MODELING 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**.** The pattern functions are employed to generate detailed, animated facial expressions. The created pattern functions include wrinkle patterns (forehead wrinkle, eye corner wrinkle, cheek wrinkle and frown wrinkle), dimple pattern and eye pouch effect pattern. The pattern functions allow a user to directly manipulate them and see immediate results, and allow expressive features to be applied to any existing animation or model. The location and style of an expressive feature can be specified, thus allowing the detailed facial expressions to be modeled as desired. Two kinds of tears can be created and combined with facial models to generate emotional facial expressions. One kind is teardrops, which continually change shape while dripping down the face. The other is shedding tears, which seamlessly connect with the skin as they flow along the surface of the face. The method provides real-time, vivid simulation of tears rolling down the face. These methods both broaden CG and increase the realism of facial expressions.

Key Words

3-Dimensional Modelling, Facial Expressions, Tears, Animation

1. Introduction

Human beings are very accustomed to looking at real faces and can easily spot artifacts in computer-generated models. Ideally, a facial model should be indistinguishable from a real face. Generating compelling animated facial expressions is an extremely important and challenging aspect of computer

*Department of Computer Science, University of Kentucky, Lexington, KY 40506, USA
Email: {ajlin0@cs.uky.edu}{cheng@cs.uky.edu}
Manuscript received March 5, 2012

graphics. The most important aspect of character animations is the character's ability to use facial expressions. Although well designed and animated in recent 3D games, virtual reality, computer vision or animation movies, computer graphics humans are not fully satisfying since they lack important details, which include detailed surface geometry and emotional expressions: tears. Detailed surface geometry contributes greatly to the visual realism of 3D face models. The more the detailed facial expressions are enhanced, the better the visual reality will be. However, it is often tedious and expensive to model facial details, which are authentic and real-time. We present a functional approach for adding detailed animated facial expressions to facial models.

Emotions play a critical role in creating engaging and believable characters. Facial movements and expressions are therefore 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. A happy face with tears appears more joyous, or something in between, perhaps described as bittersweet. With tears, the face increases the richness as an instrument for communication. The way humans read other people's emotions is markedly impacted by the presence (or absence) of tears. Provine et al. [1, 2] provided the first experimental demonstration the tears are a visual signal of sadness by contrasting the perceived sadness of human facial images with tears against copies of those images that had the tears digitally removed. Tear removal produced faces rated as less sad. The research subjects further suggest that after tear removal, the faces' emotional content showed 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. Since 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 modeling and animation research falls into two major categories, those based on geometric manipulations and those based on image manipulations. Geometric manipulations include key-framing and geometric interpolations [3], parameterizations [4], finite element methods [5], muscle based modeling [6], visual simulation using pseudo muscles [7], spline models [8] and free-form deformations [7]. Pighin et al. [9] presented techniques for creating 3D facial models from photographs of a human subject. Zhang et al. [10] presented a geometry-driven facial expression system by using an example-based approach.

Blinn introduced bump mapping [11], which consisted of modifying surface normals before lighting calculations to give a visual impression of wrinkles, without deforming the geometry. This idea has been extensively used since then, especially for facial wrinkles [12]. Bando [13] used an interface to specify wrinkles one by one on the 2D projection of the 3D mesh by drawing a Bezier curve as the wrinkle furrow. Computing a specific mesh required a costly precomputation for energy minimization. The bump map may also be obtained using complex physical-based simulations [14, 12]. Viaud [15] uses a mesh where the locations of potentially existing wrinkles are aligned with isolines of a spline surface to directly deform the geometry for wrinkle. Oat [16] presented a technique that involves compositing multiple wrinkle maps and artist animated weights to create a final wrinkled normal map. It requires manual tuning of the wrinkle map coefficients for each region.

Face animation employs anatomical models [17, 18]. 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 [19-21]. Wu et al. [22] designed procedural methods for wrinkle synthesis as a function of the strain measured in an underlying tissue model. These approaches require complex parameter tuning and expensive computations. Marker-Based Motion Capture [23, 24] acquires data with excellent temporal resolution, but due to their low spatial resolution, they are not capable of capturing expression wrinkles.

Hadap et al. [25] proposed a texture-based method to generate wrinkle textures from triangle deformation for clothes. Kang et al. modeled fine wrinkles procedurally to approximate wrinkling phenomena on cloth [26]. Cutler et al. used a stress map to synthesize new wrinkles from a database manually created by artists [27]. Popa et al. [28] 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 use particle-based methods is likely the optimal choice and has become popular in recent years [29, 30]. 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. Model water flows used volume-preserving approaches, simulating water drop interactions with other water drops and solid surfaces [31, 32]. The tear simulations rely on textures [33, 34]. 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. Since 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 toward 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.

To develop the pattern functions we first created a number of parameterized fundamental surfaces of 3D geometry, such as plane and logarithm surface (Fig. 5(a)). Then, based on these essential surfaces with mathematic savvy, artistic view, and creative thinking of spatial geometry, complex surfaces (pattern functions) were developed.

3.1 Wrinkles

For human faces, wrinkles can roughly be classified into two different types: aging wrinkles and expressive wrinkles. Expressive wrinkles are temporary wrinkles that appear on the face during expressions at all ages. They can appear anywhere on the face during expressions, but for most people, the expressive wrinkles mainly come forth on several regions of the face such as the cheeks, the forehead, and eye corners. From these effects, we have developed a set of basic pattern functions.

Forehead wrinkles:

The pattern function,

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

is for forehead wrinkles. α controls depth of wrinkles and β describes the style of wrinkles. Fig. 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).

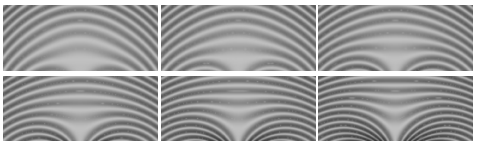


Figure 1: Six frames of a running cycle.

(a) (b)

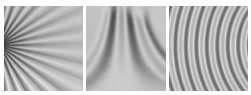
Figure 2: (a) Neutral facial expression. (b) Application of forehead wrinkles, eye corner wrinkles and cheek wrinkles.

***Eye corner wrinkles*:**

For eye corner wrinkles,

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

α describes depth of eye corner wrinkles, β is a preset constant and γ controls the number of eye corner wrinkles. Fig. 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).



(a) (b) (c)

Figure 3: Created pattern functions. (a) Eye corner wrinkles, (b) Frown wrinkles, and (c) Cheek wrinkles.

Cheek wrinkles:

The pattern function for cheek wrinkles is

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

α describes depth of cheek winkles and β controls the number of cheek wrinkles. Fig. 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).

Frown wrinkles:

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

$$p = \alpha \times \sin(x \times y / (\beta + t)) - z,$$

(a) (b) (c)

Figure 4: (a) Neutral facial expression. (b) Forehead wrinkles. (c) Frown.

Dynamic animations:

The pattern function controls the polygon and directly controls the displacements of the mesh vertices. Displacing the vertices changes the original position of the vertex, 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. Fig. 4(b) is an example.

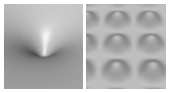
3.2 Dimple and Eye Pouch Effects

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

$$p = \log(x^2 + y^2 + t) - z.$$

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



(a) (b)

Figure 5: Created pattern functions. (a) Dimple (b) Eye pouch effects.

(a) (b)

Figure 6: (a) Neutral expression. (b) Dimple function applied to cheeks.

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

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

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

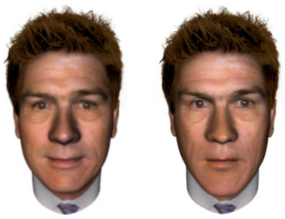


Figure 7: Eye pouch effects.

4. Generating Animated Tears

4.1 Dripping tears

There are two kinds of tear animations. One is the dripping tear (Fig. 11), 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:

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

α, β and λ are preset parameters. α and β describe the sizes of the tear, λ describes the speed of the tear dropping. t represents the time of animation. u and v are parameters. Fig. 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. 11 is an example of dripping tears on a child's face. The example shows four frames of a running cycle.

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

4.2 Shedding tears

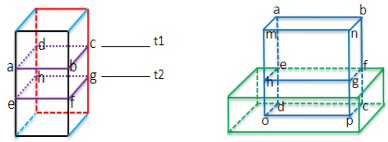
Another type of tear is the shedding tear. This kind of tear flows along the surface of the skin on the face (Fig. 12). The tear travels over the skin, but it is an individual object. The tear has to become seamlessly connected with the skin and will roll down as time passes. The procedure for generating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

Step one: We select the areas that the tear will pass through on the face, and then extract this area as an individual surface. Since the face mesh generally is not fine enough for detailed simulation, we use linear subdivision to subdivide the extracted surface to refine meshes, and call it the base surface (Fig. 9(a)).

Step two: From the outline of the base surface, we construct the object - the tear will wrap around on the base surface. Fig. 10(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. 10(a) are constructed surfaces that all connect to both parallel surfaces (red and black). Also see Fig. 9 (b) and (c).

(a) (b) (c) (d) (e) (f)

Figure 9: Generating shedding tears (a) Base surface. (b) Copy of the base surface. (c) Two connected surfaces**.** (d) Making the head of the tear. (e) Smoothed head of the tear. (f) Noise added to the top surface.



(a) (b)

Figure 10: (a) Constructed object (represents the tears). (b) Making the head of the tear.

Step three: Fig. 10 (b) shows how we make the head of the tear when the tear is rolling down the face. The plane *abcd* represents a base surface that is the same as the red plane in Fig. 10(a). The plane *mnpo* is the top surface and is the same as the black plane in Fig. 10(a). Now, we extrude the planes *gfcp, hgpo* and *hedo*. The result is shown by the green region. The green part of the object is the head of the tear (also see Fig. 9(d)). Then, we smooth the head of the tear (green region), giving us the result as Fig. 9(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 oil, 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 modeling, 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. 10(a)) to reflect the facial skin roughness. In Fig. 9(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 from the top surface. The vertices cannot go down from the top surface; otherwise the tear may intersect with the face surface. The density of noise we add depends on the skin's roughness. The rougher skin will add more bumps. The top surface with the noise is shown in Fig. 9(f).

When the top surface adds noise, the surface is deformed. The deformation is based on the method of moving least squares. Suppose that we have n points located at positions p_i, $P = \{p_i \in \mathbb{R}^d, d = 3\}$, $i \in \{1, 2, ..., n\}$. The surface function $f(x)$ is defined in 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 $F(x)$.

$$f(x) = \arg\min_{F \in \prod_r^d} \sum_{i=1}^{n} w_i(\|x - p_i\|) \|F(p_i) - f_i\|,$$

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

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

$$f(x) = \min \sum_i w_i(\|x - p_i\|) \|b^T(p_i)c(x) - f_i\|^2$$

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

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

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

$$\sum_i w_i(\|x-p_i\|)b(p_i)b^T(p_i)c(x)=\sum_i w_i(\|x-p_i\|)b(p_i)f_i$$

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

$$c(x)=A^{-1}(\sum_i w_i(\|x-p_i\|)b(p_i)f_i)\cdot$$

As a result,

$$f(x)=b^T(x)A^{-1}(\sum_i w_i(\|x-p_i\|)b(p_i)f_i)$$

***Step five*:** For the animation of shedding tears, we assume Fig. 10(a) is a path of tear flowing from the top of the object. When time starts (t_1), we select a certain portion of the vertices (see Fig. 10(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. 10(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 $h = s + c * t$, 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.

Figure 11: Four selected frames of a running cycle (dripping tears).

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

5. Discussion

Despite the complexity of human facial anatomy and people's inherent sensitivity to facial appearances, 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 is intuitive and easy to use, allowing users to directly manipulate it and see immediate results. It allows the expressive features to be applied to any arbitrary mesh structures. It works on a particular person's face quickly and effectively. Our methods are not developed with ad hoc techniques, so it is easily extendible and work in conjunction with existing software and techniques, allowing users to go from a static mesh to an animated face quickly and effortlessly. Generating expression models often require substantial artistic skills. These methods appeal to both beginner and professional users.

6. Conclusion

We have presented a novel approach for generating detailed, animated, realistic facial expressions by employing a set of pattern functions. Our method adapts to individual faces and allows the expressive features to be applied to any existing animation or model. The location and style can be specified, thus allowing the detailed facial expressions to be modeled as desired. The process for generating detailed facial expressions requires less human intervention or tedious tuning. It can also be applied to a variety of CG human body skins and animal faces.

We have also presented another novel approach for generating tears. The formula for generating realistic, animated teardrops, which continually change shape as the tear drips down the face has been introduced. The method and procedures for generating real-time vivid shedding tears also have been shown. Our results indicate that these methods are robust and accurate, and both broaden CG and increase the realism of facial expressions. It will evolve 3D facial expressions and animations significantly. These methods can also be applied to CG human body skins and other surfaces to simulate flowing water.

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.

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] F. Pighin, and J. P. Lewis, Performance-driven facial animation, *SIGGRAPH Course Notes*, 2006.

[10] Q. Zhang, Z. Liu, B. Guo, D. Terzopoulos, and H-Y Shum, Geometry-driven photorealistic facial expression synthesis. IEEE Transactions on Visualization and Computer Graphics, 12(1), 2006, 48-60.

[11] J. F. Blinn, Simulation of wrinkled surfaces, *Proceedings of SIGGRAPH'78*, 1978, 286–292.

[12] L. Boissieux, G. Kiss, N. Magnenat-Thalmann, and P. Kalra, Simulation of skin aging and wrinkles with cosmetics insight, *Proceedings Eurographics Workshop on Computer Animation and Simulation,* 2000, 15–27.

[13] Y. Bando, T. Kuratate, and T. Nishita. A simple method for modeling wrinkles on human skin, *Proceedings of the 10 th Pacific Conference on Computer Graphics and Applications,* 2002, 166-175.

[14] Y. Wu, P. Kalra, L. Moccozet, and N. Magnenat-Thalmann, Simulating wrinkles and skin aging, *The Visual Computer*, 15(4), 1999, 183–198.

[15] M.-L. Viaud and H. Yahia, Facial animation with wrinkles, *EG Workshop on Animation and Simulation*, 1992.

[16] C. Oat, Animated wrinkle maps, Advanced Real-Time Rendering in 3D Graphics and Games. *SIGGRAPH Course*, 2007, 33-37.

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

[18] 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.

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

[20] 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.

[21] K. Venkataramana, S. Lodhaa, and R. Raghavanb, A kinematic-variational model for animating skin with wrinkles, *Computers & Graphics*, 29(5), 2005, 756–770.

[22] Y. Wu, P. Kalra, and N. Magnenat-Thalmann, Simulation of static and dynamic wrinkles of skin, *Proc. of Computer Animation*, 1996, 90–97.

[23] L. Williams, Performance-driven facial animation, *Proc. of ACM SIGGRAPH*, 24, 1990, 235-242.

[24] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister and M. Gross, Multi-Scale capture of facial geometry and motion. *ACM Transactions on Graphics*, 26(3), 2007, 33.

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

[26] 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.

[27] 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.

[28] T. Popa1, 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.

[29] N. Chentanez and M. Müller, Real-time simulation of large bodies of water with small scale details, *Proceedings of Eurographics Symposium on Computer Animation*, 2010, 197-206.

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

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

[32] R. Tong, K. Kaneda, and H. Yamashita, A volume-preserving approach for modeling and animating water flows generated by metaballs, *The Visual Computer* 18(8), 2002, 469–480.

[33] C. M. de Melo and J. Gratch, Expression of emotions using wrinkles, blushing, sweating and tears, *Proc. of the 9th International Conference on Intelligent Virtual Agents,* 2009.

[34] Y. Jung and C. Knopfle, Dynamic aspects of real-time face-rendering, *Proceedings of the ACM symposium on Virtual reality software and technology*, 2006, 193–196.

Biographies