International Journal of MODELLING & SIMULATION

VOLUME 33, Number 1, 2013

(Continued on back cover)

ISSN 0228-6203 (Print)

Published by ACTA PRESS ANAHEIM • CALGARY • ZURICH

MODELLING THREE DIMENSIONAL ANIMATED, DETAILED, AND EMOTIONAL FACIAL EXPRESSIONS

Alice J. Lin* and Fuhua (Frank) Cheng*

Abstract

This paper presents methods for generating 3D facial expressions. Special pattern functions are created and employed to generate detailed, animated facial expressions. The created pattern functions include wrinkle patterns (forehead wrinkles, eye corner wrinkles, cheek wrinkles, and frown wrinkles), dimple pattern, and eye pouch effect pattern. A formula for generating realistic, animated teardrops is introduced. Methods for generating real-time vivid shedding tears is also shown. Test results show that these methods are robust, accurate, and consequently will advance 3D facial expressions and animations significantly.

Key Words

3D modelling, facial expressions, tears, animation

1. Introduction

Generating compelling animated facial expressions is an extremely important and challenging aspect of computer graphics. The most important aspect of character animations is the character's ability to render appropriate facial expressions. Although well designed and animated in recent 3D games, virtual reality, computer vision, and animation movies, computer-generated human faces lack detailed surface geometry and emotional expressions: tears. The more detailed the facial expressions, the greater the visual reality. Facial movements and expressions are the most effective and accurate ways of communicating one's emotions. Although weeping and tears are a common concomitant of sad expressions, tears are not indicative of any particular emotion, as in tears of joy. With tears, the face increases the richness as an instrument for communication. Provine *et al.* [1], [2] provided an experiment that removed tears from a sad facial picture. The picture with no tears showed faces' emotional uncertainty, ambiguity, awe, concern, or puzzlement, and not just a decrease in sadness. Sometimes, after the removal of the tears, the faces do not look sad at all and instead look neutral.

Recently, the increasing appearance of virtual characters in computer games, commercials, and movies makes the facial expressions even more important. Computer graphics researchers have been greatly interested in this subject and we have seen considerable innovation in 3D facial expressions during the last decade. However, there have been no theories and methods of making dramatic or realistic animated 3D tears. In the past, the artist usually used various textures to accentuate the static tears. As tears, which consist mostly of water, has high specular reflection and transparency, those antiquated texture tears cannot achieve this degree of realism. We thus present the methods to simulate real-time shedding of tears.

2. Related Work

Facial modelling and animation based on geometric manipulations include key-framing and geometric interpolations [3], parameterizations [4], finite element methods [5], muscle-based modelling [6], visual simulation using pseudo muscles [7], spline models [8], and free-form deformations [7]. Oat [9] presented a technique that involves compositing multiple wrinkle maps and artist animated weights to create a final wrinkled normal map. Face animation employs anatomical models [10], [11]. They allow extending the range of motions beyond muscle-driven expressions by incorporating external forces. Several approaches focus on specific anatomical models for wrinkle simulation [12]–[14]. Hadap *et al.* [15] proposed a texture-based method to generate wrinkle textures from triangle deformation for clothes. Kang *et al.* [16] modelled fine wrinkles procedurally to approximate wrinkling phenomena on cloth. Cutler *et al.* [17] used a stress map to synthesize new wrinkles from a database manually created by artists. Popa *et al.* [18] used a wrinkle generation method to improve the quality of specific frames of captured cloth, based on shadows in the recorded images.

Simulation of water in real-time using particle-based methods is likely the optimal choice and has become popular in recent years [19], [20]. Particles are typically generated according to a controlled stochastic process. Particle systems simulate certain fuzzy phenomena. The parameters in the simulation are all or mostly fuzzy values. Wang

* Department of Computer Science, University of Kentucky, Lexington, KY 40506, USA; e-mail: alice.lin@uky.edu, cheng@cs.uky.edu

Recommended by Dr. J. Shen
(DOI: 10.2316/Journal.205.2013.1.205-5782)

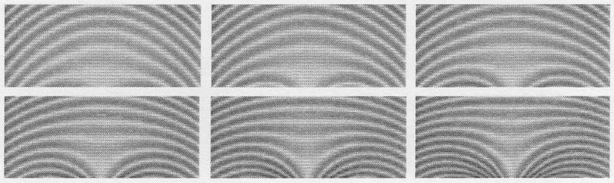

Figure 1. Six frames of a running cycle.

et al. [21] presented a method to simulate water drops on surfaces. The tear simulations rely on textures [22], [23]. These methods are not applicable to generating animated tears. Overall, our approaches achieve better realism.

3. Adding Detailed Animated Expressions

To generate detailed animated expressions, we apply pattern functions to the facial models. We defined the pattern function as a combination of primary function P and damping. Primary function $P = p_1 + p_2 + \ldots$, can be only one basic pattern function p_1 or several basic pattern functions $p_i, i = 1, 2, 3, \ldots$. The pattern function is layered on top of the selected area. It describes the surface of the skin change. As the pattern function adjusts the vertices sequentially, the form and shape in the selected area changes over time. Adjusting the parameters of the pattern functions precisely controls the shape of the expression. Moving the layer of pattern functions refines the expression location. The detailed movement of the expression is a function of the time parameter t in each time interval. The damping function influences the smoothness around the selected area where we applied expression patterns. The damping function dictates expression patterns to decrease and then vanish towards the borders of the selected region. All the pattern functions we developed below are in the Cartesian coordinate system. The variables (x, y, z) in all formulations are local coordinates; α, β, and γ are parameters; t is time.

3.1 Wrinkles

3.1.1 Forehead Wrinkles

The pattern function,

$$p = e^{\alpha\times\left|\sin((x^2+y^2)-(t\times\sin(\arctan(\frac{y}{x}))-\sin(\beta\times\arctan(\frac{y}{x})))^2)\right|} - z \tag{1}$$

is for forehead wrinkles. α controls the depth of wrinkles and β describes the style of wrinkles. Figure 1 shows six frames of a running cycle of this pattern. We applied this pattern function to the forehead surface (Fig. 2(a)). The facial model with wrinkles in the forehead is shown in Fig. 2(b). We also applied it to Fig. 4(a) and got the result in Fig. 4(b).

3.1.2 Eye Corner Wrinkles

For eye corner wrinkles,

$$p = \alpha \times e^{\beta\times\left|\sqrt{x^2+y^2}-t\times\sin(\gamma\times\arctan(y/x))\right|} - z \tag{2}$$

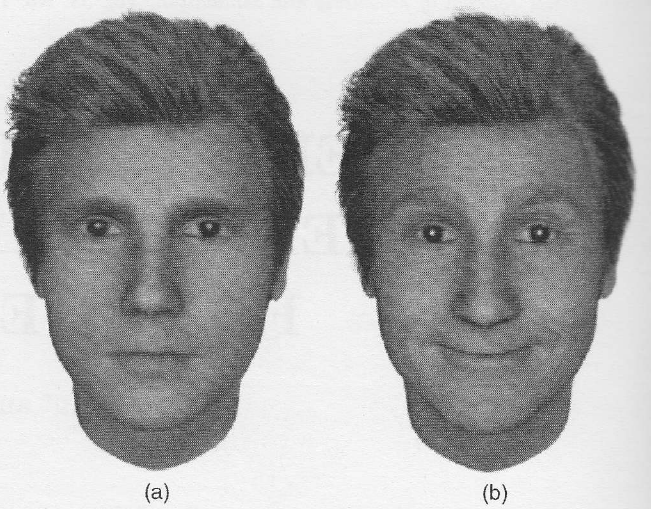

Figure 2. (a) Neutral facial expression and (b) application of forehead wrinkles, eye corner wrinkles, and cheek wrinkles.

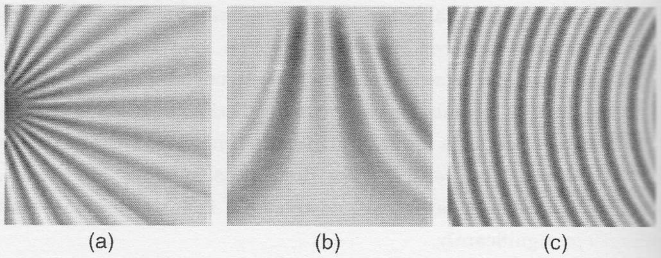

Figure 3. Created pattern functions. (a) Eye corner wrinkles; (b) frown wrinkles; and (c) cheek wrinkles.

where α describes the depth of eye corner wrinkles, β is a preset constant, and γ controls the number of eye corner wrinkles. Figure 3(a) is the pattern for eye corner wrinkles. We applied the pattern to both eye corners (Fig. 2(a)). The result is shown in Fig. 2(b).

3.1.3 Cheek Wrinkles

The pattern function for cheek wrinkles is

$$p = \alpha \times \sin(\beta \times \sqrt{x^2 + y^2} + t) - z \tag{3}$$

where α describes the depth of cheek winkles and β controls the number of cheek wrinkles. Figure 3(c) is the pattern for cheek wrinkles. We add pattern functions to cheeks (Fig. 2(a)) to improve mouth animations and make the face look alive. This achieves the realism of cheek skin. An example is shown in Fig. 2(b).

3.1.4 Frown Wrinkles

In (4) and Fig. 3(b) is the pattern function for the frown. α controls the depth of frown wrinkles and β controls the distance between wrinkles. An example of applying this function to Fig. 4(a) is shown in Fig. 4(c):

$$p = \alpha \times \sin(x \times y/(\beta + t)) - z \tag{4}$$

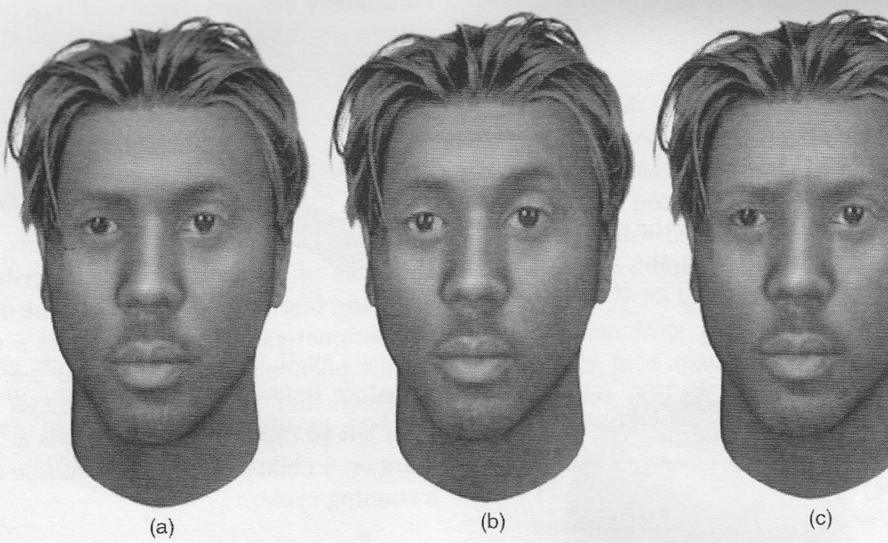

Figure 4. (a) Neutral facial expression; (b) dynamic forehead wrinkles; and (c) frown.

3.1.5 Dynamic Animations

The pattern function controls the polygons and directly controls the displacements of the mesh vertices. Displacing the vertices changes the original positions of the vertices, based on the calculated value of the pattern function. To produce dynamic animations, the pattern function is not applied to each vertex of the selected region. For a selected region with N vertices, we only set certain n vertices as valid and keep the remaining $N - n$ vertices as null. We only compute the deformation of the selected surface at each vertex that is influenced by pattern functions. The valid vertices are randomly chosen and cover all areas in the selected region. Figure 4(b) is an example.

3.2 Dimple and Eye Pouch Effects

3.2.1 Dimple

Dimples are visible indentations of the skin. In most cases, facial dimples appear on the cheeks, and they are typically not visible until someone smiles. The pattern function for a dimple is,

$$p = \log(x^2 + y^2 + t) - z \tag{5}$$

Figure 5(a) is the pattern for a dimple. We applied the dimple function on both cheeks (Fig. 6(a)) and generated the result in Fig. 6(b).

3.2.2 Eye Pouch Effects

In 3D facial expressions, people usually either exaggerate the emotion or make a straight face. Emotions, such as anger or sadness, are usually expressed as opening the mouth and eyes widely and furrowing the eyebrows or deforming the face. Subtle facial expressions also express a person's emotion. To show an angry or sad person, for instance, we can use facial skin trembling on different regions, such as the eye pouches, chin, cheeks, or temple. The pattern function for eye pouch effects is:

$$p = |(\cos(x + \alpha \times t) + \cos(y + \beta \times t))^\gamma|/t - z \tag{6}$$

where α and β describe trembling styles and γ controls the height of the trembling skin that is sticking out. The degree of impact can be manipulated by adjusting parameters of

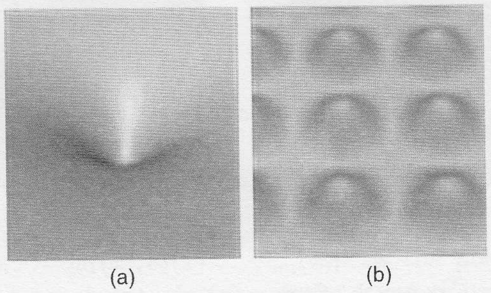

Figure 5. Created patterns. (a) Dimple and (b) eye pouch effects.

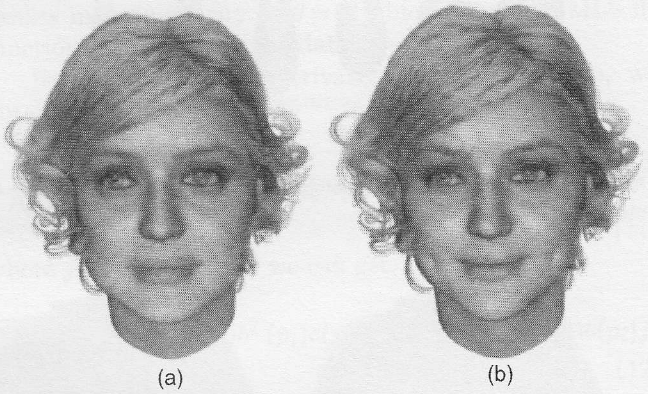

Figure 6. (a) Neutral expression and (b) dimple function applied to cheeks.

the equation. The pattern is shown in Fig. 5(b). Figure 7 shows a frame of a running cycle.

4. Generating Animated Tears

4.1 Dripping Tears

There are two kinds of tear animations. One is the dripping tear (Fig. 9), where the tear falls quickly to the ground, rather than falling slowly down the skin. The teardrop continually changes shape and falls as time increases. The formula we have developed to generate this kind of tear is:

$$x = \frac{\alpha \times \sin(u)\cos(v)}{t \times \cos(u)}, \quad y = \frac{\alpha \times \sin(u)\sin(v)}{t \times \cos(u)},$$
$$z = \beta \times t \times \cos(u) + \lambda \times t \qquad (7)$$

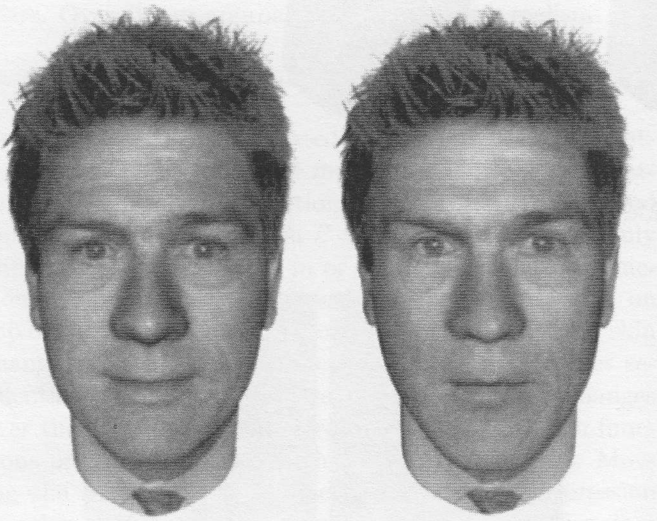

Figure 7. Eye pouch effects.

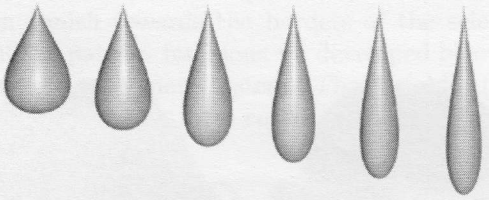

Figure 8. Six selected frames of a running cycle of a teardrop changing shape.

α, β, and λ are preset parameters. α and β describe the sizes of the tear, and λ describes the speed of the dropping tear. t represents the time of animation. u and v are parameters. Figure 8 presents a visual representation of this phenomenon. When the teardrop is falling, time increases and the teardrop's shape continually changes from left to right (Fig. 8). Fig. 9 is an example of dripping tears on a child's face. The example shows four frames of a running cycle.

4.2 Shedding Tears

Another type of tear is the shedding tear. This kind of tear flows along the surface of the skin on the face (Fig. 12). The tear travels over the skin, but it is an individual object. The tear has to become seamlessly connected with the skin and will roll down as time passes. The procedure for generating shedding tears that we have created is as follows.

Step One: We identify the area that the tear will pass through on the face, and extract this area as an individual surface. As the face mesh generally is not fine enough for detailed simulation, we use linear subdivision to refine the mesh of the extracted surface, and call it the base surface (Fig. 10(a)).

Step Two: From the outline of the base surface, we construct the object – the tear will wrap around on the base surface. Figure 11(a) is a simple illustration. The red plane is the base surface. The black plane (top surface) is a surface parallel to the base surface (red). The remaining planes in Fig. 11(a) are constructed surfaces that all connect to both parallel surfaces (red and black). Also see Fig. 10(b) and (c).

Step Three: Figure 11(b) shows how the tear head is constructed when the tear is rolling down the face. The

Figure 9. Four selected frames of a running cycle (dripping tears).

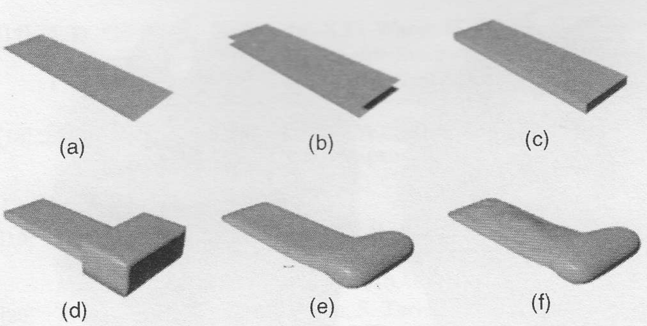

Figure 10. Generating shedding tears. (a) Base surface; (b) copying base surface; (c) two connected surfaces; (d) making tear head; (e) smoothed tear head; and (f) noise added to the top surface.

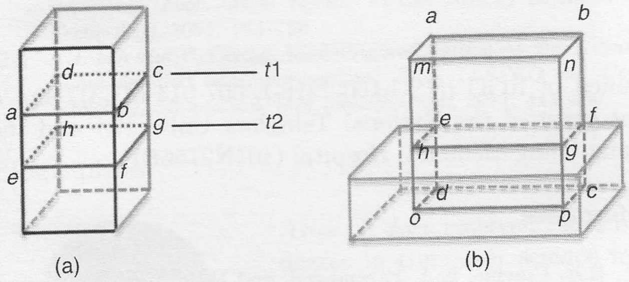

Figure 11. (a) Constructed object (represents the tears) and (b) making tear head.

plane *abcd* represents a base surface that is the same as the red plane in Fig. 11(a). The plane *mnpo* is the top surface and is the same as the black plane in Fig. 11(a). Now, we extrude the planes *gfcp*, *hgpo*, and *hedo*. The result is shown in the green region. The green part of the object is the tear head (also see Fig. 10(d)). Then, we smooth the tear head (green region), giving us the result as in Fig. 10(e).

Step Four: We randomly add noise to the tear surface to simulate the roughness of the skin. The components of human facial skin appearance are skin surface grease, hair, skin layers, fine wrinkles, wrinkles, pores, moles, freckles, *etc.* Every human face has a certain degree of roughness. When the tears flow along the surface of the skin, some amount of tears are trapped, stuck, or blocked by the skin. The tear stain (path) shows that some places are thick and some places are thin. Generally, in 3D facial modelling, the roughness of facial skin is represented by texture and the geometric structure of the facial surface is smooth. Tears have high specular reflection and transparency. With the smooth facial surface, the tear will be evenly distributed in the path of the tear. The tear stain will look like a belt, and the simulation will not properly achieve realism. Thus, we add random noise to the top surface (black in Fig. 11(a)) to reflect the facial skin roughness. In Fig. 10(b), we see two parallel planes. This is only for demonstration. These two surfaces are seen as almost one surface in the simulation. They are very close, so we randomly choose vertices to rise above the top surface. The vertices cannot go below the top surface; otherwise the tear may intersect with the face surface. The density of noise we add depends on the skin's roughness. Rougher skin will add more bumps. The top surface with noise is shown in Fig. 10(f).

When noise is added to the top surface, the surface is deformed. The deformation is based on the method of moving least squares (MLS). Suppose there are n points located at positions $p_i, P=\{p_i \in \mathbb{R}^d, d=3\}, i \in \{1,2,\ldots,n\}$, the surface function $f(x)$ is defined over an arbitrary parameter domain Ω, which approximates the given scalar values f_i for a moving point $x \in \mathbb{R}^d$ in the MLS sense. f is taken from $\prod_r^d$, the space of polynomials of total degree r in d spatial dimensions. $F(x)$ is a weighted least squares (WLS) formulation for an arbitrary fixed point in $\mathbb{R}^d$. When this point moves over the entire parameter domain, a WLS fit is evaluated for each point individually. Then, the fitting function $f(x)$ is obtained from a set of local approximation functions $F(x)$.

$$f(x) = \arg\min\nolimits_{F \in \prod_r^d} \sum\nolimits_{i=1}^{n} w_i(\|x - p_i\|) \|F(p_i) - f_i\| \quad (8)$$

$$F(x) = b^T(x)c(x) = \sum\nolimits_{j \in [1,m]} b_j(x)c_j(x) \quad (9)$$

then, $f(x)$ can be expressed as

$$f(x) = \min \sum\nolimits_i w_i(\|x - p_i\|) \left\|b^T(p_i)c(x) - f_i\right\|^2 \quad (10)$$

where $b(x) = [b_1(x), b_2(x), \ldots, b_m(x)]^T$ is the polynomial basis vector and $c(x) = [c_1(x), c_2(x), \ldots, c_m(x)]^T$ is the vector of unknown coefficients, which we want to resolve. The number m of elements in $b(x)$ and $c(x)$ is given by $m = \frac{(d+r)!}{d!r!}$. $w_i(\|x - p_i\|)$ is the weighting function by distance to x. $w(\theta) = \frac{1}{\theta^2 + \varepsilon^2}$. Setting the parameter ε to zero results in a singularity at $\theta = 0$, which forces the MLS fit function to interpolate the data.

Setting the partial derivatives of $f(x)$ to zero, we obtain,

$$\frac{\partial f(x)}{\partial c_j(x)} = 2\sum\nolimits_i w_i(\|x - p_i\|) b_j(p_i) \left[b^T(p_i)c(x) - f_i\right] = 0 \quad (11)$$

where $j = 1, \ldots, m$, and we can get,

$$\sum\nolimits_i w_i(\|x - p_i\|) b(p_i) b^T(p_i) c(x) = \sum\nolimits_i w_i(\|x - p_i\|) b(p_i) f_i \quad (12)$$

If the matrix $A = \sum_i w_i(\|x - p_i\|) b(p_i) b^T(p_i)$ is not singular, *i.e.*, its determinant is not zero, then,

$$c(x) = A^{-1}\left(\sum\nolimits_i w_i(\|x - p_i\|) b(p_i) f_i\right) \quad (13)$$

As a result,

$$f(x) = b^T(x) A^{-1}\left(\sum\nolimits_i w_i(\|x - p_i\|) b(p_i) f_i\right) \quad (14)$$

Step Five: For the animation of shedding tears, we assume Fig. 11(a) is a path of tear flowing from the top of the object. When time starts (t_1), we select a certain

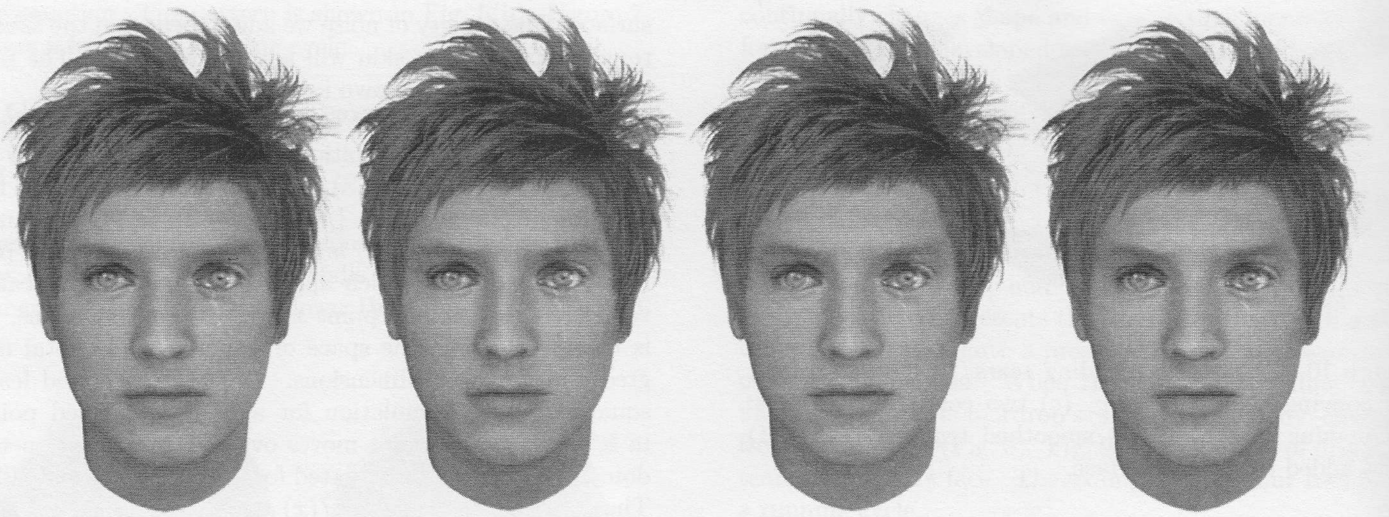

Figure 12. Four selected frames of a running cycle (shedding tears).

portion of the vertices (see Fig. 11(a), the vertices above the plane *abcd*) on the top side of the object. As time passes (t_2), the selected portion of the object increases in size (see Fig. 11(a), the vertices above the plane *efgh*). For each time point, there is a corresponding set of vertices. We use the height of the tear path to indicate the covered portion of the vertices. Height is $h = s + c \times t$, where s and c are constant, and t is time.

We used the above five-step procedure to make an example (Fig. 12) of shedding tears that flow along the skin surface. The example shows four frames of a running cycle.

5. Conclusion

The research of this paper continued from our previous work [24]. In this paper, we have improved and modified methods and procedures. We have developed pattern functions and tears for providing a powerful means of generating natural looking expressions and animations. Although there were a number of methods for generating wrinkles previously, only few were actually used due to their complexity, need for special devices, or production of unrealistic results. Consequently, the significant advantages of our methods compared to existing methods and software are: the methods produced realistic and expressive results with fast performance. The process for generating detailed facial expressions requires less human intervention or tedious tuning. It works on a particular person's face quickly and effectively. Generating expression models often require substantial artistic skills. These methods appeal to both beginner and professional users as they are intuitive and easy to use. Our methods are not developed with *ad hoc* techniques, so it is easily extendible and works in conjunction with existing software and techniques, allowing users to go from a static mesh to an animated face quickly and effortlessly.

Acknowledgements

This work is supported by the National Science Foundation of China (6102010661, 61170324), the National Science Council of ROC (NSC-100-2218-E-007-014-MY3), and a joint grant of the National Tsinghua University and the Chang-Gung Memorial Hospital (101N2756E1).

References

[1] R.R. Provine, K.A. Krosnowski, and N.W. Brocato, Tearing: Breakthrough in human emotional signalling, *Evolutionary Psychology*, *7*(1), 2009, 52–56.

[2] R.R. Provine, Emotional tears and NGF: A biographical appreciation and research beginning, *Archives Italiennes de Biologie*, *149*(2), 2011, 269–274.

[3] F.I. Parke, Computer generated animation of faces, *Proc. ACM Annual Conf.*, 1972.

[4] F.I. Parke, Parameterized models for facial animation revisited, *ACM SIGGRAPH Facial Animation Tutorial Notes*, 1989, 53–56.

[5] B. Guenter, A system for simulating human facial expression, *State of the Art in Computer Animation*, 1992, 191–202.

[6] N. Magnenat-Thalmann, H. Minh, M. Angelis, and D. Thalmann, Design, transformation and animation of human faces, *Visual Computer*, *5*, 1988, 32–39.

[7] P. Kalra, A. Mangili, N.M. Thalmann, and D. Thalmann, Simulation of facial muscle actions based on rational free from deformations, *Computer Graphics Forum*, *11*(3), 1992, 59–69.

[8] M. Nahas, H. Huitric, and M. Saintourens, Animation of a B-spline figure, *The Visual Computer*, *3*(5), 1988, 272–276.

[9] C. Oat, Animated wrinkle maps, advanced real-time rendering in 3D graphics and games, *SIGGRAPH Course*, 2007, 33–37.

[10] I. Essa, S. Basu, T. Darrell, and A. Pentland, Modeling, tracking and interactive animation of faces and heads: Using input from video, *Proc. Computer Animation' 96 Conf.*, 1996.

[11] E. Sifakis, I. Neverov, and R. Fedkiw, Automatic determination of facial muscle activations from sparse motion capture marker data, *ACM Transactions on Graphics*, *24*(3), 2005, 417–425.

[12] N. Magnenat-Thalmann, P. Kalra, J.L. Leveque, R. Bazin, and D. Batisse, A computational skin model: Fold and wrinkle formation, *IEEE Transactions on Information Technology in Biomedicine*, *6*(4), 2002, 317–323.

[13] Y. Zhang and T. Sim, Realistic and efficient wrinkle simulation using an anatomy-based face model with adaptive refinement, *Computer Graphics International*, 2005, 3–10.

[14] K. Venkataraman, S. Lodha, and R. Raghavan, A kinematic-variational model for animating skin with wrinkles, *Computers and Graphics*, *29*(5), 2005, 756–770.

[15] S. Hadap, E. Bangerter, P. Volino, and N. Magnenat-Thalmann, Animating wrinkles on clothes, *Proc. 10th IEEE Visualization Conf., IEEE Computer Society*, 1999.

[16] Y.-M. Kang, J.-H. Choi, H.-G. Cho, and D.-H. Lee, An efficient animation of wrinkled cloth with approximate implicit integration, *The Visual Computer*, *17*(3), 2001, 147–157.

[17] L.D. Cutler, R. Gershbein, X.C. Wang, C. Curtis, E. Maigret, L. Prasso, and P. Farson, An art-directed wrinkle system for CG character clothing and skin, *Graphical Models*, *69*(5–6), 2007, 219–230.

[18] T. Popa, Q. Zhou, D. Bradley, V. Kraevoy, H. Fu, A. Sheffer, and W. Heidrich, Wrinkling captured garments using space-time data-driven deformation, *Computer Graphics Forum*, *28*(2), 2009, 427–435.

[19] N. Chentanez and M. Müller, Real-time simulation of large bodies of water with small scale details, *Proc. Eurographics Symp. Computer Animation*, 2010, 197–206.

[20] T. Pfaff, N. Thuerey, J. Cohen, S. Tariq, and M. Gross, Scalable fluid simulation using anisotropic turbulence particles, *Proc. ACM SIGGRAPH Asia*, *29*(6), 2010.

[21] H. Wang, P.J. Mucha, and G. Turk, Water drops on surfaces, *ACM Transactions on Graphics*, *24*(3), 2005, 921–929.

[22] C.M. de Melo and J. Gratch, Expression of emotions using wrinkles, blushing, sweating and tears, *Proc. 9th Int. Conf. Intelligent Virtual Agents*, 2009.

[23] Y. Jung and C. Knopfle, Dynamic aspects of real-time face-rendering, *Proc. ACM Symp. Virtual Reality Software and Technology*, 2006, 193–196.

[24] A.J. Lin and F. Cheng, Modeling and animating three dimensional detailed facial expressions, *Proc. IASTED Int. Conf., Modeling, Simulation, and Identification*, 2011, 443–450.

Biographies

Alice J. Lin received her Ph.D. degree in computer science from the University of Kentucky, Lexington, KY, USA. Her research areas include 3D computer animation and simulation, 3D computer modelling and algorithms, computer-aided geometric design, computer graphics, image processing, visualization, and virtual reality.

Fuhua (Frank) Cheng is a professor of computer science and director of the Graphics & Geometric Modeling Lab at the University of Kentucky, KY, USA, and a joint professor of computer science at the National Tsinghua University in Taiwan, ROC. He received his B.S. and M.S. degrees in mathematics from the National Tsinghua University in Taiwan, ROC. He received his M.S. degree in computer science and Ph.D. degree in mathematics and computer science from Ohio State University, OH, USA. His research interests include computer-aided geometric modelling, computer graphics, parallel computing in geometric modelling and computer graphics, approximation theory, and collaborative CAD. He is on the editorial board of *Computer Aided Design & Applications*, *Journal of Computer Aided Design & Computer Graphics*, *Computer Aided Drafting, Design and Manufacturing*, and *ISRN Applied Mathematics*.