Author(s): Sheng, Y., Willis, P., Gonzalez Castro, G. and Ugail, H.

Conference Paper title: PDE Face: A Novel 3D Face Model

Publication year: 2008

Publication title: Proceedings of the 8th IASTED International Conference on Visualization, Imaging and Image Processing

ISBN: 978-0-88986-759-8

Publisher: ACTA Press

Publisher's site: *http://www.actapress.com*

Original online publication is available at:
*http://www.actapress.com/Content_Of_Proceeding.aspx?ProceedingID=494*

# PDE FACE: A NOVEL 3D FACE MODEL

Yun Sheng and Phil Willis
Media Technology Research Centre
Department of Computer Science
University of Bath, United Kingdom
Emails: y.sheng@bath.ac.uk

Gabriela González Castro and Hassan Ugail
EIMC, School of Informatics
University of Bradford, United Kingdom

**ABSTRACT**
We introduce a novel approach to face models, which exploits the use of Partial Differential Equations (PDE) to generate the 3D face. This addresses some common problems of existing face models. The PDE face benefits from seamless merging of surface patches by using only a relatively small number of parameters based on boundary curves. The PDE face also provides users with a great degree of freedom to individualise the 3D face by adjusting a set of facial boundary curves. Furthermore, we introduce a *uv*-mesh texture mapping method. By associating the texels of the texture map with the vertices of the *uv* mesh in the PDE face, the new texture mapping method eliminates the 3D-to-2D association routine in texture mapping. Any specific PDE face can be textured without the need for the facial expression in the texture map to match exactly that of the 3D face model.

**KEY WORDS**
Face Modelling, partial differential equations, texture mapping.

## 1. Introduction

3D computer graphics technology is advancing at a rapid pace and those model-based applications, ranging from video coding [1], immersive telecommunication [2] to the video games industry, are constantly emerging to take advantage of this sophistication and power. 3D face modelling is one of the most promising research fields, since humans are especially sensitive to communicating via facial appearance and expressions. Modelling a specific 3D face is normally implemented by integrating a face texture with a generic 3D face model. Thus, a generic 3D face model is paramount, and this model should be geometrically flexible enough to simulate any facial expressions.

Since a first parameterised face model was technically reported in [3], considerable effort has been made by researchers for answering the question of how to model a face more realistically with less degree of computational complexity. Some well-known examples are FACES (Facial Animation, Construction and Editing System), a software simulator of the human face [4] and *Waters*' vector approach to simulate the facial muscle [5]. Later, *Terzopoulos et al* proposed a three-layer face model for modeling the detailed anatomical structure and dynamics of the human face [6]. The three layers correspond to the skin, fatty tissue and muscle. Elastic spring elements connect each mesh node and each layer. Muscle forces propagate through the mesh systems to create animation. Although this model can achieve accurate realism, it is computationally expensive to simulate such multi-layer lattices. CANDIDE models have been widely exploited because of their simplicity and public availability. The original CANDIDE consists of 75 vertices and 100 triangles [7] before *Welsh* [8] upgraded the face model with 160 vertices and 238 triangles covering the entire frontal head and shoulders, as named CANDIDE-2. With the finalisation of the MPEG-4 standard, there was an emerging need of upgrading the existing CANDIDEs in order to make them compliant with the MPEG-4 Face Definition Parameters (FDPs). In light of this demand, CANDIDE-3 [9] was invented based on the original CANDIDE model. The upgrading has been done by adding details into the mouth, nose and eyes of the original model so that the vertices of CANDIDE-3 correspond to the MPEG-4 Facial Feature Points (FFPs). Moreover, the *Instituto Superior Tecnico* also defined a simply textured 3D head model [10], developed from Parke's well-known parameterised model, which has been embedded in many MPEG-4 players, and widely used by researchers [11] [12].

All the above are methods based on a relatively low number of triangles. Moreover, they are generic models, not derived from individuals, so they are less useful to model individuals. In order to model a specific face more realistically, it is common practice to scan the person's head (for example, using a Cyberware scanner [13]), and use the resulting 3D point dataset, often containing more than one million points, to generate a high quality model. This gives a better fit to the source face but it requires a heavy computational load to manipulate it. Additionally, a surface fitting technique is required to describe the geometry of the human face generated by the scanner. For example, B-splines [14] are widely used for their smoothness, compact representation, and compatibility with many existing CAD systems. Surface generation using the B-splines normally requires a combination of several surface patches, which results in difficulties producing a smooth blend between the patches, and thus requires further trimming.

Our approach differs from both of these. We used a PDE model of the face. It can be used to model a specific face, for example by direct interactive modelling or from sparse scanned data, but can be manipulated at low computational cost subsequently. The use of a PDE model means that we have a smooth 3D surface and so we do not suffer from problems associated with mesh representations and their rendering.

The PDE method is capable of parameterising complex geometry with a relatively small set of parameters. It has proven to be an effective and powerful tool in several areas, such as surface design and blending [15], complex shape modelling [16] and 3D object morphing [17]. This method adopts a boundary value approach whereby a surface is designed by defining a number of space curves. The surface is generated between these curves by solving an elliptic partial differential equation, rather than by using the polynomial surface patches of conventional methods. Like other boundary representation schemes it describes complex surfaces by means of a collection of sub-patches joined together, but it generally does so using a small number of patches. Moreover, unlike spline-based approaches, the PDE method produces surface patches that exactly meet at their mutual boundaries, without the need for complicated trimming.

Our work uses the *Bloor-Wilson* PDE (BWPDE) method [18] as a means for generating a parametric surface representation of the human face. Typical applications of the PDE method are in computer-aided design, where there is little need to animate a surface in a controlled fashion. This animation aspect of the PDE method is one we explore here. In comparison with existing face models, especially those aiming at accurately modelling the face and those directly acquired by 3D scanners, the advantages of our PDE face are:

- The PDE face produces a 3D face with a relatively small number of parameters.
- The PDE face produces seamless merging of surface patches, inherently resolving the continuity problem found with surface patches.
- The PDE face allows intuitive generation and manipulation of complex geometry.
- The developed PDE face allows users to specify the shape of the face model by adjusting the position of boundary curves of the model.

In addition, facial animation requires carefully placed texture, which remains accurately located on the face even when it animates. In this paper, we develop a texturing method based on a 2D parametric *uv* mesh of the PDE face. This is a second aspect of our contribution. Our approach offers these advantages over other face models during texturing:

- The PDE algorithm associates texels of the texture map with the vertices, in the *uv* domain, eliminating the conventional need of 3D-to-2D association in texturing, and thus simplifying the process.
- This inherent attribute of the PDE algorithm allows synthesis of the textured 3D face without the need that the facial expression in the texture map must exactly match that of the 3D face model.

We need to be clear about the terminology. Our method uses three different mesh structures. There is a 2D mesh, the *uv mesh*, used to assist PDE surface generation and also with texture mapping. There is a 3D mesh, the *face mesh*, which represents the output face. There is a 2D mesh, the *warp mesh*, which we use to assist with texture map warping.

The rest of this paper is structured as follows: Section 2 gives the theory of how to generate a PDE face. In order to generate face model using the PDE face with a texture map, a *uv*-mesh texture mapping algorithm is introduced in Section 3. Section 4 is dedicated to a simple example of PDE face generation and manipulation, using our newly-developed PDE face generator. Section 5 concludes the paper, and discusses the future work.

## 2. PDE face Generating

The BWPDE method is one of several surface generation techniques based on the use of partial differential equations. The original use for this method was as a blend generation technique [18]. Further uses comprise interactive design [16], design optimisation and applications to physical and biological systems. This section enunciates how to generate a 3D face using the BWPDE method. Section 2.1 is dedicated to an elaboration of the BWPDE method for surface design. Section 2.2 introduces our PDE face based on the BWPDE method. Section 2.3 puts forward the possibility of shape variation of the PDE face.

### 2.1 *Bloor-Wilson* PDE method for surface design

The BWPDE method treats surface design as a boundary value problem, where surfaces can be defined using a small number of parameters. The method produces a parametric surface $S(u, v)$, defined as the solution to an elliptic PDE:

$$\left( \frac{\partial^2}{\partial u^2} + a^2 \frac{\partial^2}{\partial v^2} \right)^2 S(u,v) = 0 \qquad (1)$$

where $u$ and $v$ are the parametric surface independent variables, and $a \geq 1$, is a parameter that controls the relative rates of smoothing between the $u$ and and $v$ parameter directions. The reason for using the fourth order PDE is that a lower order PDE provides no freedom to specify both $S$ and its derivatives on the boundary, which is necessary if there is to be tangent continuity between the blend and adjacent surfaces; whereas the higher orders are more time-consuming. When generating a blending surface, it is necessary to be conscious of the

fact that it must end on the pre-specified primary surfaces and meet them with the required degree of continuity. However, when generating a free-form surface one has considerable freedom with which to choose the boundary conditions in order to achieve the desired shape. The partial differential operator in Equation (1) represents a smoothing process in which the value of the function at any point on the surface is, in a certain sense, a weighted average of the surrounding values. Thus, a surface is obtained as a smooth transition between the boundary conditions.

In *Cartesian* coordinates, $S(u, v) = [x(u, v), y(u, v), z(u, v)]$. Assuming that the effective region in $uv$ space is restricted to $0 \le u \le 1$ and $0 \le v \le 2\pi$, and using the method of separation of variables, the analytic solution to Equation (1) is generally given by

$$S(u,v) = A_0(u) + \sum_{n=1}^{\infty}\left[A_n(u)\cos(nv) + B_n(u)\sin(nv)\right] \quad (2)$$

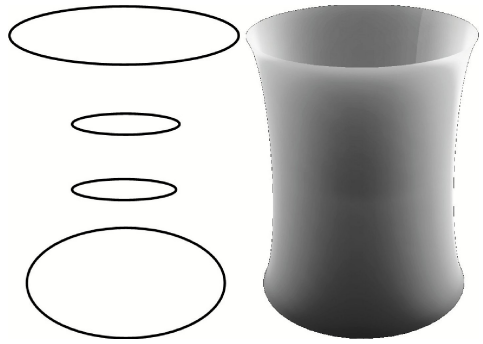
where

$$A_0(u) = \alpha_{00} + \alpha_{01}u + \alpha_{02}u^2 + \alpha_{03}u^3 \quad (3)$$

$$A_n(u) = \alpha_{n1}e^{anu} + \alpha_{n2}ue^{anu} + \alpha_{n3}e^{-anu} + \alpha_{n4}ue^{-anu} \quad (4)$$

$$B_n(u) = \beta_{n1}e^{anu} + \beta_{n2}ue^{anu} + \beta_{n3}e^{-anu} + \beta_{n4}ue^{-anu} \quad (5)$$

Coefficients $\alpha_{00}$, $\alpha_{01}$, …, $\alpha_{n3}$, $\alpha_{n4}$ and $\beta_{11}$, $\beta_{12}$,…, $\beta_{n3}$, $\beta_{n4}$ are determined by the predefined boundary conditions.



(a) (b)

Fig. 1 Example of a PDE surface generated by BWPDE method. (a) Boundary curves (b) 3D surface generated.

For a given number of boundary conditions, in order to solve the above coefficients a *Fourier* analysis is imposed. Since Equation (2) is a periodic *Fourier* series expansion for $v \in (-\infty, \infty)$, it can be approximated by the sum of a finite number of *Fourier* modes (typically $N \le 6$) and a remainder term:

$$S(u,v) = A_0(u) + \sum_{n=1}^{N}\left[A_n(u)\cos(nv) + B_n(u)\sin(nv)\right] + R(u,v) \quad (6)$$

where $R$ is defined as:

$$R(u,v) = r_1(v)e^{wu} + r_2(v)e^{-wu} + r_3(v)e^{wu} + r_4(v)e^{-wu} \quad (7)$$

where $w = a(N + 1)$, while functions $r_1(v)$, $r_2(v)$, $r_3(v)$ and $r_4(v)$ can be solved by calculating the difference between the original boundary conditions and the ones satisfied by the following finite *Fourier* series expansion:

$$F(u,v) = A_0(u) + \sum_{n=1}^{N}\left[A_n(u)\cos(nv) + B_n(u)\sin(nv)\right] \quad (8)$$

Fig. 1 shows an example of surface generation using the BWPDE method. Fig. 1 (a) illustrates the boundary curves, which are used to generate the surface showed in Fig. 1 (b). *Bloor* and *Wilson* used two of these curves as boundary conditions and the other two to determine derivatives. As a result the generated surface does not pass through all four curves. In order to make the PDE surface pass through all the boundary curves, the four boundary curves can be directly employed as the boundary conditions to the solution [17]. This is particularly helpful to modelling a face. This will be enunciated in detail in the following.

### 2.2 Generation of PDE face

There are three key factors worth noting in PDE face generation, called boundary curves, boundary conditions and $uv$ mesh.

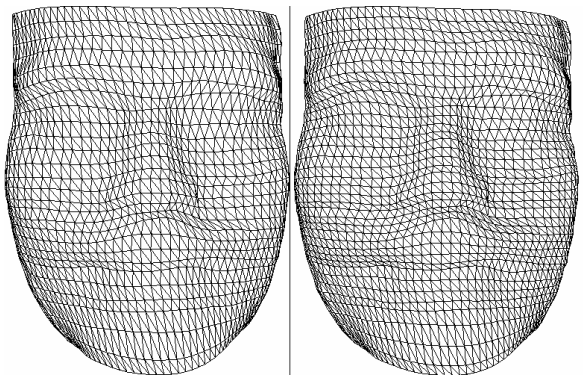
(a) (b) (c)

Fig. 2 Discrete representation of 28 boundary curves of PDE face. (a) Frontal view (b) Profile (c) 45-degree rotational view.

*Boundary curves*: To create a simple 3D surface patch using the fourth order PDE, four different boundary curves are required. Nevertheless, when modelling a more complex object like the face, four boundary curves hardly provide sufficient information to depict the face precisely. Thus, in the developed PDE face, 28 facial boundary curves acquired by scanning a human head using a 3D scanning apparatus are exploited. The face was captured with the eyes open and mouth closed in order to make the generated boundary curves most neutral and natural. The 28 facial curves cover the whole facial areas from the chin up to the forehead of the face. Each facial feature is under control of a group of facial curves. Each of these facial curves is discretely represented as a closed loop, consisting of 46 points, so that the curves can be formulated using the periodic *Fourier* series (Equation (6)), with a period of $0 \le v \le 2\pi$. The first 23 of the 46 points on each curve represent the frontal part of the face, while the other 23 correspond to reflection of the first 23 points in a given plane in order to make the curves periodic. Fig. 2 shows different views of the points for a PDE face, where each of 28 horizontal curves is defined by 46 of these points. Now surface generation becomes blending the nine different facial surface patches, each of which is generated by four consecutive boundary curves. In other words, the PDE face is produced by resolving

nine variant fourth order PDEs, each of which corresponds to one unique surface patch. Any two adjacent surface patches share one common boundary curve so that the continuity along the blended surface patches is ensured.

*Boundary conditions*: To create a 3D face, the BWPDE method introduced in the preceding subsection is utilised. As mentioned previously, the facial curves are directly substituted into the PDE method as boundary conditions in order to assure that the generated 3D face surface will pass through all the facial curves. Not only does this give the generated face a high fidelity, but some of the facial curves are also associated with facial features vital to shape variation, as well as to facial animation.



(a) (b)

Fig. 3 PDE face. (a) Using a 28 by 41 *uv* mesh (b) Using a 37 by 41 *uv* mesh.

*uv mesh*: This is the mesh samples in the parametric *uv* domain. Instead of one *uv* mesh responsible for each surface patch, only one *uv* mesh is required for the whole face, although the 3D face is generated using nine different PDEs. With the range still remaining at $0 \leq u \leq 1$, the only difference for variable $u$ is that each of the nine surface patches merely corresponds to one ninth of the whole range of $u$; while variable $v$ is set to $0 \leq v \leq \pi$, for the face side of the head. The choice of resolution of the *uv* mesh is variable, depending on the practical demand for the surface quality. The more samples in the *uv* domain, the smoother the generated surface, but the more computationally complex. This is a useful trade-off which can be exploited according to the display and specific applications. Fig. 3 illustrates neutral views of the PDE face with two arbitrary *uv* meshes. Due to the fact that we have an analytic solution expressing each PDE surface patch, the resolution of the mesh is independent of the resolution of the PDE curves. In order to achieve a balance between the surface quality and computational complexity, we used a 28-by-41 *uv* mesh in all the experiments carried out in this paper. Furthermore, the *uv* mesh plays a crucial role in *uv* mesh texture mapping that will be elaborated in Section 3.

### 2.3 Shape variation

One of the advantages of the developed PDE face is that it allows users to specify the shape of the face model by adjusting the position of the facial boundary curves. This gives users a high degree of freedom to model a specific face. Given that the PDE face is geometrically shaped by the 28 facial boundary curves, and can be manipulated to obtain the desired shape of the face, the problem becomes how to modify the initial boundary conditions to achieve various shapes of the PDE.

Studying the anatomy of the human head [4], especially the structure of the skull, points on the frontal half of the boundary curves are grouped and associated with relevant bones of the skull. This is due to the fact that change of shape of the face is mainly caused by shape change of the skull, making human beings differ person from person. For example, the points on the frontal half of the first four facial curves relating to the forehead are used to generate a flat or protuberant forehead. The results from shape variation of the PDE face will be shown in Section 4.

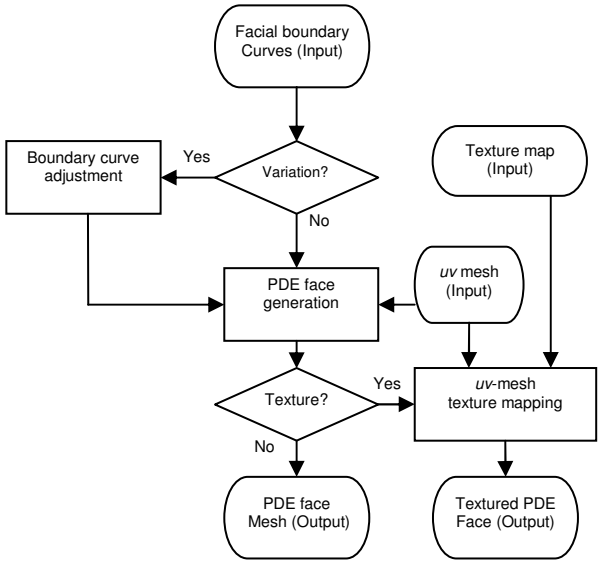



Fig. 4 Flowchart of the developed PDE face algorithm.

The flowchart in Fig.4 shows how our PDE face algorithm works. The algorithm uses a group of facial boundary curves to generate the PDE face. The boundary curves can also be adjusted to a desired shape for a specified PDE face. Texturing of the PDE face is implemented by employing a new *uv*-mesh texture mapping algorithm with a texture map combined with the *uv* mesh of the PDE face.

## 3. *uv*-mesh Texture Mapping

There are two issues we need to address. We need to apply the texture to the face. However, we also need to make sure that the texture aligns with the key features of the face. To do this we first need to warp the texture. We now describe these two processes.

### 3.1 Mesh-assisted texture map warping

As described earlier, the *uv* mesh associated with the PDE face is a rectangular parametric domain with ranges of $0 \leq u \leq 1$, $0 \leq v \leq \pi$. It determines the resolution of the generated 3D surface. In order to achieve accurate correspondence between the *uv* mesh and the texture map, the texture map has to be distorted to fit the *uv* mesh. The other way around is infeasible because the topology and coordinates of the nodes in the *uv* mesh must remain unchanged under the construction of correspondence. Therefore, it is necessary to apply an image warping algorithm ahead of texture mapping.

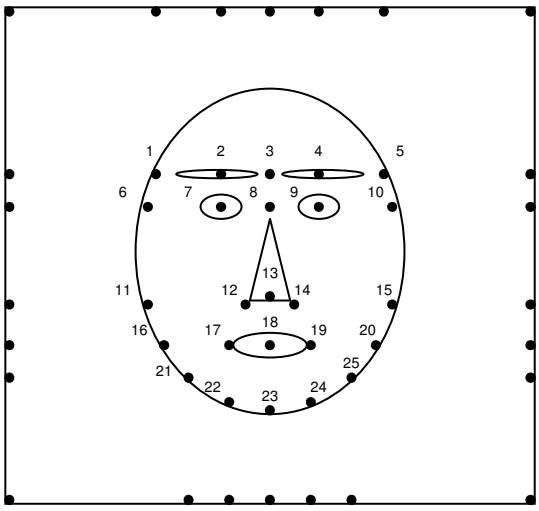



Fig. 5 Topology of warp mesh nodes:
Key feature points:

1: Extension point of right eyebrow.
2: Middle point of right eyebrow.
3: Centre of eyebrows.
4: Middle point of left eyebrow.
5: Extension point of left eyebrow
6: Extension point of right eye
7: Middle point of right eye
8: Centre of eyes.
9: Middle point of left eye.
10: Extension point of left eye.
11: Right cheek point.
12: Right nostril point.
13: Nose tip.
14: Left nostril point.
15: Left cheek point.
16: Extension of right mouth corner.
17: Right mouth corner.
18: Mouth centre.
19: Left mouth corner.
20: Extension of left mouth corner.
21, 22, 23, 24 and 25: Five points on the chin boundary (Points 22, 23 and 24 are vertically aligned with points 17, 18 and 19, respectively; points 21 and 25 should be on the perpendicular bisectors of points 16 and 17, and points 19 and 20, respectively.)

The two-pass mesh warping algorithm presented in [19] is exploited to deform the texture map. Briefly, mesh warping can be generalised as follows: Assume that $I_s$ and $I_t$, respectively, are the source and target images, *i.e.* the original and warped images, while $M_s$ and $M_t$ indicate the warp meshes associating with the source and target images, respectively. The warp meshes $M_s$ and $M_t$ are used to specify coordinates of the control points, which help warping from source image $I_s$ toward target image $I_t$, while the control points, *i.e.* where the mesh nodes are, correspond to a number of salient features. In general, $M_s$ and $M_t$ are used to define the spatial transformation that maps all points in $I_s$ onto $I_t$. The warping process here is carried out as a cascade of orthogonal 1D transformations, which is referred to as two-pass mesh warping [19].

Our method works as follows. We adopt a 7-by-7 warp mesh, whose nodes involve the 25 most salient facial feature points and 24 sub-feature points on the boundaries of the texture map. The 25 feature points, which are manually located during the warp, embrace not only the facial boundary, but the facial features, such as the centres of the eyebrows, eyes and mouth etc; while the 24 sub-feature points are in turn assigned onto the horizontal and vertical extensions of those facial feature points. The black dots in Fig. 5 illustrate all 49 mesh nodes with the numbered nodes being the 25 facial feature points, whose names are listed the figure.

We assume that the centre points of the two eyes are the most important locators for the texture, with the vertical separation of the eyes and the mouth being the next most important. We need to align the face texture with the *uv* grid which will generate the PDE surface. We scale the grid horizontally so that the two eye points fall directly on grid points. We scale the grid vertically so that the two mouth corners fall on a horizontal of the grid. This gives us an approximate fit of the texture to the *uv* space.

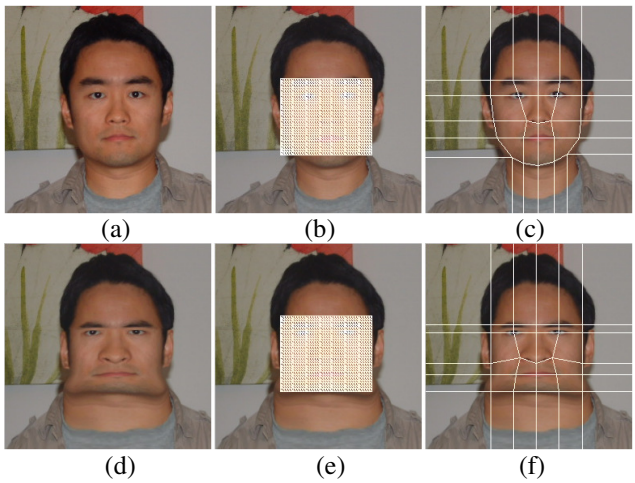



Fig. 6 Illustration of mesh-assisted texture map warping. (a) Original texture map, *i.e.* $I_s$ (b) $I_s$ with uv mesh (c) $I_s$ with $M_s$ (d) Warped texture map, *i.e.* $I_t$ (e) $I_t$ with *uv* mesh (f) $I_t$ with $M_t$.

Fig 6 (a) shows the 256-by-256-pixel source image which we wish to apply the face. Fig 6 (b) shows the 28-by-41 scaled *uv* mesh on top of the source image. Fig 6 (c) shows the source warp mesh $M_s$ formed by the unwarped feature points. The feature points in the *uv* mesh in Fig 6 (b) do not match the points in $M_s$, and this is why we need to perform a warp. Indeed it is the correspondence between points in these two structures that determines the warp.

We need to warp the face texture so that the other facial points are aligned with the *uv* mesh. These points form the warp mesh to guide the texture warping operation. Thus, in accordance with the position of their corresponding nodes in the *uv* mesh, the warp mesh $M_t$ associating with the target image for the warping process can be established.

The target warp mesh resulting from the *uv* mesh shown is that of Fig 6 (f), where we have also shown the target image. This is now correctly warped. For completeness, Fig 6 (e) shows the target image with the (now matching) uv mesh; and Fig 6(d) shows the target image alone. As can be seen, the face in the texture map has been accurately deformed to fit the *uv* mesh. Once the warp is complete, we can run over all of the 28-by-41 points of the *uv* mesh and read off the texture value for each. This is the required texture mapping.

### 3.2 *uv*-mesh texture mapping

The classical way to texture a surface in 3D is known as uv mapping [20]. This requires a mapping between the 3D coodinates $(x, y, z)$ of the surface and the 2D $(u, v)$ coordinates of the texture image. In a typical implementation suited to rendering by graphics cards, the 3D object is represented by a polygonal mesh. Each vertex records the $(u, v)$ coordinates of the texture required at that point and the rendering engine fills the polygon with texture by interpolating across the polygon the $(u, v)$ coordinates of the corresponding vertices. In this way, the texture is pinned to the polygon mesh.

Our method requires only a 2D to 2D mapping to be established. The PDE face creates correspondence directly between the *uv* mesh and the texture map, completing the mapping operation in two dimensions throughout. Eliminating the need for a 3D-to-2D association simplifies the operations. Note that this *uv* is not the same as the *uv* space of the texture, as used in the traditional method. Fig. 7 illustrates the difference. As highlighted by the upper double arrow, texture mapping is conventionally carried out between the 2D texture map and the 3D face mesh; whilst the *uv* mesh texturing method developed here performs a mapping between the 2D texture map and the 2D PDE *uv* mesh, as highlighted by the lower left arrow. Since the PDE *uv* mesh is inherently related to the generated 3D face mesh in the PDE algorithm, the texture can be pinned to the mesh.

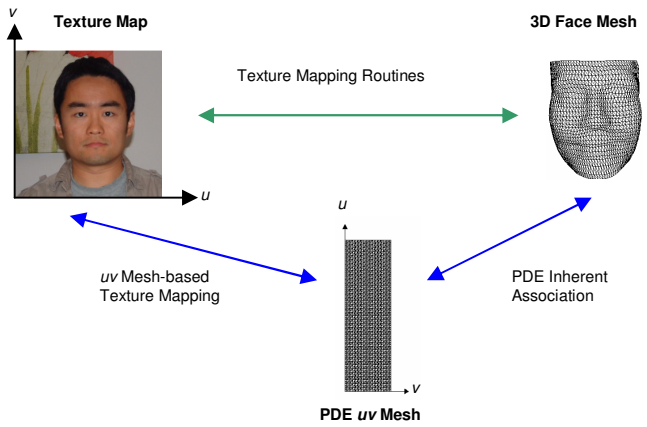



Fig. 7 Relations in texture mapping.

To help readers understand the algorithm, the whole *uv*-mesh texture mapping process is summarised as follows:
Step 1. Fit the texture approximately to the *uv* mesh by scaling the *uv* grid both vertically and horizontally to align on eyes and mouth.
Step 2. Determine the source warp mesh $M_s$ by manually selecting the facial features from the given texture map.
Step 3. Establish the target warp mesh $M_t$ from the *uv* mesh derived in Step 1.
Step 4. Apply the two-pass mesh warping algorithm to the source image, to produce the target image, which is the required texture map.
Step 5. Texture the *uv* mesh with the target image.

We now need to generate the 3D face from this textured *uv* mesh, as follows.

## 4. Example of PDE Face Generation with Shape Variation

In this section we go through a simple example of how a PDE face is generated, and how it performs under shape variations, as well as under a basic facial animation. We have designed an interface for this using C++ combined with OpenGL, called the PDE Face Generator. Fig. 8 illustrates a snapshot of our PDE Face Generator, with boundary curves of the PDE face displayed on the left side screen and the control panel on the right side margin. As implied by the control panel, the generator allows users to reshape and animate the PDE face by modifying its boundary curves. For demonstration purposes, only the shape variations on the forehead, nose and chin are implemented and shown here. For the sake of consistency, the texture map to be demonstrated in this example is Fig. 6 (d). However, other colour images in a 256-by-256 resolution, as long as they contain one human face

neutrally centred in the image without occlusions, can be input to the system as the texture map.

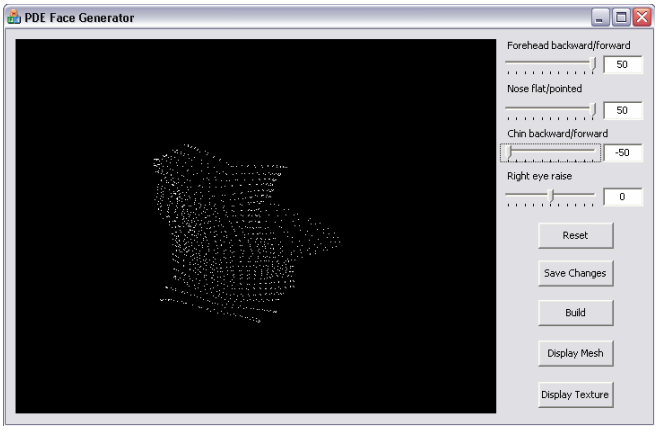



Fig. 8 Lateral view of exaggerated PDE boundary curves.

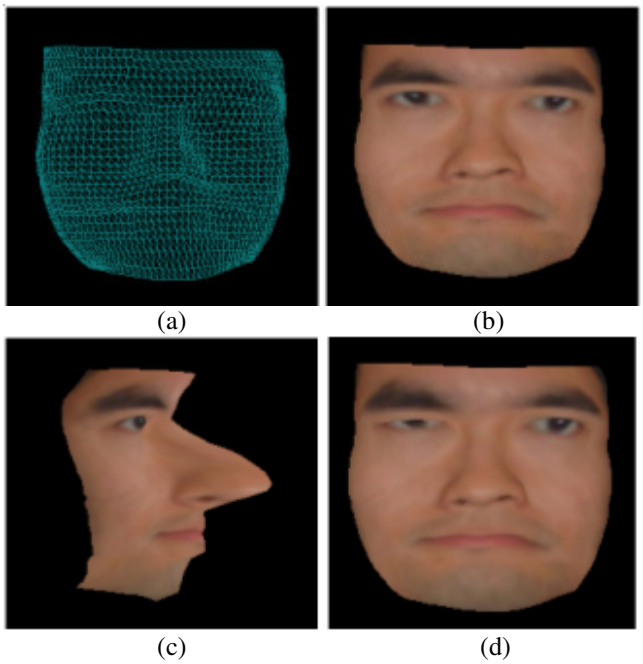



Fig. 9 Generated PDE faces. (a) PDE face model (b) Frontal view of the textured PDE face (c) Profile of the textured PDE face (d) Textured PDE face with the right eye blinking.

Given 28 boundary curves of the PDE face, one may either want to use the curves as they are, or wish to slightly reconfigure their initial positions by scrolling the shape variation sliders around. In Fig. 8, all the sliders except 'Right eye raise' are moved to extremes, in order to make the face as exaggerated as possible. Fig. 8 shows the lateral view of such a face, and its neutral form is Fig. 3 (a). Once the boundary curves of the face shape are finalised, these are saved as the initial boundary conditions, for input to the PDE algorithm. Once the algorithm is executed, the PDE face can be displayed either as a mesh model or as a textured face. Fig. 9 shows the PDE faces generated. The face mesh model produced by the PDE Face Generator is shown in Fig. 9 (a). Using our *uv*-mesh texturing method, Fig. 9 (b) and 9 (c) show the frontal view and the profile respectively of the textured PDE face, using the warped texture map from Fig. 6 (d). Resetting the facial boundary curves, followed by setting the 'Right eye raise' slider to an arbitrary position, Fig. 9 (d) shows the textured PDE face with the right eye blinking.

## 5. Conclusion

Aiming to provide a solution to face modelling problems such as model representation and texturing, this paper introduced a novel approach to generate 3D faces based on the BWPDE algorithm. The 3D face model is generated by importing a number of adjustable facial boundary curves as boundary conditions into the algorithm. The initial position of the facial curves can be modified on demand, providing users a great degree of freedom in individualising the 3D face. Moreover, the developed PDE face can also be textured. Rather than conventionally creating correspondence between the 2D texture map and 3D model before texture mapping is carried out, we developed a texturing method based on the 2D *uv* mesh, associating texels of the texture map with the vertices of the *uv* mesh. Its advantage over the existing texturing approaches is that it eliminates the need for 3D-to-2D association in texturing, and thus, simplifies the process. Since the PDE algorithm brings the *uv* domain into play, any specific face can be textured without the need for the facial expression in the texture map to match exactly that of the 3D face model.

An ideal parameterised model should be one that allows any possible face with any possible expressions to be specified by merely selecting the appropriate facial parameter values [21]. Our PDE Face Generator has an interface with basic shape variation control and facial animation function for demonstration purposes. In order to implement a more sophisticated parameterised model, two broad categories of parameters should be developed in the future, comprehensively encompassing both expression and conformation parameters. As the names imply, the expression parameters should control a full range of facial animation including eyes opening and mouth stretching etc, while the conformation parameters, such as those governing the height and width of the head, should be used to reshape the neutral face. All the above parameters should be developed in relation to those facial boundary curves. Moreover, in order to incorporate these parameters with the PDE Face Generator, a more sophisticated interface is also desired. Our work provides the technological basis for such developments.

## Acknowledgements

This work is funded by the Engineering and Physical Sciences Research Council (EPSRC), U.K. project, Efficient Geometry Parameterisation for Modelling and Animation (EP/D000017/1).

## References

1. D. Pearson, Developments in model-based video coding, *Proceedings of the IEEE, vol.83*, No.6, 1995, pp.892-906.
2. F. Isgro, E. Trucco, P. Kauff & O. Schreer, Three-dimensional image processing in the future of immersive media, *IEEE Trans. on CSVT, Vol. 14,* No. 3, 2004, pp. 288-303.
3. F. Parke, *A parametric model for human faces*, Tech. Report UTEC-CSc-75-047, University of Utah, 1974.
4. M. Patel & P. Willis, The facial animation, construction and editing system. *Proc. of Eurographics 91 Conference,* September 1991, pp. 33-45.
5. K. Waters, A muscle model for animating three-dimensional facial expression, *Computer Graphics, vol. 21,* No. 4, 1987, pp. 17-24.
6. D. Terzopoulos & K. Waters, Analysis and synthesis of facial image sequences using physical and anatomical models, *IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 15,* No. 6, 1993, pp. 569-579.
7. M. Rydfalk, *CANDIDE, a parameterised face*. Report No. LiTH-ISY-I-866, University of Linkoping, Sweden 1987.
8. B. Welsh, *Model-based coding of images,* PhD Dissertation, British Telecom Research Lab, Jan. 1991.
9. J. Ahlberg, *CANDIDE-3: an undated parameterised face.* Report No. LiTH-ISY-R-2326, Linkoping University, Sweden, January 2001.
10. G. Abrantes & F. Pereira, MPEG-4 facial animation technology, survey, implementation, and results. *IEEE trans. on CSVT*, *vol 9*, no.2, 1999, pp. 290-305.
11. J. Kim, M. Song, I. Kim, Y. Kwon, H. Kim & S. Ahn, Automatic FDP/FAP generation form an image sequence, *IEEE Int. Symposium on Circuits and Systems,* May 2000, pp. I40-I43.
12. N. Sarris, N. Grammalidis & M. Strintzis, FAP extraction using 3D motion estimation, *IEEE trans. on CSVT, vol.12,* no.10, Oct. 2002, pp. 865-876.
13. CYBERWARE Home Page, http://www.cyberware.com
14. M. Hoch, G. Fleischmann & B. Girod, Modeling and animation of facial expression based on B-Splines, *Visual Computer, vol.* 11, 1994, pp. 87-95.
15. J. Zhang & L. You, Fast Surface Modelling Using a 6th Order PDE, *Computer Graphics Forum, Vol. 23*, No. 3, 2004, pp.311-320.
16. H. Ugail, M. Bloor & M. Wilson, Techniques for interactive design using the PDE method, *ACM Trans. On Graphics, Vol. 18,* no. 2, 1999, pp. 195-212.
17. G. González Castro, H. Ugail, P. Willis & I. Palmer, Shape morphing using PDE surfaces. *Int. Conf. on Visualisation, Imaging and Image Processing,* Palma De Mallorca, Spain, Aug 2006.
18. M. Bloor & M. Wilson, Using partial differential equation to generate free-form surfaces. *Computer Aided Design, vol. 22,* no. 4, 1990, pp. 202-212.
19. G. Wolberg, *Digital Image Warping* (IEEE Computer Society Press, Los Alamitos, CA, 1990).
20. E. Angel, *Interactive Computer Graphics* (Pearson Education, Inc, 2006).
21. F. Parke & K. Waters, *Computer facial animation* (A K Peters, Ltd, 1996).