LiveMail: Personalized Avatars for Mobile Entertainment

Miran Mosmondor[1], Tomislav Kosutic[2], Igor S. Pandzic[3]
[1]Ericsson Nikola Tesla, Krapinska 45, p.p. 93, HR-10 002 Zagreb
miran.mosmondor@ericsson.com
[2]KATE-KOM, Drvinje 109, HR 10 000 Zagreb
tomislav.kosutic@vip.hr
[3]Faculty of electrical engineering and computing, Zagreb University, Unska 3, HR-10 000 Zagreb
igor.pandzic@fer.hr

Abstract— **LiveMail is a prototype system that allows mobile subscribers to communicate using personalized 3D face models created from images taken by their phone cameras. The user takes a snapshot of someone's face - a friend, famous person, themselves, even a pet - using the mobile phone's camera. After a quick manipulation on the phone, a 3D model of that face is created and can be animated simply by typing in some text. Speech and appropriate animation of the face are created by speech synthesis. Animations will be sent to others as short videos in MMS. They can be used as fun messages, greeting cards etc. The system is based on a client/server communication model. The clients are mobile devices or web clients so messages can be created, sent and received on the mobile phone or on a web page. The client has a user interface that allows the user to input a facial image and place a simple mask on it to mark the main features. The client then sends this data to the server that builds a personalized face model. The client also provides an interface that lets the user request the creation of animated messages using speech synthesis. The animations are created on the server. It is planned to have several versions of the client: Symbian, midlet-based, web-based, wap-based, etc. The server, Symbian client, midlet-based client and the web client have been implemented as prototypes. We present the system architecture and the experience gained building such a system.**

I. INTRODUCTION

Mobility and the Internet are the two most dynamic forces in communications technology today. In parallel with the fast worldwide growth of mobile subscriptions, the fixed Internet and its service offerings have grown at a rate far exceeding all expectations. Number of people connected to the Internet is continuing to increase and GPRS and WCDMA mobile networks are enabling connectivity virtually everywhere and at any time with any device. A new form of interactive communication behavior is emerging from the combination of different media with applications that are randomly invoked by multiple users and end-systems. With such advances in computer and networking technologies comes the challenge of offering new multimedia applications and end user services in heterogeneous environments for both developers and service providers.

The goal of the project was to explore the potential of existing face animation technology [11] for innovative and attractive services for the mobile market, exploiting in particular the advantages of more recent technologies – primarily MMS and GPRS. The new service will allow customers to take pictures of people using the mobile phone camera and obtain a personalized 3D face model of the person in the picture through a simple manipulation on the mobile phone.

Creating animated human faces using computer graphics techniques has been an increasingly popular research topic the last few decades [1], and such synthetic faces, or virtual humans, have recently reached a broader public through movies, computer games, and the world wide web. Current and future uses include a range of applications, such as human-computer interfaces, avatars, video communication, and virtual guides, salesmen, actors, and newsreaders [12].

There are various techniques on how to produce personalized 3D face models. One of them is to use 3D modeling tool such as 3D Studio Max or Maya. However, manual construction of 3D models using such tools is often expensive, time-consuming and it sometimes doesn't result with desirable model. Other way is to use specialized 3D scanners. In this way face models can be produced with very high quality but impracticability of such approach in mobile environment is more than obvious. Also, there have been methods that included usage of two cameras placed at certain angle and special algorithm for picture processing to create 3D model [8]. Some other methods, like in [9], are using three perspective images taken from a different angles to adjust the deformable contours on the generic head model.

Our approach in creating animatable personalized face models is based on face model adaptation of existing generic face model, similar to [14]. However, in order to achieve simplicity on a camera-equipped mobile device, our adaptation method uses a single picture as an input.

Created personalized face model can be animated using speech synthesis [10] or audio analysis (lip synchronization)[13], and such personalized animated messages can be delivered to other customers using MMS or GPRS. Our face animation system is based on the MPEG-4 standard on Face and Body Animation (FBA) [5][2]. This standard specifies a set of Facial Animation Parameters (FAPs) used to control the animation of a face model. The FAPs are based on the study of minimal facial actions and are closely related to muscle actions. They represent a complete set of basic facial actions, and therefore allow the representation of most natural facial expressions. The lips are particularly well defined and it is possible to precisely define the inner and outer lip contour.

Exaggerated values permit to define actions that are normally not possible for humans, but could be desirable for cartoon-like characters.

All the parameters involving translational movement are expressed in terms of the Facial Animation Parameter Units (FAPU) (Figure 1.). These units are defined in order to allow interpretation of the FAPs on any facial model in a consistent way, producing reasonable results in terms of expression and speech pronunciation. They correspond to fractions of distances between some key facial features (e.g. eye distance). The fractional units used are chosen to allow enough precision.

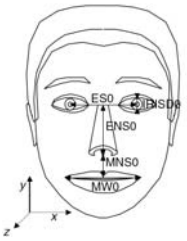

Figure 1. A face model in its neutral state and defined FAP units (FAPU) (ISO/IEC IS 14496-2 Visual, 1999)

The FAP set contains two high level FAPs for selecting facial expressions and visemes, and 66 low level FAPs. The low level FAPs are expressed as movement of feature points in the face, and MPEG-4 defines 84 such points (Figure 2.). The feature points not affected by FAPs are used to control the static shape of the face. The viseme parameter allows rendering visemes on the face without having to express them in terms of other parameters or to enhance the result of other parameters, insuring the correct rendering of visemes. Similarly, the expression parameter allows definition of high-level facial expressions.

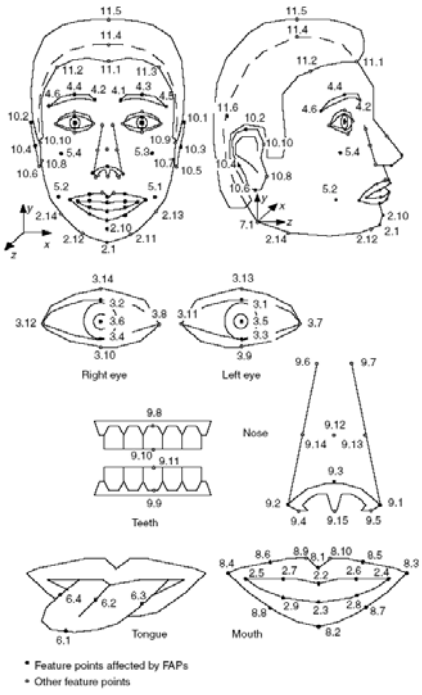

Figure 2. Face feature points (ISO/IEC IS 14496-2 Visual, 1999)

The FAPs can be efficiently compressed and included in an FBA bitstream for low bitrate storage or transmission. An FBA bitstream can be decoded and interpreted by any MPEG-4 compliant face animation system [3][4], and a synthetic, animated face be visualized.

Customers can thus deliver extremely personalized, attractive content using simple manipulations on their phones. A parallel web service can be offered for people who do not (yet) have mobile phones and subscriptions allowing them to access all functions of the system.

II. SYSTEM ARCHITECTURE

Basic functionalities of LiveMail service are simple creation of personalized face model, and creation, transmission and display of such virtual character on various platforms. Procedure in creation of such personalized face model can simply be described as recognition of characteristic face lines on the taken picture and adjustment of generic face model to those face lines. Most important face lines are size and position of face, eyes, nose and mouth. Speech animation of the character is created from the input text. More details on the each module implementation are given in next chapters and in this chapter system architecture is described.

Personalization of virtual characters and creation of animated messages with speech and lips synchronization is a time-consuming process that requires a lot of computational power. Our system architecture is mostly defined by this fact. Capabilities of mobile devices have improved in last few years, but they are still clearly not capable of such demanding computing. Thus, our system is not implemented on one platform, rather is divided in several independent modules. The system is based on a client/server communication model. Basic task of server is to perform computationally expensive processes like, previously mentioned, personalization of virtual character and creation of animated messages with speech and lips synchronization. On the other hand, the client side is responsible for displaying the animated message of the personalized virtual character, and to ensure a proper user interface for the creation of a new personalized virtual character and its animated message (Figure 3.). In this way LiveMail clients are implemented on various platforms: Symbian-based mobile devices, mobile devices with Java support (Java 2 Micro Edition), WAP and Web interface.

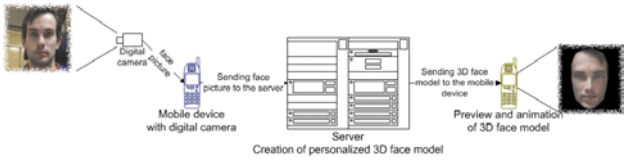

Figure 3. Simplified use-case scenario

Client application consists of a user interface through which users can create new personalized virtual characters and send animated messages, preview created messages and view received animated messages. When creating the personalized character the interface allows the user to input a facial image and place a simple mask on it to mark the main features. The client then sends this data to the server that builds a personalized face model (Figure 4.).

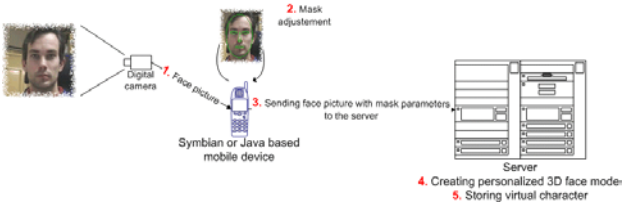

Figure 4. Personalized 3D face model creation

When creating animated messages, the user selects a virtual character, inputs text and addresses the receiver. Client application then sends a request for creation of the animated message to the server, which then synthesizes the speech and creates matching facial animation using the text-to-speech framework. Animated message is then adjusted to the capabilities of the receiving platform. Also, based on the client, animated message could be transformed to the MMS and sent to the any mobile device that supports it (Figure 5.).

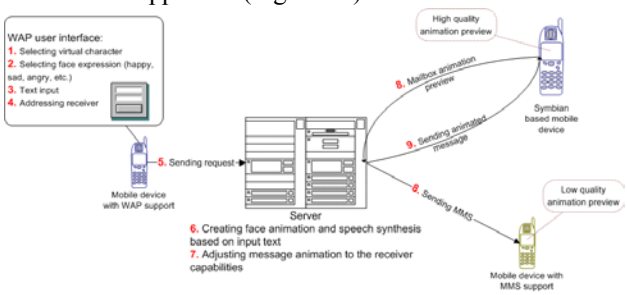

Figure 5. Creating animation through WAP interface

Data transfer between client application on mobile device and server is achieved through General Packet Radio Service (GPRS). For reliability of transported data TCP (Transmission Control Protocol) is used on transport and HTTP (Hyper Text Transport Protocol) on application layer. HTTP is selected because of a well software support and availability on almost all platforms. Client applications are sending its requests using HTTP Post method in the same way that Web browsers are sending data from HTML form. More precisely, mobile clients, in our case are using chunked transfer encoding. LiveMail service is not designed as a Web service due to incomplete support for Simple Object Access Protocol (SOAP) on Symbian and J2ME platforms.

III. THE SERVER

The architecture of LiveMail server prototype is a combination of a light HTTP server and an application server (Figure 6.). HTTP server provides clients with user interface and receives requests, while application server processes client requests: creates new personalized 3D face models and animation messages. User interface is dynamically generated using XSL transformations from XML database each time client makes a request. Database holds information about user accounts, their 3D face models and contents of animated messages. LiveMail server is a multithreaded application. Light HTTP server as a part of it can simultaneously receive many client requests and pass them to application server for processing. Application server consists of many modules assigned for specific tasks like: 3D face model personalization, animation message creation and more. During the process of 3D face model personalization and animated message creation there are resources that cannot be run in multithread environment. Therefore they need to be shared among modules. Microsoft's Text to speech engine, which is used for speech synthesis and as a base of 3D model animation, is a sample of a shared resource. To use such a resource all application server modules need to be synchronized. So, it is clear that technologies used to build LiveMail server prototype have a direct impact on its architecture.

Adaptation of the virtual character is done on the server. The server receives from the client the entry parameters: the picture of person whose 3D model we want to create and the characteristic facial points in that picture. The server also has a generic 3D face model that is used in adaptation. Based on these inputs, the server deforms and textures the generic model in such a way that it becomes similar to the face in the picture, and thus produces the new 3D mode ready for animation - the .wra file.

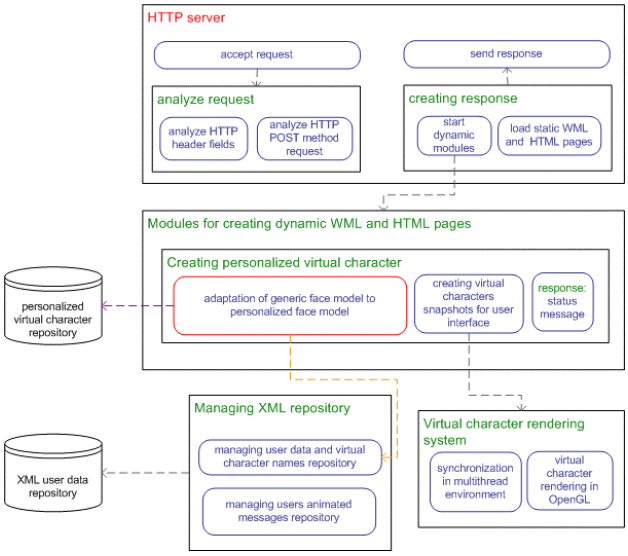

Figure 6. Simplified server architecture

The algorithm starts by receiving characteristic points from mobile device that are assigned to indices in .fdp file. This file contains coordinates of characteristic points and indexes in corresponding VRML model. Together with .fdp comes .wra file. That is the generic 3D model. There are three stages of adaptation. The first stage is normalization. Modulation of highest (11.4) and lowest (2.1) point (see Figure 2.) between known model and received points is made here.

We translate the generic face model to highest point (11.4) from mobile device. Translated model does not suit size of received model so we have to scale it. We calculate vertical ratio of both models and move every point of the generic model in corresponding ratio. We distinguish vertical and horizontal scaling. In horizontal scaling we look at the horizontal distance between every point to 11.4 point, relative to face axis symmetry. Vertical scaling is easier, due there is no axis symmetry.

The next stage is processing. We distinguish texture processing and model processing. With N existing points we create net of triangles that covered all normalized space. It's important to notice that beyond the face the net must be uniform.

Based on known points and known triangles the interpolator algorithm is able to determine the coordinates of any new point in that space using interpolation in barrycentric coordinates. The interpolator used here is described in [15]. We forward characteristic points received from mobile device to interpolator. Interpolator (based on triangles net) will determine location of other points that we need to create new model.

The last stage of the algorithm is renormalization. It is practically the same algorithm as in first segment, except we roll back model from normalized space back to space where it was before starting of algorithm.

IV. THE CLIENT

The animation itself is created on the server that provides users with transparent access, meaning various client types could be used. After personalized face model with proper animation was created, it is send back to the client, where it can be previewed. Multi-platform delivery, and the capability to implement support for virtually any platform is one of the stronger points of this system. The strategy to achieve this is to use a bare-minimum face animation player core. This core can be easily ported to any platform that supports 3D graphics.

The face animation player is essentially an MPEG-4 FBA decoder. When the MPEG-4 Face Animation Parameters (FAPs) are decoded, the player needs to apply them to a face model. Our choice for the facial animation method is interpolation from key positions, essentially the same as the morph target approach widely used in computer animation and the MPEG-4 FAT approach [2]. Interpolation was probably the earliest approach to facial animation and it has been used extensively. We prefer it to procedural approaches and the more complex muscle-based models because it is very simple to implement, and therefore easy to port to various platforms; it is modest in CPU time consumption; and the usage of key positions (morph targets) is close to the methodology used by computer animators and should be easily adopted by this community.

The way the player works is the following. Each FAP (both low- and high-level) is defined as a key position of the face, or morph target. Each morph target is described by the relative position of each vertex with respect to its position in the neutral face, as well as the relative rotation and translation of each transform node in the scene graph of the face. The morph target is defined for a particular value of the FAP. The position of vertices and transforms for other values of the FAP are then interpolated from the neutral face and the morph target. This can easily be extended to include several morph targets for each FAP and use a piecewise linear interpolation function, like the FAT approach defines. However, current implementations show simple linear interpolation to be sufficient in all situations encountered so far. The vertex and transform movements of the low-level FAPs are added together to produce final facial animation frames. In case of high-level FAPs, the movements are blended by averaging, rather than added together.

Due to its simplicity and low requirements, the face animation player is easy to implement on a variety of platforms using various programming languages. Additionally, for the clients that are not powerful enough to render 3D animations, the animations can be pre-rendered on the server and sent to the clients as MMS messages containing short videos or animated GIF images. In next chapters, we describe the following implementations of the client: the Symbian client, Java applet-based web client, and a generic mobile phone client built around J2ME, MMS and WAP. The first two implementations are full 3D clients, while the last one only supports pre-rendered messages.

A. Symbian client

The last few years have seen dramatic improvements in how much computation and communication power can be packed into such a small device. Despite the big improvements, the mobile terminals are still clearly less capable than desktop computers in many ways. They run at a lower speed, the displays are smaller in size and have a lower resolution, there is less memory for running the programs and for storing them, and you can use the device for a shorter time because the battery will eventually run out.

Rendering 3D graphics on handheld devices is still a very complex task, because of the vast computational power required to achieve a usable performance. With the introduction of color displays and more powerful processors, mobile phones are becoming capable of rendering 3D graphics at interactive frame rates. First attempts to implement 3D graphics accelerators on mobile phones have already been made. Mitsubishi Electric Corp. announced their first 3D graphics LSI core for mobile phones called Z3D in March 2003. Also other manufacturers like Fuetrek, Sanshin Electric, Imagination Technologies and ATI published their 3D hardware solution for mobile devices a few months after.

Beside hardware solutions, other important thing for 3D graphics on mobile devices is availability of open-standard, well-performing programming interfaces (APIs) that are supported by handset manufacturers, operators and developers alike. These are OpenGL ES (OpenGL for Embedded Systems) and Java Mobile 3D graphics (also known as JSR 184) that have emerged in last several months.

Our mobile client is implemented on Symbian platform as standalone C++ application. After taking a photo with camera or simply upload it to application, user needs to adjust the mask with key face part outlined (Figure 7.). Mask is used to define 26 feature points on the face that are then, together with picture send to the server for face adaptation, that was described previously.

Figure 7. Symbian user interface and mask adjustment

After creation of personalized face model, it is sent back to the user where it can be previewed (Figure 8.). The face animation player on Symbian platform for mobile devices is based on DieselEngine. Because of low CPU time consumption and low memory requirements, MPEG-4 FBA decoder can be used on mobile device. Most important issues concerned rendering 3D graphics on mobile device. For that purpose DieselEngine was used. It is collection of C++ libraries that helps building applications with 3D content on various devices. DieselEngine has low-level API (Application Program Interface) that is similar to Microsoft DirectX and high level modules had to be implemented. The most important is VRML parser that is used to convert 3D animatable face model from VRML format to Diesel3D scene format (DSC). Other modules enable interaction with face model like navigation, picking and centering.

Figure 8. 3D face model preview on Symbian platform

We have tested this implementation on Sony Ericsson P800 mobile device with various static face models. Interactive frame rates were achieved with models containing up to several hundreds polygons. Generic face model that is used in LiveMail system uses approximately two hundred polygons and its performances are shown in Figure 9. After animation has started, in this case in 21st second, frame rate drops to average of 10 frames per second (FPS), but this is still relatively high above the considered bottom boundary for interactive frame rates on mobile devices.

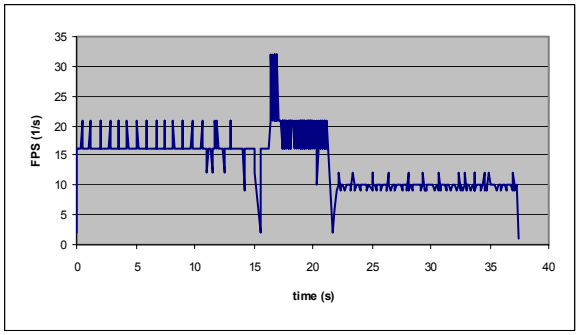

Figure 9. Generic face model performances on SE P800 mobile device

B. *Java applet-based web client*

A parallel web service is also offered as an interface to the system that allows users to access all the functionality: creating new personalized virtual characters and composing messages that can be sent as e-mail. The system keeps the database of all the created characters and sent messages for a specific user identified by e-mail and password.

As another way of creating new personalized virtual characters, Java applet is used with the same functional interface described earlier. Compared to mobile input modules, the Java Applet provides more precision in defining feature points since the adjustment of the mask is handled in higher resolution of the desktop computer. Also the cost of data transfer, which is significantly cheaper compared to mobile devices, makes is possible to use higher resolution portrait pictures in making better looking characters. Portrait pictures are uploaded from the local computer.

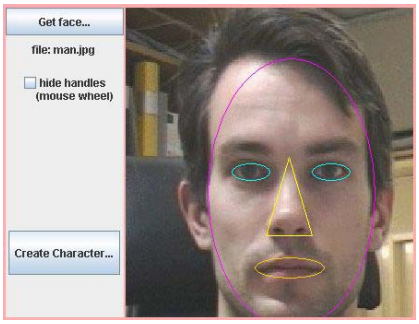

Figure 10. Mask adjustment on Java applet-based web client

New messages through the web client can be made with any previously created characters. The generated animation is stored on the server and the URL to the LiveMail is sent to the specified e-mail address. The html page URL points to is produced dynamically. Player used to view LiveMails is a Java applet based on the Shout3D rendering engine. Since it is a pure Java implementation and requires no plug-ins, the LiveMail can be viewed on any computer that has Java Virtual Machine installed.

It shows performance of 15-40 fps with textured and non-textured face models of up to 3700 polygons on a PIII/600MHz, growing to 24-60 fps on PIII/1000, while the required bandwidth is approx 0.3 kbit/s for face animation 13 kbit/s for speech, 150K download for the applet and approx. 50K download for an average face model. This performance is satisfactory for today's mobile PC user connecting to the Internet with, for example, GPRS. More details on this implementation and performances can be found in [11].

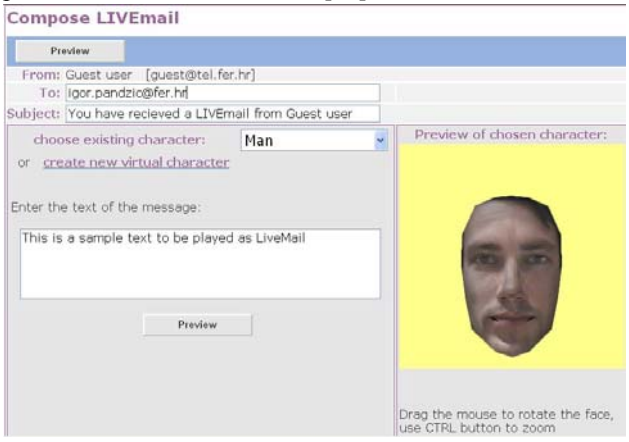

Figure 11. Composing LiveMail message on web client

C. *Generic mobile client built around J2ME, MMS and WAP*

The face animation player is easy to implement on a variety of platforms using various programming languages. However, for the clients that are not powerful enough to render 3D animations we offer alternative. For this type of devices, the animations can be pre-rendered on the server and sent to the clients as MMS messages containing short videos or animated GIF images.

WAP service is also provided for mobile phones. It provides the functionality of creating and sending LiveMail with previously generated personalized characters.

Input module through which personalized face model is created is also implemented on Java 2 Micro Edition (J2ME) platform. However, in this case Mobile Media API (MMAPI), that is additional J2ME package, needs to be used because J2ME itself does not define access to native multimedia services.

V. CONCLUSIONS

LiveMail is a pure entertainment application. Users deliver personalized, attractive content using simple manipulations on their phones. They create their own, fully personalized content and send it to other people. By engaging in a creative process - taking a picture, producing a 3D face from it, composing the message - the users have more fun, and the ways they use the application are only limited by their imagination.

LiveMail is expected to appeal to younger customer base and to promote more recent services, primarily GPRS and MMS. It is expected to directly boost revenues from these services by increasing their usage.

Due to highly visual and innovative nature of the application, there is a considerable marketing potential. The 3D faces can be delivered throughout various marketing channels, including the web and TV, and used for branding purposes.

VI. ACKNOWLEDGMENTS

We would like to acknowledge Visage Technologies AB, Linköping, Sweden, for providing the underlying face animation technology used for the system described in this paper.

REFERENCES

[1] F.I. Parke, K. Waters: “Computer Facial animation”, A.K.Peters Ltd, 1996., ISBN 1-56881-014-8

[2] Igor S. Pandzic, Robert Forschheimer (editors): “MPEG-4 Facial Animation - The standard, implementations and applications", John Wiley & Sons, 2002, ISBN 0-470-84465-5.

[3] M. Escher, I. S. Pandzic, N. Magnenat-Thalmann, “Facial Deformations for MPEG-4,” Proc. Computer Animation 98, Philadelphia, USA, pp. 138-145, IEEE Computer Society Press, 1998.

[4] F. Lavagetto, R. Pockaj, “The Facial Animation Engine: towards a high-level interface for the design of MPEG-4 compliant animated faces,” IEEE Trans. on Circuits and Systems for Video Technology, Vol. 9, No. 2, March 1999.

[5] ISO/IEC 14496 - MPEG-4 International Standard, Moving Picture Experts Group, http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm

[6] 3D Arts, DieselEngine SDK, http://www.3darts.fi/mobile/de.htm

[7] T. Fuchs, J. Haber, H.-P. Seidel: MIMIC - A Language for Specifying Facial Animations, Proceedings of WSCG 2004, 2-6 Feb 2004, pp. 71-78.

[8] Z. Liu, Z. Zhang, C. Jacobs, M. Cohen: “Rapid Modeling of Animated Faces From Video”, In Proceedings of The Third International Conference on Visual Computing (Visual 2000), pp 58-67, September 2000, Mexico City

[9] H. Gupta, A. Roy-Chowdhury, R. Chellappa: “Contour based 3D Face Modeling From A Monocular Video”, British Machine Vision Conference, 2004.

[10] Pelachaud, C., Badler, N., and Steedman, M.: “Generating Facial Expressions for Speech”, Cognitive, Science, 20(1), pp.1–46, 1996.

[11] "Faces Everywhere: Towards Ubiquitous Production and Delivery of Face Animation", Igor S. Pandzic, Jörgen Ahlberg, Mariusz Wzorek, Piotr Rudol , Miran Mosmondor, Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia, Norrkoping, Sweden, 2003

[12] Igor S. Pandzic: “Life on the Web”, Software Focus Journal, John Wiley & Sons, 2001, 2(2):52-59.

[13] P. Hong, Z. Wen, T. S. Huang, “Real-time speech driven Face Animation”, in I. S. Pandžić, R. Forchheimer, Editors, “MPEG-4 Facial Animation - The Standard, Implementation and Applications “, *John Wiley & Sons Ltd, 2002.*

[14] "Fast Head Modeling for Animation", W.Lee, N.Magnenat-Thalmann, Journal Image and Vision Computing, Volume 18, Number 4, pp.355-364, Elsevier, March, 2000

[15] Igor S. Pandzic: “Facial Motion Cloning”, Graphical Models journal.