Mobile Information Systems 2 (2006) 95–110
IOS Press

Personalized avatars for mobile entertainment

Tomislav Kosutic[a], Miran Mosmondor[a,*], Ivan Andrisek[a], Mario Weber[b], Maja Matijasevic[c] and Igor Pandzic[c]
[a]*Ericsson Nikola Tesla, Krapinska 45, HR-10000 Zagreb, Croatia*
E-mail: {tomislav.kosutic,miran.mosmondor,ivan.andrisek}@ericsson.com
[b]*KATE-KOM, Drvinje 109, HR-10000 Zagreb, Croatia*
E-mail: mario.weber@kate-kom.com
[c]*University of Zagreb, FER, Unska 3 HR-10000 Zagreb, Croatia*
E-mail: {maja.matijasevic,igor.pandzic}@fer.hr

Abstract. With evolution in computer and mobile networking technologies comes the challenge of offering novel and complex multimedia applications and end-user services in heterogeneous environments for both developers and service providers. This paper describes one novel service, called LiveMail that explores the potential of existing face animation technologies for innovative and attractive services intended for the mobile market. This prototype service allows mobile subscribers to communicate using personalized 3D face models created from images taken by their phone cameras. The user can take 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 automatically by speech synthesis. Furthermore, these highly personalized animations can be sent to others as real 3D animated messages or as short videos in MMS. The clients were implemented on different platforms, and different network and face animation techniques, and connected into one complex system. This paper presents the architecture and experience gained in building such a system.

Keywords: Face animation, virtual characters, MPEG-4 FBA, visual text-to-speech, 3D modelling, 3D graphics

1. Introduction

The two most dynamic forces in communications technology today are certainly mobility and the Internet. 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. The number of people connected to the Internet is continuing to increase and new generations of mobile networks are enabling connectivity virtually everywhere and at any time with any device. With 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.

Graphical simulations of real or imaginary persons capable of talking and gesturing have been an active research area for a long time. The advances in animation systems have prompted interest in use of animation to enrich the human-to-computer interface and also human-to-human communication. Together with ever more widespread multimedia and multimodal end-user equipment, ranging from

*Corresponding author: Miran Mosmondor, Ericsson Nikola Tesla d.d., Krapinska 45, P.O. Box 93, HR-10000 Zagreb, Croatia. Tel.: +385 1 365 3492; Fax: +385 1 365 3548; E-mail: miran.mosmondor@ericsson.com.

devices specifically designed for a particular purpose to generic laptops and mobile phones, a wide range of new applications for face animation systems may be envisioned.

The goal of the project described in this chapter was to explore the potential of existing face animation technologies [7] for innovative and attractive services intended for the mobile market, exploiting in particular the advantages of technologies like 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. In this paper we present architecture of LiveMail system. We describe how unique personalized virtual characters are created with our face adaptation procedure. Also, we describe clients that are implemented on different platforms, most interestingly on mobile platform, since 3D graphics on mobile platforms is in its early stages. Various network and face animation techniques were connected into one complex system and we presented the main performance issues of such a system. Also, if correctly implemented, face animation systems have low bandwidth requirements and high interaction capacity, making them a natural replacement for video streaming in "talking head" scenarios. Combined with facial feature tracking, technology presented in this chapter could be used for very low bandwidth visual communication.

The chapter is organized as follows. In the next section we give a brief introduction on used standards and technologies. Next, we present overall system functionalities and architecture continuing with more details on server and client implementation in the following sections.

2. Background

Before going further into the in-depth description of LiveMail system, some additional details on used technology and standards are given. The purpose of this section is not to give in-depth coverage of particular standards and technologies but to give enough background information for understanding the rest of the chapter.

2.1. 3D face modeling and visualization

Creating personalized 3D face models is not as simple task as it seems. There are various techniques that can be used. However, most of them can not be applied on mobile terminals nor they are appropriate for end-user usage. For example, utilization of 3D modeling tool such as 3D Studio Max or Maya and manual construction of 3D models is often expensive, time-consuming and it sometimes does not result with a desirable model. Another way is to use specialized 3D scanners. In this way face models can be produced with very high quality but using them in a mobile environment is not practical. Also, there have been methods that included use of two cameras placed at certain angle and algorithms for picture processing to create 3D model [19]. Some other methods, like in [5], use three perspective images taken from different angles to adjust deformable contours on a generic head model.

Our approach in creating animated personalized face models is based on the adaptation of an existing generic face model, similar to [18]. However, in order to achieve simplicity on a camera-equipped mobile device, our adaptation method uses a single picture as an input.

Another important issue, besides 3D face modeling, is the 3D visualization on mobile devices [13]. Because of the vast computational power required to achieve a usable performance, 3D graphics rendering on handheld devices can still present a problem. Hardware and software capabilities of 3D graphics on handheld devices are clearly smaller than on desktop computers in many ways. Lower speed, smaller display size and resolution, less memory for running and storing programs, are just some of the

restrictions. With the introduction of 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 solutions for mobile devices a few months after.

Beside hardware solutions, another important prerequisite 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. The efforts made by the Khronos Group were clearly one of the biggest steps towards this goal. The Khronos Group is the industry consortium founded by a number of leading media-centric companies in January 2000. One of open standard APIs they have created is the OpenGL ES (OpenGL for Embedded Systems) announced in July 2003. It is a royalty-free, cross-platform low-level API for full-function 2D and 3D graphics on embedded systems.

Another important effort was Mobile 3D Graphics API (JSR-184) specification developed under the Java Community Process. It was published by the end of October 2003. JSR-184 (also known as M3G) is a high-level API for Java mobile 3D graphics and it is designed to provide an efficient 3D graphics API suitable for the J2ME platform, and, in particular, for the CLDC/MIDP profile. OpenGL ES low-level API and M3G high-level API represent de facto industry standards in area of 3D graphics API for mobile devices.

2.2. MPEG-4 Facial Animation

Creating animated human faces using computer graphics techniques has been a 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, salespersons, actors, and newsreaders [8].

Created personalized face model can be animated using speech synthesis [15] or audio analysis (lip synchronization) [14]. Our face animation system is based on the MPEG-4 standard on Face and Body Animation (FBA) [9,10]. 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) (Fig. 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.

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 movements of feature points in the face, and MPEG-4 defines 84 such points (Fig. 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

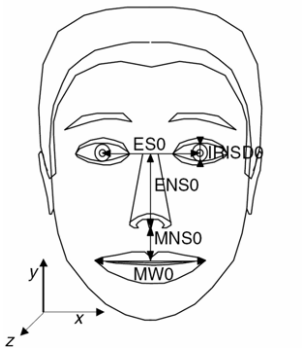

Fig. 1. A face model in its neutral state and defined FAP Units (FAPU) (ISO/IEC IS 14496-2 Visual, 1999).

of visemes. A viseme is a visual correlate to a phoneme. It specifies shape of a mouth and tongue for minimal sound unit of speech. In MPEG-4 there are 14 static visemes that are clearly distinguished and included in the standard set. For example, phonemes /p/, /b/ and /m/ represent one type of viseme. Important thing for their visualization is that the shape of the mouth of a speaking human is not only influenced by the current phoneme, but also by the previous and the following phoneme. In MPEG-4, transitions from one viseme to the next are defined by blending only two visemes with a weighting factor. The expression parameter allows definition of high-level facial expressions.

The FAPs can be efficiently compressed and included in a Face and Body Animation (FBA) bit stream for low bit rate storage or transmission. An FBA bit stream can be decoded and interpreted by any MPEG-4 compliant face animation system [2,12], and a synthetic, animated face be visualized.

3. System functionalities

After a short review on technologies and standards, we present basic functionalities of LiveMail service. LiveMail service provides a simple creation of personalized face model, transmission and display of such virtual character as animated message 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.

Once a user creates a personalized face model, he/she can communicate with others by sending animated messages of that face model. Animated message contains a speech animation synthesized from text that user inputs.

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. Capabilities of mobile devices have improved in last few years, but they are still clearly not capable of such a demanding computing. Therefore all time-consuming processes are done on one or more servers while mobile devices are clients. A simplified LiveMail use-case scenario is depicted in Fig. 3.

Server's basic task is to perform computationally expensive processes, like previously mentioned personalization of virtual character and creation of animated message with speech and lips synchronization. The client side is responsible for displaying the animated message and handling user requests for the creation of a new personalized virtual character and its animated message through proper user interface.

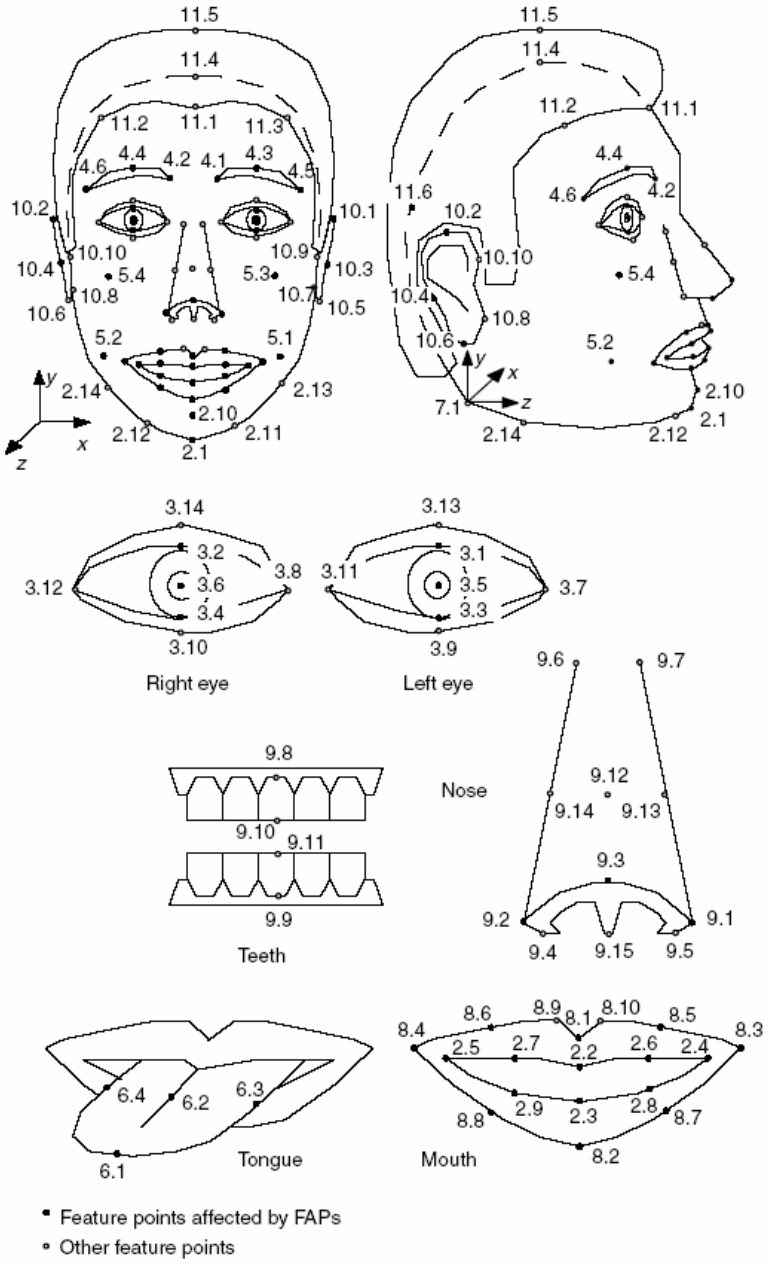

Fig. 2. Face feature points (ISO/IEC IS 14496-2 Visual, 1999).

When creating a personalized character, the interface allows user to input a facial image and place a simple mask on it to mark the main features. Although this could be done automatically recognition of face features is not the scope of this work. After selecting main face features the client uploads the picture and face features parameters to the server. The server then builds and stores the personalized face model, and then notifies the client (Fig. 4).

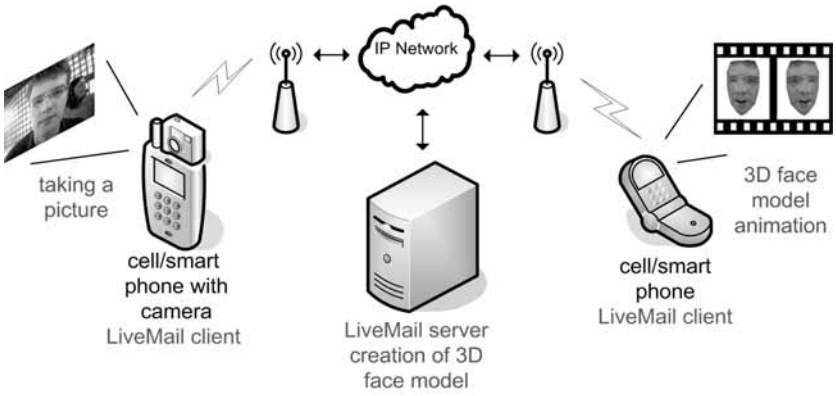

Fig. 3. Simplified LiveMail use-case scenario.

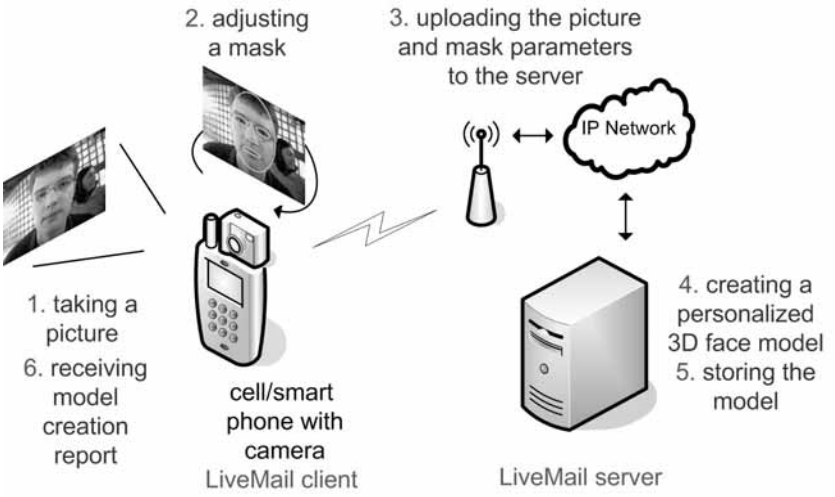

Fig. 4. Personalized 3D face model creation.

When creating animated messages, the user selects a previously created 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. Because the animated message preview completely depends on recipient's phone capabilities, animation is adjusted prior to sending to the recipient. For mobile devices that cannot display even medium quality 3D animation, animated message is converted to a short animated GIF that only contains essential frames of the animation. Animated GIF is sent to the recipient as MMS (Fig. 5).

4. System architecture

LiveMail system architecture is based on a client/server communication model. Tasks that the server must perform are very time-consuming; to be more precise it takes up to 10 seconds to build a new personalized virtual character on today's standard PC (CPU 2-3 GHz, 512 MB RAM). To establish LiveMail as a broad-based service that serves many users simultaneously the server application must be

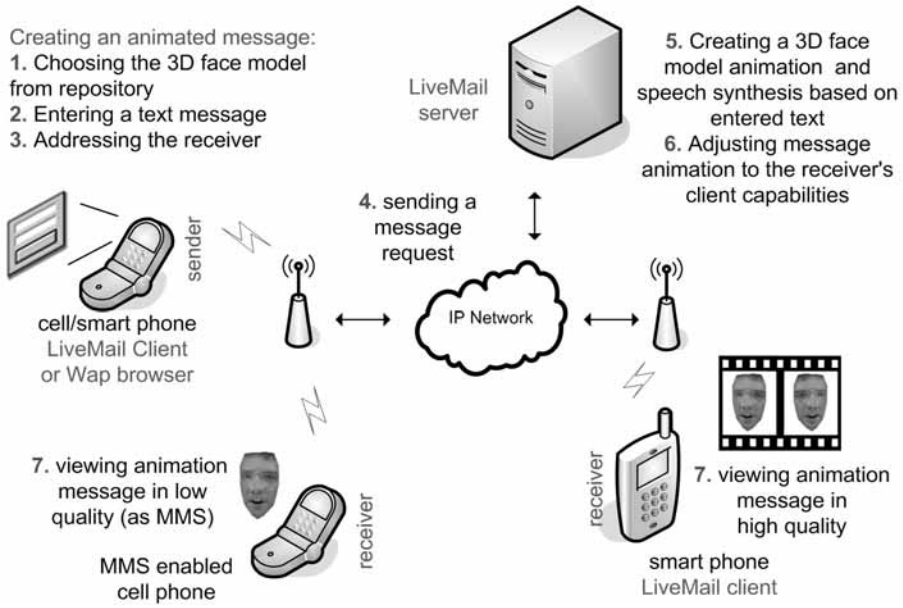

Fig. 5. Creating an animated message through client user interface.

modular and scalable in design. Today, there are many platforms specifically intended for developing and deploying distributed component based enterprise applications. Some popular platforms are Java 2 Enterprise Edition, Distributed Component Object Model (DCOM) and .NET Framework. Implementing LiveMail on one of these platforms will meet previously mentioned requirements [4].

A generic LiveMail system architecture is depicted in Fig. 6. It is a 3-tier architecture that has a thick client and a scalable application server.

3D face model adaptor module is used to create personalized virtual characters, while 3D face model animator and text-to-speech engine modules are used to create animated messages. 3D face model OpenGL renderer module is used to create virtual character's preview images for client user interface. It is also used in a process of converting animated message to animated GIF. Communication module is responsible for communicating with MMS service center (MMSC) and LiveMail clients. Connection to MMSC is established through MM7 protocol while LiveMail client communicates with the server by a simple text protocol built on top of HTTP. Some other application protocols like XML-RPC or SOAP could be used for client/server communication, but HTTP is more widespread and has minimal data overhead [4].

5. LiveMail server

While described architecture is best for commercial deployment of LiveMail service, as an enterprise application it is very complex and hard to implement. Therefore, the first step in that direction was to build a prototype that would have the same modules and share the concept with the described architecture, only in smaller proportions.

The prototype of a LiveMail server is a single application (written in C++) that consists of a lightweight HTTP server and an application server. HTTP server provides clients with user interface and receives requests, while application server processes client requests: creates new personalized 3D face models

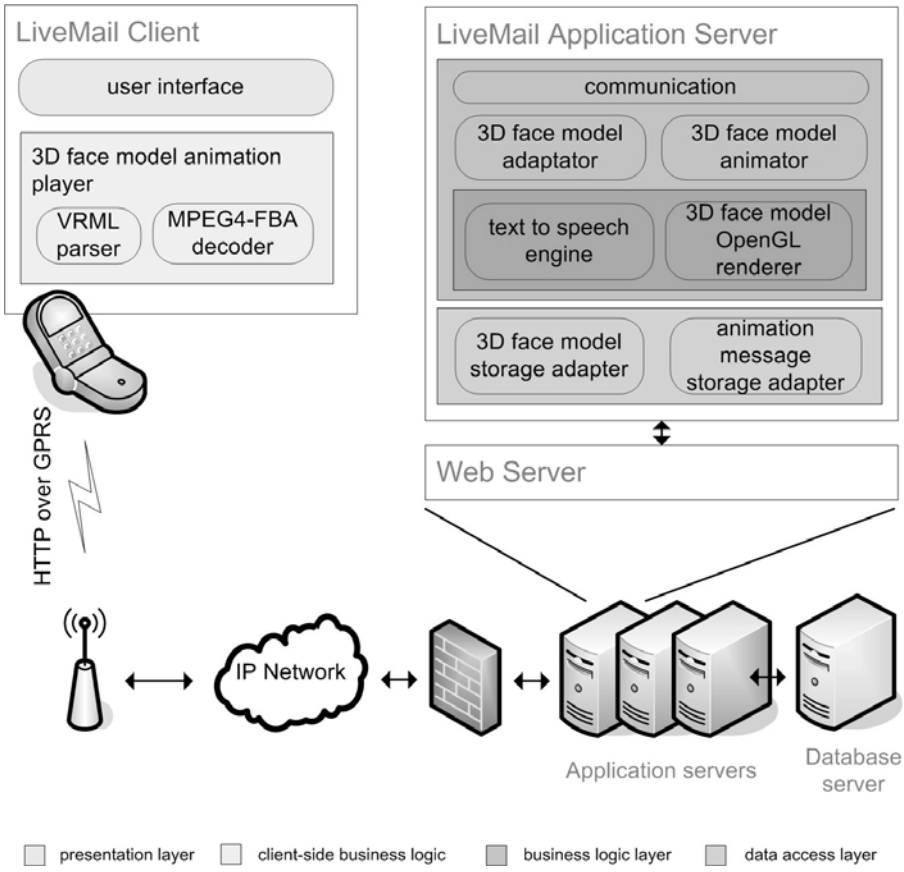

Fig. 6. LiveMail system architecture.

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. As every server, LiveMail is naturally multithreaded. 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 the base for 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.

Server's most valuable task is virtual character adaptation. The client's request for new virtual character creation holds entry parameters for the adaptation process: the picture of a person whose 3D model is created and the characteristic facial points in that picture (creating face mask). 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 model ready for animation. Model is stored on the server for later use in VRML format.

The virtual character adaptation algorithm starts by mapping characteristic facial points from a facial

picture into characteristic points of a corresponding generic 3D face model. Initial mapping is followed by three main stages of adaptation. The first stage is normalization. The highest and the lowest point of the generic 3D face model are modulated according to the characteristic facial points in the facial picture.

The generic 3D face model is then translated to the modulated highest point. As a rule, the translated model does not suit the size of facial picture mask so it has to be scaled. Thus, vertical ratio of the generic model and facial picture mask is calculated and every point of the generic model moved in corresponding ratio. There is vertical and horizontal scaling. Horizontal scaling involves adjusting the horizontal distance between the highest point and every other characteristic facial point, relative to face axis symmetry. Vertical scaling is easier, because there is no axis symmetry.

The next stage is processing. There is texture processing and model processing. N existing points map out a net of triangles that covers 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 [6]. We forward characteristic points received from client's request to interpolator. Interpolator (based on triangles net) will determine locations of other points needed for the creation of the new model.

The last stage of the algorithm is renormalization, and it is practically identical to the first stage of the algorithm, except it involves rolling back the model from normalized space back to space prior to starting of the algorithm.

Described adaptation algorithm is very time consuming. Measuring each step of new virtual character creation shows that 98% of entire process goes on adaptation algorithm. The rest (2%) is spent on model storage and preview images for client user interface. With the generic 3D face model constructed of approx. 200 polygons, adaptation algorithm takes 7.16 seconds on an AMD Athlon XP 2800+ (Fig. 7). Each adaptation algorithm started by client's request is run in a separate thread so multiprocessor systems can handle simultaneous client requests much faster.

Server's next important task is creation of animated messages. Its execution time depends on message text length (Fig. 8). Also, Microsoft Text-to-Speech engine can not process simultaneous text-to-speech conversions so all client requests are handled one at the time (while all others wait in a queue).

6. LiveMail client

LiveMail client can be easily implemented on various client platforms because the animation is generated on the server. Our approach was to build an independent face animation player core that can be ported to any platform that supports 3D graphics.

The face animation player is MPEG-4 FBA decoder that decodes MPEG-4 Facial Animation Parameters (FAPs). After decoding FAPs are applied to face model. Interpolation from key positions is used as a facial animation method, similar to morph target approach in computer animation and the MPEG-4 Face Animation Tables (FAT) approach. FATs actually define spatial deformation of a model as a function of the amplitude of the FAPs. We took the interpolation approach because it is very simple to implement and therefore it could be easily ported to various platforms. Also it could be easily adapted to the computer animation because of similar methodology.

Here is how the player really works. Each FAP (both low- and high-level) is defined as a key position of the face, or morph target. Each morph target is described as 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

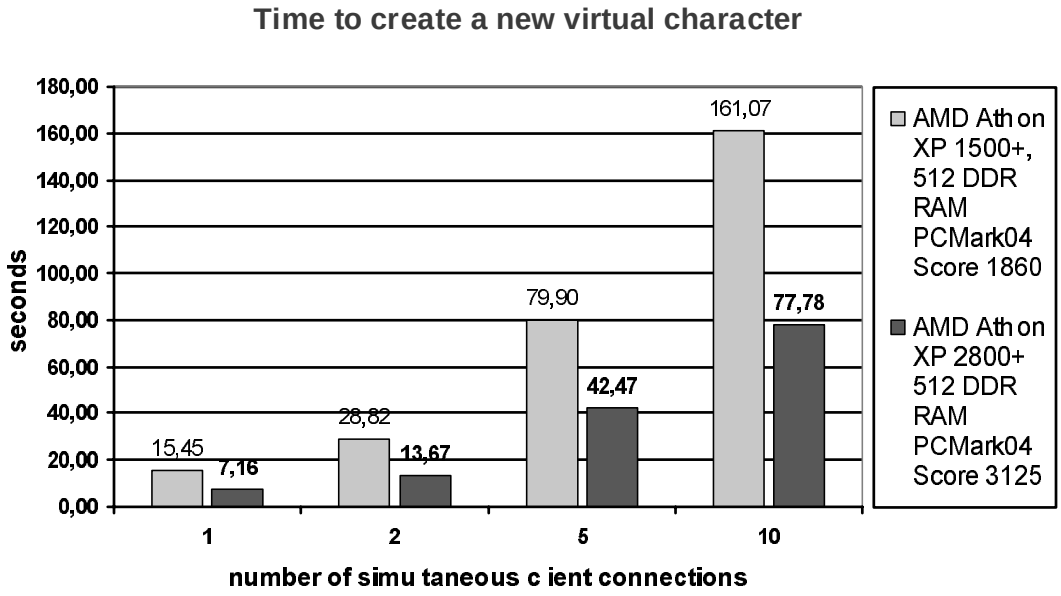

Fig. 7. Time to create a new virtual character in respect to number of simultaneous client connections.

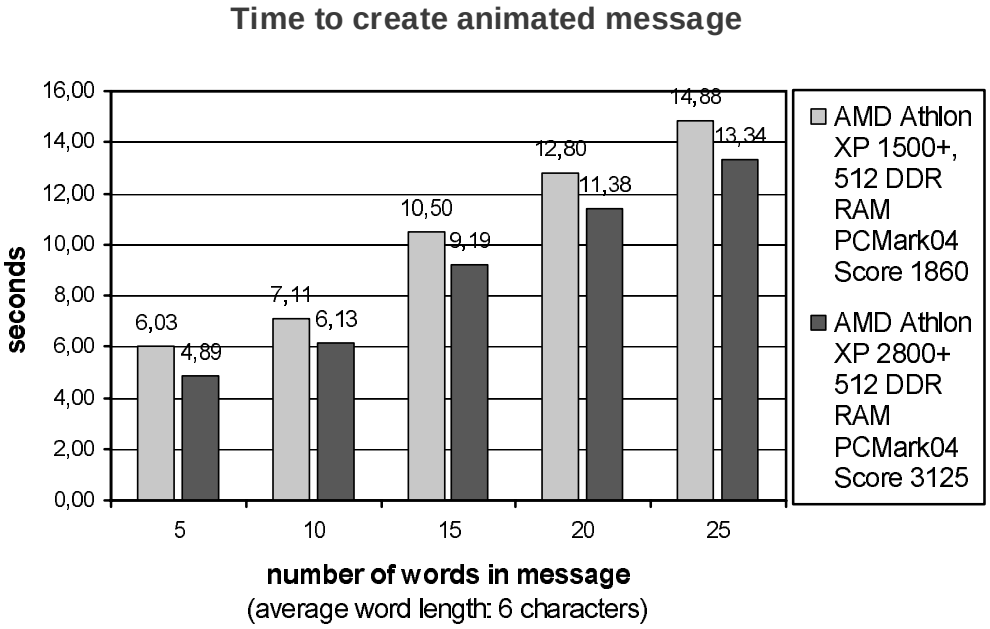

Fig. 8. Time to create an animated message in respect to number of words in message.

node in the scene graph of the face. They are 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. However, current simple linear implementation seems to be

Fig. 9. 3D face model creation and preview on Symbian platform.

sufficient in all situations tested 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.

Various programming languages are used to implement face animation player on a variety of platforms. Implementation of the face animation player is easy due to its simplicity and low requirements.

6.1. LiveMail client for mobile platforms

Usage of LiveMail system on different platforms is one of its key features. Mobile platforms are in many ways restrictive in comparison to desktop PCs. Lower processing speed, small display size, storage and memory limitation are just some of the main disadvantages.

As mentioned earlier, there are three distinguish use cases for LiveMail service. First is creation of a face model with snapshot of someone face. After taking a photo with camera, the user needs to adjust the mask with key face part outlined. The mask is used to define 26 feature points on the face that are then, together with picture sent to the server for face adaptation, as described previously. The user interface that is used to create personalized face model requires access to some additional services of the phone. For this reason it is implemented on Java 2 Micro Edition (J2ME) platform that has support for additional package, called Mobile Media API (MMAPI). MMAPI allows access to native multimedia services like, for example, camera manipulation and picture access that is not granted with standard J2ME platform. Also, this user interface is implemented for Symbian platform where it is used in the same way as on J2ME platform (Fig. 9).

Second case is the process of creating and sending the LiveMail message with previously generated personalized characters. Because of the low requirements in the second case, the functionality can be implemented on virtually any of the today's mobile phone platform. In the most restrictive case, the LiveMail WAP service can be used.

Third use case is the previewing and receiving the animated message on the client platform. This is most demanding use case since it requires implementation of the face animation player. Face animation player has to be able to decode a face animation stream and render 3D face model with corresponding animation. This functionality for LiveMail client on mobile platform is currently implemented as a

Table 1
Supported nodes and affiliated fields list compliant to VRML97 specification in LiveMail client for Symbian platform

Node type	Node name	Supported fields
Grouping	Group	Children
	Transform	center, children, hidden, rotation, scale, translation
Common	Shape	appearance, geometry
Sensor	TouchSensor	Enabled
Geometry	Box	Size
	Cone	bottomRadius, height, side, bottom
	Cylinder	bottom, height, radius, side, top
	IndexedFaceSet	coord, coordIndex, texCoord, texCoordIndex, color, normal
	Sphere	Radius
Geometric Properties	Color	Color
	Coordinate	Point
Appearance	Appearance	material, texture, textureTransform
	Material	ambientIntensity, diffuseColor, emissiveColor, shininess, specularColor, transparency
Interpolators	CoordinateInterpolator	key, keyValue
	OrientationInterpolator	key, keyValue
Bindable	Background	skyColor
	Viewpoint	fieldOfView, orientation, position

standalone C++ application for Symbian OS (Fig. 9). For the clients that are not powerful enough to render 3D animations there is an alternative: the animations can be pre-rendered on the server and sent to the clients as MMS messages containing short animated GIF images.

The face animation player on Symbian platform for mobile devices is based on DieselEngine. The DieselEngine 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 a 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.

The VRML parser is the most important part of application. It is the bridge between high-level interface of VRML format and low level DieselEngine® API. Its main task is to covert 3D content from VRML file to Diesel3D scene format. Building a complete VRML parser is a long process and for face animation player complete parser is not needed. Rather, subset of VRML objects that are enough to render face model are supported. Beside support for primitive objects like box, sphere, cylinder and cone this VRML parser has support for arbitrary 3D mesh shapes that are defined with *IndexedFaceSet* VRML nodes. In addition, it is also capable of applying textures from .jpg, .bmp or .gif file format. Detailed list of supported nodes and fields is given in Table 1.

Second functionality that needed to be implemented was interaction. The module for interaction implements navigation and manipulation with the virtual camera in 3D scene functionalities and enables selecting particular part of a virtual face.

For manipulation with the virtual camera, basic matrix transforms were used. Using and combining these matrices navigation types roll, move and zoom were implemented. Roll is a navigation type that rotates face around x- and y-axis in a camera coordinate system. Move and zoom are navigation types that basically translate virtual face parallel to x-, y- and z-axis of a camera coordinate system. Possibility for such manipulation with virtual face is major difference between other similar animated messages implementations that are using facial morphing in just two dimensions.

Selecting particular part of a face that could for example result in forming particular face gesture is an interesting feature for face animation player. Currently it is implemented on lower level where it enables

selecting subset of triangles of which virtual face is formed.

Generally speaking, picking or selecting objects in 3D scene with a device pointer is not as easy as it seems at first glance. There are several ways of selecting objects like ray casting, color-coding, name lists, etc. Each of these methods tests some kind of intersection with 3D objects. DieselEngine® does not have any function for testing intersections so that has to be done manually. For face animation player on Symbian platform a rendering method was used [16].

In the rendering method each 3D object, or to be precise, every triangle of its face mesh is transformed into screen coordinates. So, intersection is tested in a 2D coordinate system. Testing intersection with each triangle of the object is time-consuming and to speed things up bounding volumes were used. In this case we used a bounding sphere. Instead of testing intersection with a couple of dozen triangles for each object, intersection is tested with the one bounding sphere what is very simple to implement using a rendering method. However, the bounding sphere formation itself is not as clear-cut because its centre and radius have to be determined in such a way that with minimum radius all vertices are inside of the bounding sphere. For primitive objects like sphere, box, cone or cylinder this is simple to determine.

For other complex objects or face meshes there are a number of algorithms that perform this task, and these have speed versus quality tradeoffs. For this implementation the Ritter's algorithm [11] that creates a near-optimal bounding sphere was used. If there is a multiple intersection, a value in Z-buffer is compared and the nearest point is selected.

6.2. LiveMail client for desktop PCs

A parallel web service is offered as a full featured interface towards LiveMail system: it is possible to create new personalized virtual characters and compose messages that can be sent as e-mail. The system maintains the database of all previously created characters and sent messages for a specific user identified with e-mail and password.

Creating new personalized virtual characters (Fig. 10) is based on Java Applet technology and provides the same functionality as described in previous section. Compared to mobile platform clients, Java Applets on desktop computers provide considerably more possibilities in terms of user interfaces. More precision is allowed in defining feature points since the adjustment of the mask is performed on larger displays. Also, the cost of data transfer, which is significantly cheaper compared to mobile devices, makes it possible to use higher resolution portrait pictures thus making better looking avatars.

It is possible to create new messages using any of the previously created characters. Animation, the LiveMail, is stored on the server and the link, dynamically generated URL, is sent to the recipient's e-mail address (Fig. 11). The player used to view animation is a Java Applet as well, based on the Shout3D rendering engine. It shows satisfying performance of 24–60 frames per second with average sized face models, on standard desktop computer.

Since the web client is completely Java based and requires no plug-ins, the LiveMail service is widely accessible – from any computer with Java Virtual Machine installed.

7. Conclusion

This chapter presents a prototype system that enables users to 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,

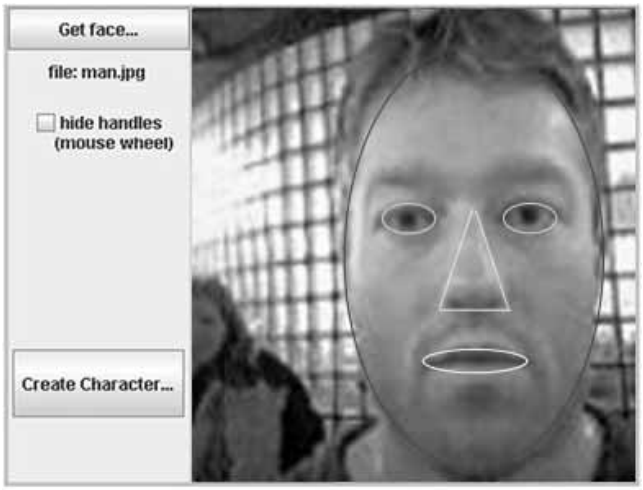

Fig. 10. Mask adjustment on Java applet-based Web client.

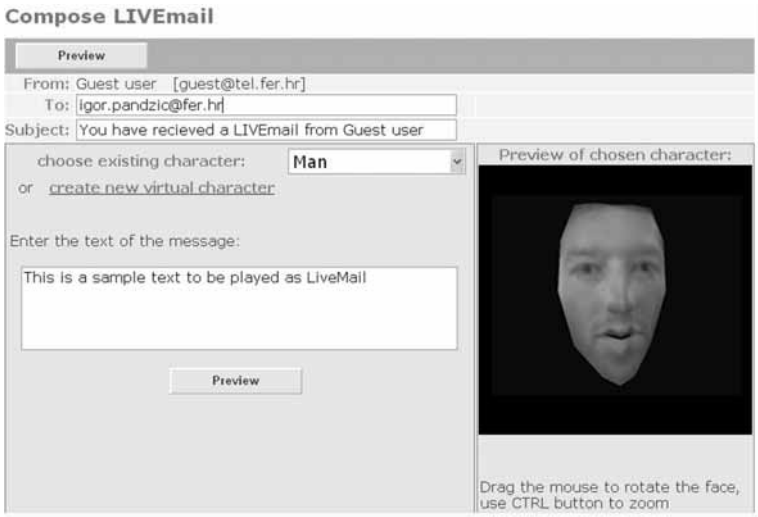

Fig. 11. Composing LiveMail message on Web client.

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 services like 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.

Introduced system can be used as a pure entertainment application but there are also other revenues from the system. Barriers were crossed with the face animation player on mobile platform and with 3D face model personalization on the server. Various different network technologies and face animation techniques were connected into one complex system and the experiences gained building such a system

were presented. This face animation system, based on MPEG-4 FBA standard, has low bandwidth requirement, and if combined with right facial feature tracking, technology presented in this chapter could be used for very low bandwidth visual communication.

Acknowledgments

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

References

[1] 3D Arts, DieselEngine SDK, http://www.3darts.fi/mobile/de.htm.
[2] F. Lavagetto and 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* **9**(2) (March 1999).
[3] F.I. Parke and K. Waters, *Computer Facial Animation*, A.K. Peters Ltd, 1996., ISBN 1-56881-014-8.
[4] Forum Nokia, *Enterprise: Developing End-to-End Systems*, Version 1.0, 2006, http://www.forum.nokia.com.
[5] H. Gupta, A. Roy-Chowdhury and R. Chellappa, *Contour based 3D Face Modeling From A Monocular Video*, British Machine Vision Conference, 2004.
[6] Igor S. Pandzic, *Facial Motion Cloning*, Graphical Models Journal.
[7] Igor S. Pandzic, Jörgen Ahlberg, Mariusz Wzorek, Piotr Rudol, Miran Mosmondor, *Faces Everywhere: Towards Ubiquitous Production and Delivery of Face Animation*, Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia, Norrkoping, Sweden, 2003.
[8] Igor S. Pandzic, Life on the Web, *Software Focus Journal*, **2**(2) (2001), 52–59, John Wiley & Sons.
[9] Igor S. Pandzic, Robert Forschheimer (editors), *MPEG-4 Facial Animation – The Standard, Implementations and Applications*, John Wiley & Sons, 2002, ISBN 0-470-84465-5.
[10] ISO/IEC 14496 – MPEG-4 International Standard, Moving Picture Experts Group, http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm.
[11] J. Ritter, Jack, An efficient bounding sphere, in: *Graphics Gems*, (Chapter V), A.S. Glassner, ed., Academic Press, San Diego, CA, 1990, pp. 301–303.
[12] M. Escher, I.S. Pandzic and N. Magnenat-Thalmann, *Facial Deformations for MPEG-4*, Proc. Computer Animation 98, Philadelphia, USA, IEEE Computer Society Press, 1998, 138–145.
[13] M. Mosmondor, H. Komericki, I.S. Pandzic and S. Igor, *3D Visualization of Data on Mobile Devices*, Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference – MELECON 2004, Dubrovnik, Croatia, 2004, 645–648.
[14] P. Hong, Z. Wen and T.S. Huang, Real-time speech driven Face Animation, in: *MPEG-4 Facial Animation – The Standard*, I.S. Pandzic and R. Forchheimer, eds, Implementation and Applications, John Wiley & Sons Ltd, 2002.
[15] C. Pelachaud, N. Badler and M. Steedman, Generating Facial Expressions for Speech, Cognitive, *Science* **20**(1) (1996), 1–46.
[16] T. Akenine-Möller and E. Haines, *Real-Time Rendering*, (2nd ed.), A K Peters, Natick, Massachusetts, 2002.
[17] T. Fuchs, J. Haber and H.-P. Seidel, *MIMIC – A Language for Specifying Facial Animations*, Proceedings of WSCG 2004, 2–6 Feb 2004, pp. 71–78.
[18] W. Lee and N. Magnenat-Thalmann, Fast Head Modeling for Animation, *Journal Image and Vision Computing* **18**(4), (March, 2000), 355–364, Elsevier.
[19] Z. Liu, Z. Zhang, C. Jacobs and M. Cohen, *Rapid Modeling of Animated Faces From Video*, In Proceedings of The Third International Conference on Visual Computing (Visual 2000), September 2000, Mexico City, 58–67.

Tomislav Kosutic is the System Integrator at the Enterprise Customized Solutions department of Ericsson Nikola Tesla, Zagreb, Croatia. He received his Dipl. Ing. degree in Electrical Engineering from University of Zagreb in 2004 and an Annual Rector's Reward of University of Zagreb for his diploma study: "Personalized Avatars in Mobile Environment". He is currently a postgraduate student at University of Zagreb. His research interests are in distributed multimedia systems, with focus on system architecture of personalized avatar services in mobile environments.

Miran Mosmondor received his Dipl. Ing. (2004) degree in electrical engineering from the University of Zagreb, Croatia. Since 2004 he has been employed as a research engineer in the Research and Development Center of the Ericsson Nikola Tesla

company in Croatia, working in the area of networked virtual reality. Currently he is also working towards his Ph.D. degree at the Faculty of Electrical Engineering and Computing, University of Zagreb. He published several conference and journal papers, some of which were awarded. His main research interests are in the field of multimedia communications, virtual environments and mobile applications development.

Mario Weber received his Dipl. Ing. degree in electrical engineering from the University of Zagreb, Croatia, in 2004. In the same year he received an annual Rector's reward for his diploma thesis. Currently he is postgraduate student at University of Zagreb. Since 2004 he has been employed as a software engineer in the KATE-KOM company. His main research interests are in the field of IP Multimedia Subsystem.

Ivan Andrisek received his Dipl. Ing. (2005) degree in computing science from the Faculty of Electrical Engineering and Computing, University of Zagreb. During studies he participated in projects in the area of virtual environments and mobile entertainment. Presently he is working as a software engineer at Ericsson Nikola Tesla, Zagreb. His current research interests are in the field of data warehouse systems and distributed environments.

Igor S. Pandzic is an Assistant Professor at the Department of Telecommunications, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. He teaches several undergraduate and postgraduate courses in the fields of virtual environments and multimedia communications. His main research interests are in the field of computer graphics and virtual environments, with particular focus on facial animation, embodied conversational agents, and their applications in networked and mobile environments. Igor also worked on networked collaborative virtual environments, computer generated film production and parallel computing. He published three books and around 60 papers on these topics. Formerly he worked as a Senior Assistant at MIRALab, University of Geneva, Switzerland, where he obtained his PhD in 1998. The same year he worked as visiting scientist at AT&T Labs, USA. In 2001–2002 Igor was a visiting scientist in the Image Coding Group at the University of Linkoping, Sweden, and in 2005 at the Department of Intelligence Science and Technology, Kyoto University, on a Fellowship awarded by Japan Society for Promotion of Science. Igor received his BSc degree in Electrical Engineering from the University of Zagreb in 1993, and MSc degrees from the Swiss Federal Institute of Technology (EPFL) and the University of Geneva in 1994 and 1995, respectively. Igor was one of the key contributors to the Facial Animation specification in the MPEG-4 International Standard for which he received an ISO Certificate of Appreciation in 2000. He held various functions in organizing numerous international conferences and served as guest editor for a special topic in IEEE Communications Magazine. Igor has been active in international research collaborations within the EU research programmes since the 3rd Framework Programme, as well as in national research projects and collaboration with industry.

Maja Matijasevic received her Dipl.-Ing. (1990), M.Sc. (1994), and Ph.D. (1998) degrees in Electrical Engineering from the University of Zagreb, Croatia, and the M.Sc. in Computer Engineering (1997) from the University of Louisiana at Lafayette, LA, USA. She is presently an Associate Professor at the University of Zagreb, Faculty of Electrical Engineering and Computing in Croatia. She was a Research Associate in the Virtual Reality and Multimedia Laboratory of the A-CIM Center at the University of Louisiana at Lafayette, USA, between 1996 and 1999, and a visiting researcher at the Technical University of Graz, Austria, on several occasions between 1992 and 1996. Her main research interests include networked virtual environments, quality of service, and advanced multimedia services in next generation networks. She is currently a team member of a Croatian Ministry of Science, Education and Sports funded national project no. 0036030, and she leads a research project in the area of networked virtual reality funded by Ericsson Nikola Tesla. She was also involved in joint projects with the Croatian Academic and Research Network CARNet. She has authored more than 40 refereed conference and journal publications. She has been involved in organization of conferences, serving as a Program, Registration and Publicity Chair. She was also a guest editor for the IEEE Communications Magazine feature topic on networked virtual environments. Dr. Matijasevic is a member of IEEE, ACM, and Upsilon Pi Epsilon Honor Society in the Computing Sciences. She is presently the Chair of the IEEE Communications Society Croatia Chapter.