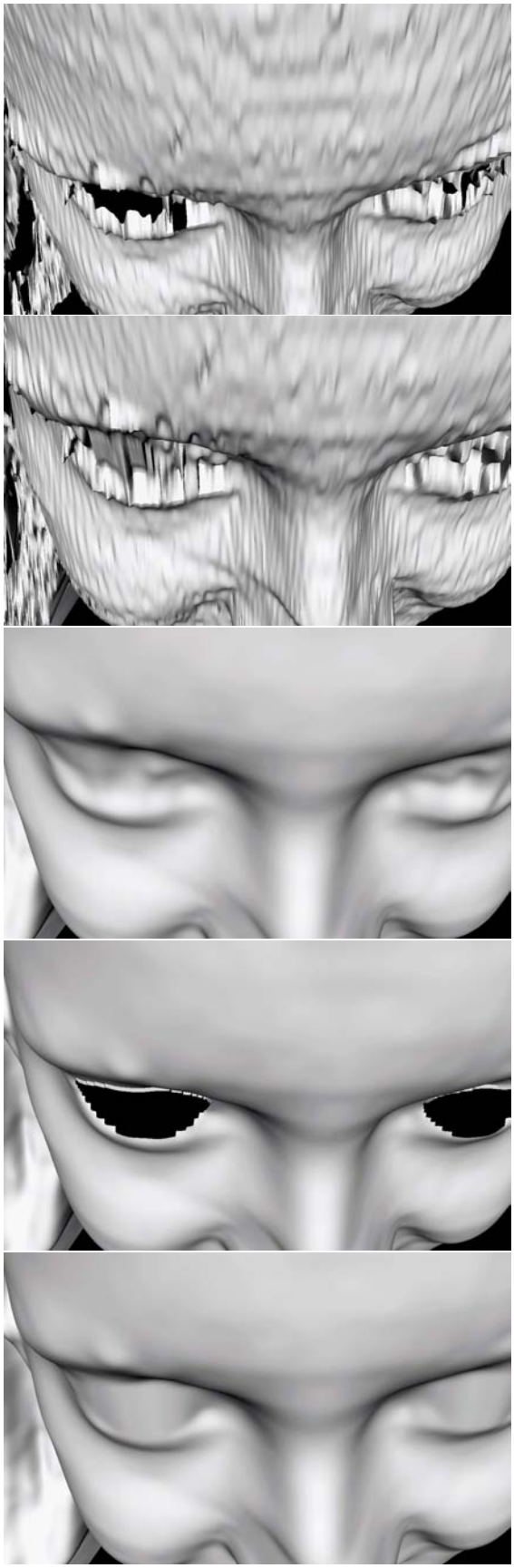

Fig 8: The post-processing pipeline from the top: noisy model with holes, hole filling, smoothing, punching eye holes, filled eye holes.

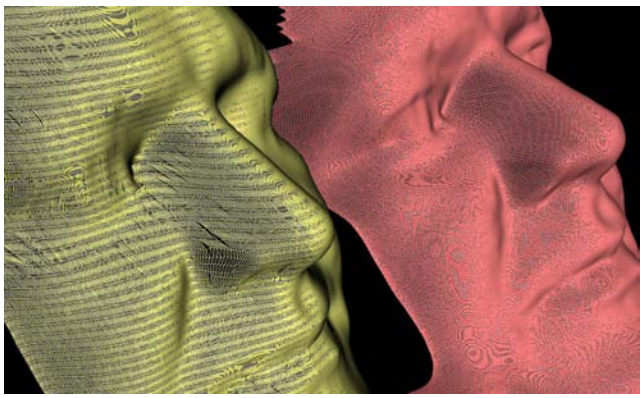

Fig 9: The effects of processing only white stripes (left) and by processing both white and dark followed by one subdivision step across stripes.

native feature extraction methods are developed. We start by detecting face and eyes in 2D through Haar transforms so the exact eye positions in 2D can be precisely known. We use the Intel OpenCV libraries for this task [5]. The assumptions thus are that the 2D eye positions are known and that the face is upright. The following iterative method will transform the face to its standard pose:

1. determine the 3D position of the eyes through their 2D texture positions
2. search for a point in 3D on the front above eye positions located at 0.75 times the distance between the two eyes
3. determine the highest point in 3D below eye positions, this is an approximation of the tip of the nose
4. translate the mesh to the tip of the nose and rotate the mesh to a constant angle between the X-axis and the point on the front
5. repeat steps 3 and 4 until the position of the tip of the nose moves less than a set threshold.

This method has proved to work successfully even if the subject is not directly facing the camera, it has been tested on images facing over 45 degrees to either side so in those cases the 3D model tends to be more of a profile model.

Figure 10 shows two models of two different subjects in their automatically normalised pose. Note that the tips of the nose coincide and that the angle between the X axis (yellow) and the point on the front are the same for the two models. The trasnparent green model allows us to inspect the area around the mouth of the underneath pink model.

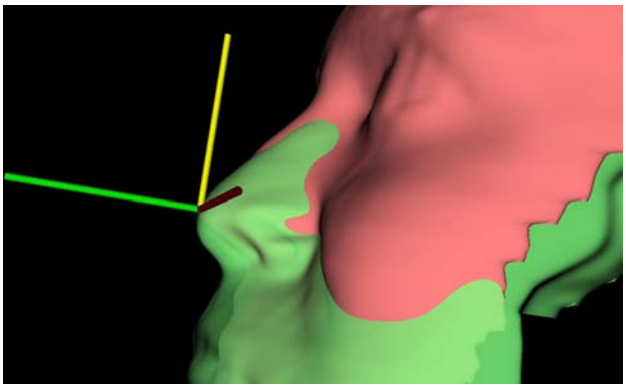

Fig 10: Pose normalisation: centred at the tip of the nose with a constant angle between X-axis (yellow) and a point between the two eye positions.

4 3D Recognition

The theoretical and practical issues related to robust 3D face recognition can be translated to a number of other applications. Much research has been undertaken in the area of 2D face recognition [8], [1] while 3D is incipient. It is often said that 3D face recognition has the potential for greater accuracy than 2D techniques, as 3D is invariant to pose and illumination and incorporates important face descriptors embedded within the 3D features [1]. The challenges to improved 3D face recognition for real-time applications reflect the shortcomings of current methods:

1. the need for fast and accurate 3D sensor technology,
2. improved algorithms to take into consideration variations in size and facial expression, and
3. improved methodology and datasets allowing algorithms to be tested on large databases, thus removing bias from comparative analyses.

Many approaches to 3D face recognition are based on 2.5D (e.g. [6] and references therein) and most try to compare scans through registration such as ICP (iterative closest point estimation) and their variants [10]. Performing recognition by comparing 3D scans through registration in this way becomes impractical for many reasons. First, there are too many data points leading to exponential time consuming algorithms and this can only work if one is to search a relatively small database. Second, there is a practical constraint on registration, as it works best when models are acquired with scanning devices with the same characteristics. There is also an issue of defining what is a match in terms of global error between two surfaces and, perhaps equally important, which exactly are the data points being used to define a match.

This leads us naturally to feature point extraction as the most likely solution to 3D recognition. The problem of 3D recognition can thus be stated as:

1. Define a set of stable measures $m_i (i = 1, 2, \ldots, n)$ on a 3D scan and build a vector $M = (m_1, m_2, \ldots, m_n)^T$ that uniquely characterises a given face
2. Build a matrix Ω of vectors M where the index of M points to the identity of the scanned object: $\Omega = (M_1, M_2, \ldots, M_s)^T$ where s is the total number of scans in the database
3. Define a method to identify a given scanned vector M with the most similar vector in the database (e.g. Principal Components Analysis).

In order to apply this method we start with pose normalisation. Once pose is normalised we already know a set of initial points namely the position of the tip of the nose, eyes, and point on the front. We then define a set of 43 points located in planes parallel to the axes at the tip of the nose and within geometric relationships defined by the set of initial points. An example of such points is depicted in Fig 11 below. Measurements are taken from such points as distances and ratios in addition to area, volume, perimeter, and various types of diameters such as breath and length resulting in a set of 191 measurements per face model.

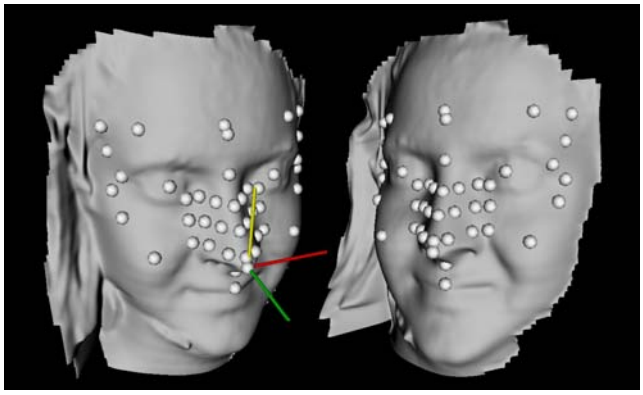

Fig 11: Automatically detected feature points on the face model.

The discussion on recognition presented here is only related to sensitivity analysis. We wanted to established whether or not we could achieve equivalent recognition rates by using a standard model processed from the stripe data (white plus dark stripes) as compared with data from a subdivided model. We collected data from 69 subjects, from each subject 6 shots were taken: frontal, smiley, looking slightly to one side (then this sequence was repeated). We used the first set for enrolment by including in the database

only the frontal and smiley models at standard and subdivided mesh densities. Thus, every subject appears in the database 4 times.

We then used the remaining set of data for testing purposes, only testing the frontal models at both densities using Principal Components Analysis. Two important observations were made that require further investigation. All subdivided models had their closest match to another subdivided model while the standard model had closest match to both. We had 2/69 mismatches for subdivided (97% accuracy) while 8/69 for standard models (88%). The accuracy obviously reflects the particular set of measurements taken and the relative influence of a subset on overall performance. Despite lower accuracy rate, the standard models cannot be discarded at this stage as it suggests that it can become more robust in the long run for large databases.

5 Conclusions

This paper has discussed methods for incorporating data acquired as 3D surface scans of human faces into biometric applications. We start by introducing our current method of fast 3D acquisition using multiple stripes which allows 3D reconstruction from a single 2D video frame. This lends the technique suitable for capturing moving objects such as a moving face in multimedia applications. We then discussed methods for noise removal, hole filling, mesh smoothing and subdivision.

Our method includes automatically eye detection in 2D and pose normalisation in 3D based on facial and scanning constraints. A method for automatically detecting feature points on a face model was presented in conjunction with feature based recognition using PCA. Sensitivity analysis was conducted on both standard and subdivided face models which calls for further investigation on the relative influence of measured parameters on recognition rates. Research is under way and results will be published in the near future.

References:

[1] K. Bowyer, K. Chang, and P. Flynn. A survey of 3d and multi-modal 3d+2d face recognition. *ICPR 2004*, pages 324–327, Cambridge 2004.

[2] W. Brink. 3D Scanning and Alignment for Biometric Systems. *PhD Thesis, Sheffield Hallam University*.

[3] M. Dong and R. Kotharib. Feature subset selection using a new definition of classifiability. *Pattern Recognition Letters*, 24:1215–1225, 2003.

[4] R. B. Fisher and D. K. Naidu. A comparison of algorithms for subpixel peak detection, *Advances in Image Processing, Multimedia and Machine Vision*, J.L.C. Sanz (ed.), Springer-Verlag, Heidelberg, 1996.

[5] Intel Integrated Performance Primitives 5.3. *Intel Software Network*, http://www.intel.com/cd/software/products/asmo-na/eng/219967.htm

[6] X. Lu, A. Jain, and D. Colbry. Matching 2.5d face scans to 3d models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):31–43, 2006.

[7] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London*, B:301–328, 1979.

[8] T. Nagamine, T. Uemura, and I. Masuda. 3d facial image analysis for human identification. *ICPR 1992*, pages 324–327, the Netherlands 1992.

[9] A. Robinson, L. Alboul, and M. Rodrigues. Methods for indexing stripes in uncoded structured light scanning systems. *Journal of WSCG*, 12(3):371–378, February 2004.

[10] M. Rodrigues, R. Fisher, and Y. Liu. Registration and fusion of range images. *CVIU Computer Vision and Image Understanding*, 87(1-3):1–131, July 2002.

[11] M. Rodrigues, A. Robinson, L. Alboul, and W. Brink. 3d modelling and recognition. *WSEAS Transactions on Information Science and Applications*, 3(11):2118–2122, 2006.

[12] L. S. Tekumalla and E. Cohen. A hole filling algorithm for triangular meshes. *tech. rep. University of Utah*, December 2004.

[13] J. Wang and M. M. Oliveira. A hole filling strategy for reconstruction of smooth surfaces in range images. *XVI Brazilian Symposium on Computer Graphics and Image Processing*, pages 11–18, October 2003.

[14] J. Wang and M. M. Oliveira. Filling holes on locally smooth surfaces reconstructed from point clouds. *Image and Vision Computing*, 25(1):103–113, January 2007.