Faces Everywhere:

Towards Ubiquitous Production and Delivery of Face Animation

Igor S. Pandzic[1], Jörgen Ahlberg[2], Mariusz Wzorek[2], Piotr Rudol[2] , Miran Mosmondor[1]

[1]Department of Telecommunications

Faculty of electrical engineering and computing

Zagreb University

Unska 3, HR-10000 Zagreb, Croatia

{Igor.Pandzic, Miran.Mosmondor}@fer.hr

[2]Visage Technologies AB

Tröskaregatan 90, SE-58334 Linköping, Sweden

www.visagetechnologies.com

{jorgen, mariusz, piotr}@visagetechnologies.com

Abstract

While face animation is still considered one of the toughest tasks in computer animation, its potential application range is rapidly moving from the classical field of film production into games, communications, news delivery and commerce. To support such novel applications, it is important to enable production and delivery of face animation on a wide range of platforms, from high-end animation systems to the web, game consoles and mobile phones. Our goal is to offer a framework of tools interconnected by standard formats and protocols and capable of supporting any imaginable application involving face animation with the desired level of animation quality, automatic production wherever it is possible, and delivery on a wide range of platforms. While this is clearly an ongoing task, we present the current state of development along with several case studies showing that a wide range of applications is already enabled.

Keywords: face animation, virtual characters, embodied conversational agents, visual text-to-speech, face tracking, lip sync, MPEG-4 FBA

1 Introduction

The human face is one of the most expressive channels of communication and appears in multimedia contents so universally that we take it for granted. Researchers have been fascinated with the possibility to recreate and animate human-like faces on computers since decades [1]. Early face animation research proposed various models for animating a 3D face model: procedural [8], pseudo-muscle [9] and muscle simulation [10][11] were the main categories. More recently, researchers worked on more realistic face models [12][13][14][24]. In parallel, work progressed on face animation production methods such as visual text-to-speech [6], automatic lip-sync [17], and face feature tracking in video [26][27][28].

Application areas for face animation are expanding from film production and advertising into such diverse areas as games, teleconferencing, messaging [25], news delivery [16], education and commerce [7]. In particular, research on Embodied Conversational Agents [15] is going towards the notion of human-like user interface that we can simply talk to – applications of such technology could be very wide in all kinds of automated services, support, consulting and more.

After three decades of research on face animation, most developed systems are still proprietary and do not talk to each other. It is rare, for example, that a lip sync system from one research group or company can directly drive a muscle-based animated face from another group. Yet this kind of compatibility, together with widespread support on various platforms, is essential for widespread applications. The same face animation content, regardless how it was produced, should be playable on platforms and systems as diverse as mainstream 3D animation tools, PCs, games consoles, set-top boxes and mobile phones (Figure 1).

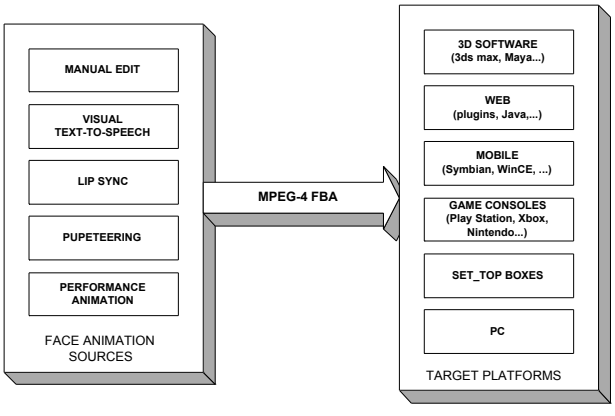

Figure 1: Portability of face animation

This kind of widespread portability is essentially made possible by the recent MPEG-4 Face and Body Animation (FBA) standard [3][4]. In our work, we take this idea one step forward and build a working set of tools that make this promise a reality: the visage framework. It is a set of software

components interfaced through standards such as MPEG-4 and VRML. Subsets of these components are assembled together, or with other 3rd party standard components, to deliver a wide range of applications based on face animation on various platforms. We believe that most currently considered applications can be successfully delivered in this way.

We present an overview of the visage framework in section 2, and describe the various components in sections 1 - 5. Section 6 showcases several case studies that demonstrate the versatility of the framework. The final section brings conclusions and future work ideas.

2 Overview of the visage framework

The visage framework consists of three major categories of modules: face model production, face animation production and multi-platform delivery. Figure 2 shows all modules. In a typical application only a selection of modules is used (see case studies, section 6).

Making a face model and preparing it for animation is typically time consuming. In the visage framework, static face models are imported in standard VRML format from mainstream modelling tools and prepared for animation using the semi-automatic Facial Motion Cloning method. This method essentially copies all necessary morph targets from an existing generic face model to the new face model. An interesting feature is the possibility to make available morph target sets with different animation styles for the generic model, and simply choose which animation style to clone onto the new model (e.g. standard, exaggerated, with wrinkles,…).

The visage framework contains tools for face animation production based on most currently known methods: video tracking, visual TTS, lip sync and manual editing. Each tool will be described in more detail in section 4. All tools produce standard MPEG-4 FBA bitstreams with face animation, making the tools completely interchangeable. Thanks to the standard format, the editing tool can be applied on the results of all other tools, and 3rd party standard tools can easily be incorporated. A particular advantage of the MPEG-4 FBA format is its efficiency – bit rates can be as low as 0.5 kbit/sec if only viseme-based speech animation is used, and typically do not exceed 5 kbit/sec for full animation.

The delivery is based on the very small and portable visage Face Animation Player core. This core exists in both Java and C++ versions, and can easily be ported on top of any software environment supporting 3D graphics, as illustrated in Figure 2.

By selecting appropriate tools, it is fairly straightforward to build applications involving face animation in various environments and platforms.

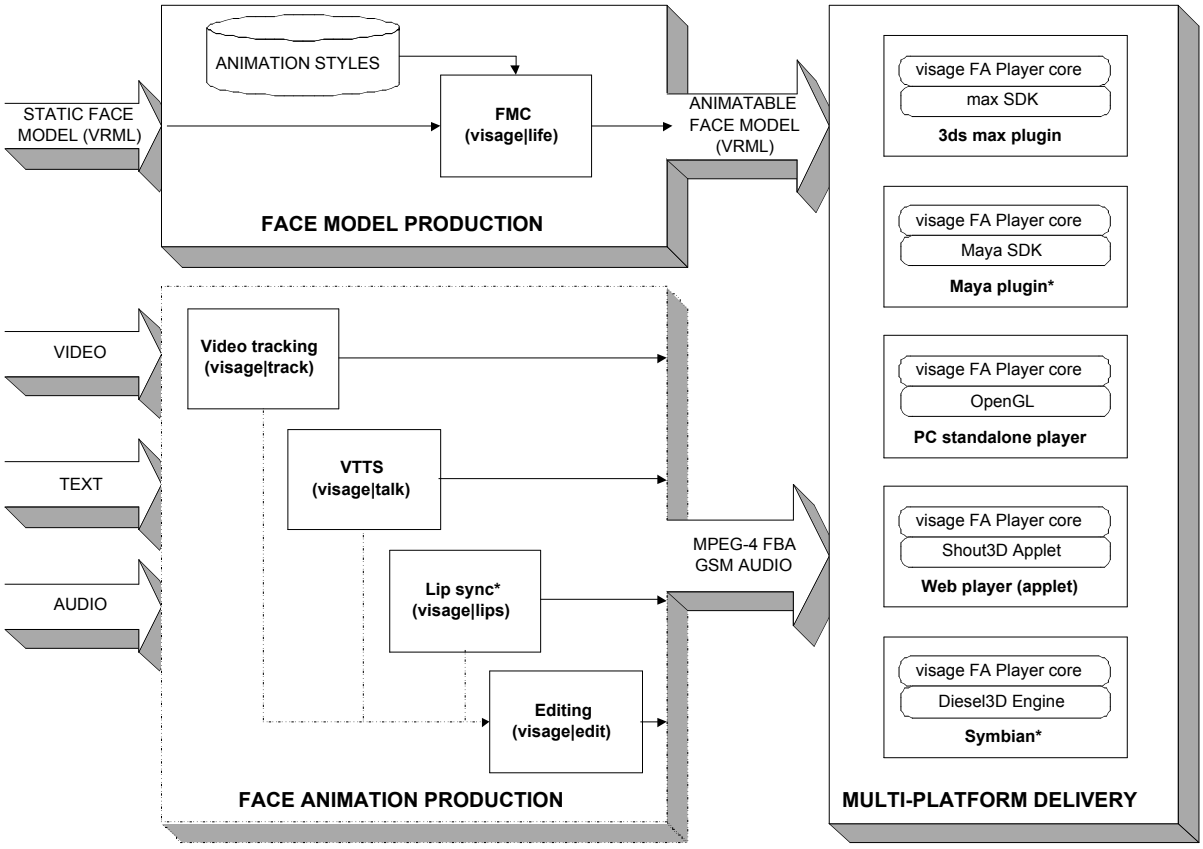

Figure 2: Overview of the visage framework (*currently under development)

3 Face model production

In this section we describe our approach to the production of face models that can be directly animated by all other tools in the described framework. We believe that the most important requirement for achieving high visual quality in an animated face is the openness of the system for visual artists. It should be convenient for them to design face models with the tools they are used to. The concept of morph targets as key building blocks of facial animation is already widely used in the animation community. However, morph targets are commonly used only for high level expressions (visemes, emotional expressions). In our approach we follow the MPEG-4 FAT concept and use morph targets not only for the high level expressions, but also for low-level MPEG-4 FAPs. Once their morph targets are defined, the face is capable of full animation by limitless combinations of low-level FAPs.

Obviously, creating morph targets not only for high level expressions, but also for low-level FAPs is a tedious task. We therefore propose a method to copy the complete range of morph targets, both low- and high-level, from one face to another. The source face with a complete set of morph targets is available, and different sets of morph targets defining various animation styles are being developed, so that a user can choose the animation style to be applied when copying the morph targets to a new face. The method we propose for copying the morph targets is called Facial Motion Cloning. Our method is similar in goal to the Expression Cloning [2]. However, our method additionally preserves the MPEG-4 compatibility of cloned facial motion and it treats transforms for eyes, teeth and tongue. It is also substantially different in implementation.

Facial Motion Cloning can be schematically represented by Figure 3. The inputs to the method are the source and target face. The source face is available in neutral position (*source face*) as well as in a position containing some motion we want to copy (*animated source face*). The target face exists only as neutral (*target face*). The goal is to obtain the target face with the motion copied from the source face – the *animated target face*.

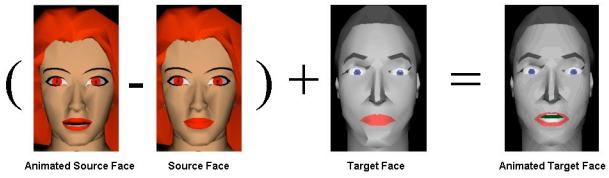

Figure 3: Overview of Facial Motion Cloning

To reach this goal we first obtain *facial motion* as the difference of 3D vertex positions between the animated source face and the neutral source face. The facial motion is then added to the vertex positions of the target face, resulting in the animated target face. In order for this to work, the facial motion must be normalized, which ensures that the scale of the motion is correct. In the *normalized facial space*, we compute facial motion by subtracting vertex positions of the animated and the neutral face. To map the facial motion correctly from one face to another, the faces need to be aligned with respect to the facial features. This is done in the *alignment space*. Once the faces have been aligned, we use interpolation to obtain facial motion vectors for vertices of the target face. The obtained facial motion vectors are applied by adding them to vertex positions, which is possible because we are working in the normalized facial space. Finally, the target face is de-normalized. The procedure is repeated for all morph targets we want to copy. The Facial Motion Cloning method is described with more detail in [5].

4 Face animation production

4.1 Video tracking

Tracking a face, and facial features like lip movements etc, in video is a simple task for the human visual system, but has shown to be a very complex problem for machine vision. There are numerous proposed methods, but quite few have so far reached the market. Our method is based on the Active Appearance Models [29], and offers 3D tracking of the face and important facial features (currently lip and eyebrow motion) in real-time or near real-time. The method is based on a statistical model of facial appearance, finding the most probable pose and deformation (i.e. facial feature motion) of the face in each frame. Technical details can be found in [26]. The current version of the tracking module typically needs to be trained for the person to be tracked, but this step is expected to be removed.

An illustration of the tracking is given in Figure 4. As can be seen, the simple face model used here is automatically adapted to the face in each frame. From the pose and deformation of the model, MPEG-4 FAPs can be computed.

The tracking module inputs a sequence of images and outputs animation parameters in an MPEG-4 FBA bitstream describing the head, lip and eyebrow motion. The animation parameters can then be applied to a model of the recorded face, potentially enabling very-low bitrate video telephony, or any other face model. Another usage is to record head motion to be used as "background motion", to which lip motion from the VTTS could be added, giving the final animation a realistic look.

Figure 4: Automatic face and facial feature tracking. The simple face model adapts to each frame in the sequence (every tenth frame shown), and MPEG-4 FAPs can then be extracted from the model.

4.2 Visual text-to-speech

The visual text-to-speech tool is based on the SAPI standard (SAPI-4 and SAPI-5 versions exist). The SAPI TTS generates events when phonemes are generated and provides timing of the phonemes. Tables are used to convert phonemes into MPEG-4 visemes, and these are encoded into an MPEG-4

FBA bitstream. In the current version co-articulation model is a simple one, using linear interpolation to blend visemes on the boundaries. Non-SAPI TTS systems can be integrated through a simple interface. Any TTS system that can generate phoneme/viseme timing can be connected, and the output is the standard MPEG-4 FBA bitstream.

4.3 Lip sync

A module for automatic lip sync is currently under development. The module inputs digitized speech and, like the VTTS module described above, outputs visemes in an MPEG-4 FBA bitstream. It is based on a method that extracts Mel Frequency Cepstral Coefficients from the audio, and then uses a set of neural networks to classify each audio frame as a certain phoneme. The module can operate in two modes: real-time and batch. In real-time mode, received audio can be played and animated in real-time with a delay of less than 80 ms. In batch mode, the delay is significantly higher, but offers a somewhat higher quality of the lip sync. Technical details will be published in the near future.

4.4 Animation editing

MPEG-4 Facial Animations can be easily manipulated using the visage|edit application. Using this program, animations can be merged or created from scratch. The user interface consists of four panels shown in Figure 5.

The first from the left is the Project Tree panel used for hierarchical selection. The first node of the tree is a virtual track. This track constitutes a reference for the project being edited. Additional nodes represent tracks included in the project. Next is the High-Level Editing panel which shows information in rows according to items visible in the Tree View. Each track in a project is represented by a bar, which can be moved (changes offset) and resized (changes scale in time) using mouse. Additional controls for choosing track manipulations are supplied. The Low-Level Editing panel consists of a plot area for displaying FAP values.

This plot can be scrolled using scroll bars or by "dragging" the plot area with Shift key pressed. There are two directions of zooming. Vertical zoom allows setting the range of FAP values to be shown. Horizontal zoom allows setting the range of frames to be displayed. The last panel is the Preview panel which shows a face model at all time for a user to see changes being done to the project animation.

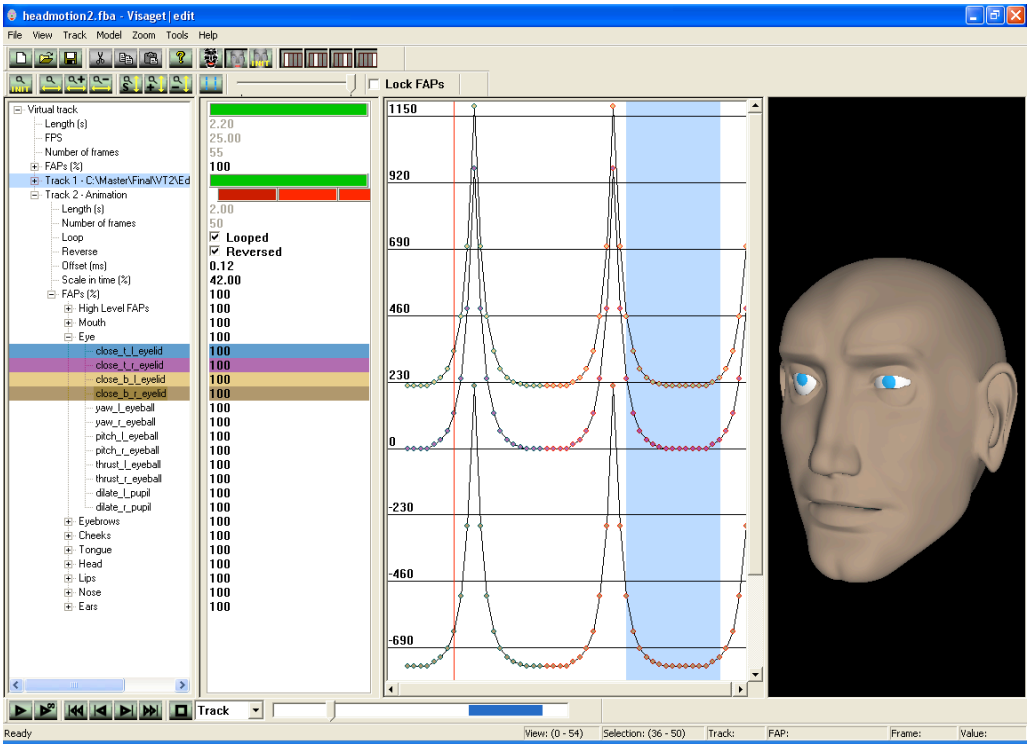

Figure 5: visage|edit application.

In order to enable users to manage projects consisting of several animations, the idea of a tree view was introduced. It makes it possible to display information about several tracks on different levels of detail at the same time. Additionally, the tree is a common reference for both high- and low-level panels – information shown in the two latter views corresponds to options chosen in the former.

The high-level mode allows editing tracks without going into details of FAP values. This mode can be used to manipulate already existing tracks. The obvious example of this mode is merging two animations in order to add eye blinks to a track obtained from the Visage|track application. Operations that can be performed in the high-level mode include adding offset, scaling, looping and reversing the track and additionally this mode allows applying multiplication factors on all FAPs, groups of FAPs and, on separate FAPs. The final value of the multiplication factor for a separate FAP is the product of the three multiplication factors and a corresponding factor from the "virtual track".

The low-level mode allows direct manipulation of FAP values. This mode is useful for fine-tuning the parameters of an animation and when "producing" new animation tracks, for example creating an eye blink track from scratch. Such an eye-blink can be looped to produce the eyelids' movement for the whole animation.

5 Multi-platform delivery

Multi-platform delivery, and the capability to implement support for virtually any platform in a very short time, is one of the strongest points of the visage framework. 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 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. 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 (Figure 2). For example, the Java applet implementation, based on the Shout3D rendering engine [18], 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 [19]. Other implementations include a PC standalone version based on OpenGL, 3ds max and Maya plugins and an implementation on a Symbian platform for mobile devices (last two currently in development).

Implementation of the face animation player on Symbian platform for mobile devices is written as C++ application and based on DieselEngine [31]. Because of low CPU time consumption and low memory requirements, MPEG-4 FBA decoder can be used almost unaltered on mobile device. Most differences were concerning 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 a low level API (Application Program Interface) that is similar to Microsoft DirectX and high level modules had to be implemented. The most important is the 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. 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 3700 polygons.

6 Case studies

The classical usage of the presented framework would be the film/video production. In this scenario, the face model is prepared using the FMC module, one or more animation production modules are used to prepare the face animation, and then the model and the animation are imported into a mainstream 3D animation package (e.g. 3ds max or Maya), incorporated into a scene and rendered. However, this is just one application scenario and there are many other potential applications that can be built based on the visage framework. In the next sections we will briefly describe several experimental applications that have been implemented.

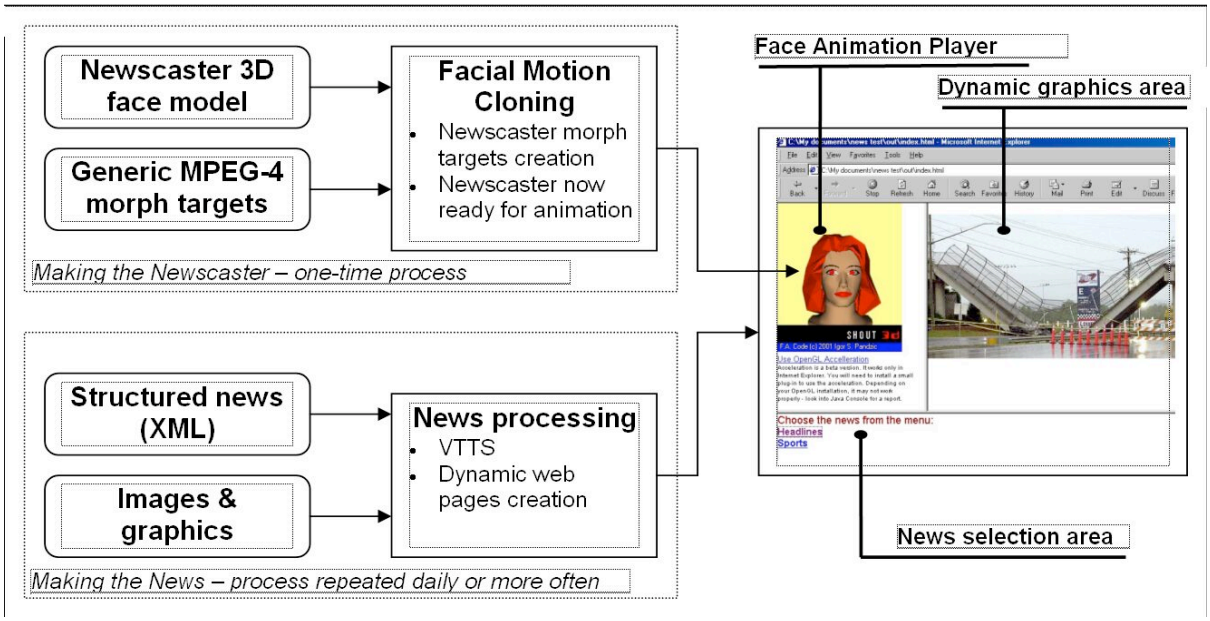

Figure 6: The virtual newscaster system architecture

6.1 Virtual newscaster

We have built a prototype of an interactive multimedia news system featuring a talking virtual character to present the news on the Web [16]. The virtual character is used as a newscaster, reading the news on the Web while at the same time presenting images and graphics. The users choose the news topics they want to hear. The content of the news is defined in an XML file, which is automatically processed to create the complete interactive web site featuring the virtual newscaster reading out the news. This allows for very frequent automatic updates of the news. The virtual character is animated on the client using a Java applet implementation of the visage face animation player, requiring no plug-ins. The bandwidth and CPU requirements are very low and this application is accessible to a majority of today's Internet users without any installation on the end-user computer. We believe that the presented news system combines qualities of other current news delivery systems (TV, radio, web sites) and therefore presents an attractive new alternative for delivering the news.

Figure 6 illustrates how components of the visage framework (Facial Motion Cloning, VTTS, Face Animation Player) are used together with application-specific components to deliver this application.

6.2 Talking email/SMS

Talking email combines a VTTS module on the server with the web version of the face animation player. It is a web application that allows the visitors of a web site to compose and send talking email straight from the web site, offering a great entertainment value. A talking email is a web page containing an interactive virtual person that talks, i.e. pronounces the email message. The sender inputs the message text and chooses the virtual person to deliver it. The speech and animation are generated on the server and the sender can immediately preview the talking email message, then simply input the email address and send it. The receiving party sees the talking virtual character in a web page delivered by email.

The SMS interface allows sending talking email messages from a mobile phone by SMS. Current development is going towards the delivery of the talking email directly on mobile phones, either through MMS or through full face animation player application running on the mobile phone.

6.3 Embodied Conversational Agents

Embodied Conversational Agents are virtual characters coupled with artificial intelligence (AI) techniques to deliver the impression of a live person that can lead a meaningful conversation. Such virtual characters are expected to represent the ultimate abstraction of a human-computer interface, the one where the computer looks, talks and acts like a human.

This would include audio/video analysis and synthesis techniques coupled with AI, dialogue management and a vast knowledge base in order to be able to respond quasi-intelligently to the user – by speech, gesture and even mood [22]. While this goal lies further on in the future, we present an architecture that reaches towards it, at the same time aiming for a possibility of practical applications in nearer future. Our architecture is aimed specifically at the Web.

Our system uses A.L.I.C.E. [20] as the AI server. It is based on Case-Based Reasoning or CBR, first used by ELIZA [21]. The AI server takes text as input, and outputs reasonable answers in form of text based on the A.L.I.C.E. knowledge base. The user interface is a web page with an integrated virtual character and text input field (Figure 7). When the user enters some text, it is sent to the server where it is first processed by the AI server in order to obtain an answer from the AI engine. The answer is then sent to the VTTS module which generates the speech and appropriate face animation; these are returned to the client and played.

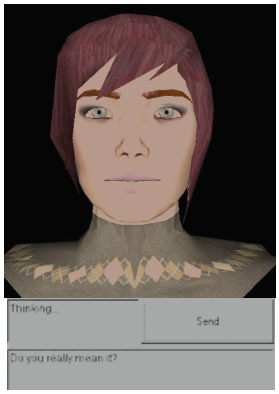

Figure 7: Embodied Conversational Agent

7 Conclusions and future work

We have introduced a framework for ubiquitous production and delivery of face animation and presented case studies showing how this framework can be configured into various applications. Work is ongoing on the development of new modules of the framework, specifically to support new platforms and improve the existing modules. In particular, the VTTS module is being extended with an automatic gesturing function that should produce natural-looking facial gestures based on lexical analysis of the text and a statistical model based on analysis of a training data set, similar to [23]. In parallel, new applications are being developed, in particular on mobile phones where we expect such innovative applications to have a great entertainment value.

8 References

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

[2] Jun-yong Noh, Ulrich Neumann: "Expression Cloning", Proceedings of SIGGRAPH 2001, Los Angeles, USA.

[3] ISO/IEC 14496 - MPEG-4 International Standard, Moving Picture Experts Group, www.cselt.it/mpeg

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

[5] Igor S. Pandzic: "Facial Motion Cloning", accepted for publication in the Graphical Models journal.

[6] C. Pelachaud, "Visual Text-to-Speech", in "MPEG-4 Facial Animation - The standard, implementations and applications", I.S. Pandzic, R. Forchheimer (eds.), John Wiley & Sons, 2002.

[7] J. Ostermann, D. Millen, "Talking heads and synthetic speech: An architecture for supporting electronic commerce", Proc. ICME 2000

[8] F.I. Parke: "A Parametric Model for Human Faces", PhD Thesis, University of Utah, Salt Lake City, USA, 1974. UTEC-CSc-75-047.

[9] Kalra P., Mangili A., Magnenat-Thalmann N., Thalmann D.: "Simulation of Facial Muscle Actions based on Rational Free Form Deformation", Proceedings Eurographics 92, pp. 65-69.

[10] S.M. Platt, N.I. Badler: "Animating Facial Expressions", Computer Graphics, 1981, 15(3):245-252.

[11] K. Waters: "A muscle model for animating three-dimensional facial expressions", Computer Graphics (SIGGRAPH'87), 1987, 21(4):17-24.

[12] Y. Lee, K. Waters, D. Terzopoulos: "Realistic modeling for facial animation", in Computer Graphics (SIGGRAPH '95 Proceedings), 55-62

[13] B. Guenter, C. Grimm, D. Wood: "Making Faces", in Computer Graphics (SIGGRAPH '98 Proceedings), 55-66

[14] V. Blanz, T. Vetter: "A morphable model for the synthesis of 3D faces" , in Computer Graphics (SIGGRAPH '99 Proceedings), 75-84

[15] Embodied Conversational Agents, edited by Cassell J., Sullivan J., Prevost S., Churchill E., The MIT Press Cambridge, Massachusetts London, England, 2000.

[16] "An XML Based Interactive Multimedia News System", Igor S. Pandzic, 10th International Conference on Human - Computer Interaction HCI International 2003, Crete, Greece

[17] M. Brand, "Voice Puppetry", Proceedings of SIGGRAPH'99, 1999.

[18] Shout 3D, Eyematic Interfaces Incorporated, http://www.shout3d.com/

[19] Igor S. Pandzic: "Facial Animation Framework for the Web and Mobile Platforms", Proc. Web3D Symposium 2002, Tempe, AZ, USA, demonstration at www.tel.fer.hr/users/ipandzic/MpegWeb/index.html

[20] Artificial Linguistic Internet Computer Entity, http://www.alicebot.org

[21] Weizenbaum, J., "ELIZA - A computer program for the study of natural language communication between man and machine", Communications of the ACM 9(1): 36-45, 1966.

[22] The InterFace project, IST-1999-10036, www.ist-interface.org

[23] S. P. Lee, J. B. Badler, N. I. Badler, "Eyes Alive", Proceedings of the 29th annual conference on Computer graphics and interactive techniques 2002 , San Antonio, Texas, USA, ACM Press New York, NY, USA, Pages: 637 – 644

[24] Eric Cosatto, Hans Peter Graf: Photo-Realistic Talking-Heads from Image Samples. IEEE Transactions on Multimedia 2 (3): 152-163 (2000)

[25] J. Ostermann, "PlayMail: The Talking Email", in "MPEG-4 Facial Animation - The standard, implementations and applications", I.S. Pandzic, R. Forchheimer (eds.), John Wiley & Sons, 2002.

[26] J. Ahlberg and R. Forchheimer, "Face Tracking for Model-based Coding and Face Animation," Int Journal of Imaging Systems and Technology, 13(1):8-22, 2003.

[27] M. La Cascia, S. Sclaroff, and V. Athitsos, "Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models," IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(4):322-336, 2000.

[28] C. S. Wiles, A. Maki, and N. Matsuda, "Hyperpatches for 3D Model Acquisition," IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(12):1391-1403, 2001.

[29] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(6):681-685, 2001.

[30] V. Mäkinen, Front-end feature extraction with mel-scaled cepstral coefficients, technical report, Laboratory of Computational Engineering, Helsinki University of Technology, September 2000.

[31] http://www.inmarsoftware.com/diesel.htm