A 3D talking head for mobile devices based on unofficial iOS WebGL support

Abstract

In this paper we present the implementation of a WebGL Talking Head for iOS mobile devices (Apple iPhone and iPad). It works on standard MPEG-4 Facial Animation Parameters (FAPs) and speaks with the Italian version of FESTIVAL TTS. It is totally based on true real human data. The 3D kinematics information are used to create lips articulatory model and to drive directly the talking face, generating human facial movements. In the last year we developed the WebGL version of the avatar. WebGL, which is 3D graphic for the web, is currently supported in the major web browsers for desktop computers. No official support is given for mobile device main platforms yet, although the Firefox beta version enables it on android phones. Starting from iOS 5 WebGL is enabled only for the advertisement library class (which is intended for placing ad-banners in applications). We were able to use this feature to visualize and animate our WebGL talking head.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—animation, color, shading, shadowing and texture, virtual reality;

Keywords: WebGL, iOS, mobile devices, talking head, facial animation, mpeg4

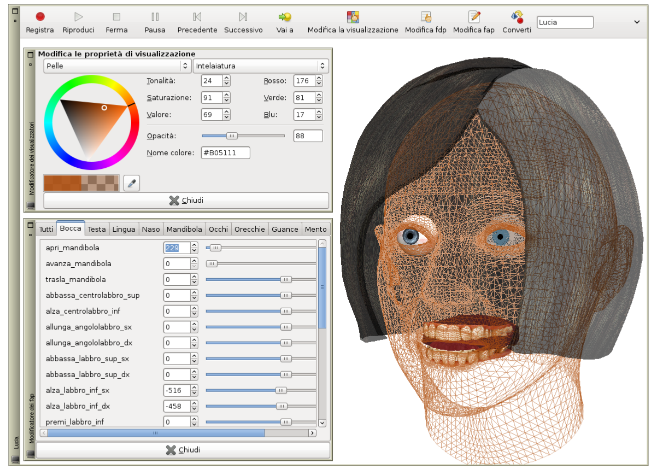

Figure 1: *Gtk-based GUI utility tools, showing the wireframe head model*

1 Introduction

Face to face communication is the main element of human-human interaction because both acoustic and visual signal simultaneously convey linguistic, extra linguistic and paralinguistic information. Therefore facial animation is a research topic since the early 70s and many different principles, models and animations have been proposed over years [Parke and Waters 1996; Ekman and Friesen 1978]. An efficient coding of shape and animation of human face was included in the MPEG-4 international standard [Pandzic and Forchheimer 2003]. We adopted this technology and we developed an open source facial animation framework which implements a decoder compatible with the Predictable Facial Animation Object Profile. The project was born about ten years ago as a standalone OpenGL application (fig. 1 shows some tools developed for the linux version). With the introduction of WebGL [Khronos Group 2012], which is 3D graphics for web browsers, we enhanced the possibility to embed the avatar in any internet site. Now its time for mobile devices. As far as we know this is the first webGL talking head native application running on iOS mobile devices.

2 System architecture

It follows the common client-server paradigm. First off the client (a web browser or a mobile device application) opens a connection with the server; the answer is an HTML5 web-page which delivers the multimedia contents to start the MPEG4 player. The overall system is depicted in fig. 2

2.1 The webgl client

The typical WebGL application is composed by three parts: the standard html code, the main JavaScript program and a new shading language section. The html section is intended mainly for user interaction; the JavaScript part is the core of the application: the graphic library itself, all the matrix manipulation, support and utility functions take place here; the input from the user is connected with JavaScript variables via ad-hoc event- driven procedures. The novelty is the third part which is the Shading Language code. This software runs on the Video Card. It is called GLSL and it derives from the C programming language. Actually these are the instructions that calculate every pixel color value on the screen whenever the drawing function is called in the JavaScript main program. To be able to change the values of the GLSL variables from the JavaScript WebGL Application Program Interface implements special methods to connect them with JavaScript objects, arrays and variables. During the initialization of the WebGL page the shader code is compiled and copied to the video card memory ready to be executed on the Graphic Processing Unit. At the beginning of the connection model parts data are fetched using the lightweight data-interchange format JSON [Network Working Group 2006]. This is the only moment where you could wait for a while because of the amount of the data to be transmitted, while right after this phase all the facial movements are almost real time.

The implementation of the face model uses a 3D mesh polygonal model. At the current stage of development, the avatar is a textured young female 3D face model built with 22237 polygons, divided into 7 independent components: the skin (14116 polygons), the hair (4616 polygons), the two eyes (1120x2 polygons), the tongue (236 polygons) and the teeth (1029 polygons). All these components derived from a static VRML [Bell et al. 1995; Schneider and Martin-Michiellot 1998] model with the exception of the eyes, whose models are dynamically generated at runtime, allowing to specify the desired level of detail. Some model details. are visible inf fig. 1.

The adopted animation model uses a pseudo-muscular approach, in which muscle contractions are obtained through the deformation of the polygonal mesh around feature points corresponding to the skin muscle attachments. The animation is built in real-time by modifying this structure and rendering it onto screen. The subdivision of the model in components allows to divide the skin, whose reticule of polygons is directly driven by the pseudo-muscles and constitutes a continuous and unitary element, from the inner articulators, such as the tongue and the teeth, and all the other anatomical com-

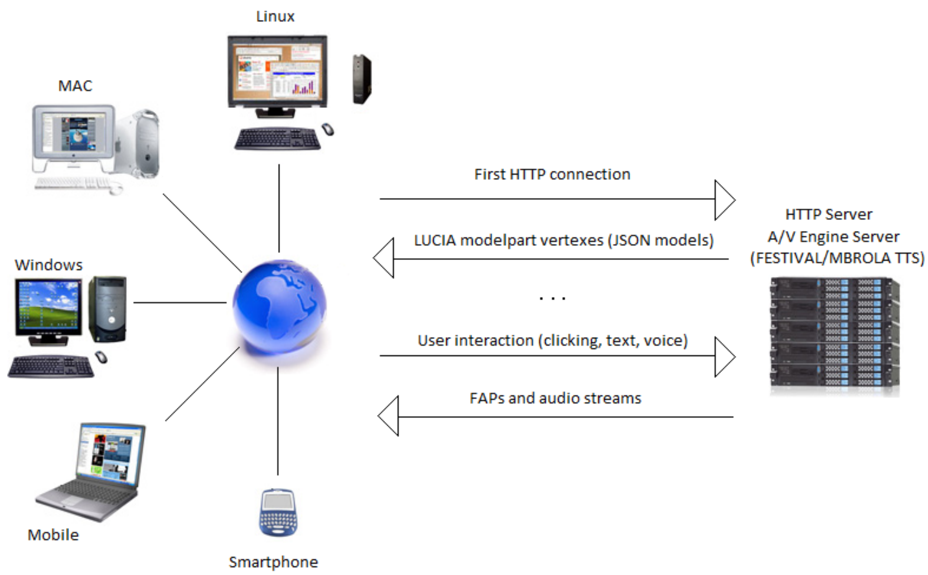

Figure 2: *The new client-server architecture: WebGL allows any system, even smart-phone and P.D.A, to interact with the avatar via standard web browsers. At the beginning of the connection the model-parts are fetched from the server in the JSON format. After that every communication involves only FAPs and audio streams with a very low bandwidth consumption*

ponents that move themselves independently, following translations and rotations (for example the eyes rotate around their centre). According to this strategy the polygons are distributed in such a way that the resulting visual effect is quite smooth with no rigid "jumps" over all the 3D model.

2.2 Technical details of unofficial use of WebGL

The realization of any iOS project is based on a concept (paradigm) called Model-View-Controller (MVC), which is a very logical way of dividing the code that makes up a GUI-based application. The MVC pattern divides all functionality into three categories:

- Model, which holds the application's data
- View, composed by all the elements that the user can see and interact with.
- Controller, which binds the Model and View together and is the application logic that decides how to handle the user's input.

The `UIView` class is the one that implements the View functionality. We can choose among different types of subclass depending on the type of application we want to develop.

`UIWebView` class is an `UIView` sub-class and it is used to embed web contents into iOS applications [Kochan 2003]. An `UIWebView` object can receive requests to load web contents (also moving back and forward in the navigation history with instance methods `goForward` and `goBack`), while remaining in the user's application. The main advantage of using `UIWebView` objects while developing applications is the cross-platform compatibility.

Although `UIWebView` class can easily manage all common browser functionalities, no WebGL rendering is offered by the public API. WebGL has been supported by Apple mobile devices since iOS version 5.0, but it's officially available only to iAd developers. The iAd framework indeed lets applications to display WebGL advertisements to the user. We found on the opensource Apple website an header file (`WebPreferencesPrivate.h`) with the preferences that might be public in the future. This file contains two methods concerning WebGL:

```
(BOOL)webGLEnabled;
(void)setWebGLEnabled:(BOOL)enabled;
```

Assuming that these symbols were linked against the object created by iAd framework, we guessed if it had worked with the standard `UIWebView` class instances. Assuming also that Apple reserved variables starting with underscore char for its own uses, we successfully detected `_setWebGLEnable` private method and we started writing our custom method.

```
(void)setWebGLEnabled:(BOOL)activateWebGL {
 UIWebDocumentView* myWebDocumentView=[self _browserView];
 WebView* webGLView=[myWebDocumentView webView];
 [webGLView _setWebGLEnabled:activateWebGL];
}
```

Using this custom method to allocate the `WebView` object (`webGLView`) the private method on the last raw enables WebGL rendering and it is suddenly available for any 3D graphics visualization.

2.3 The AudioVideo engine server

Audio Video speech synthesis, that is the automatic generation of voice and facial animation parameters from arbitrary text, is based on parametric descriptions of both the acoustic and visual speech modalities. The acoustic speech synthesis uses an Italian version of the FESTIVAL di-phone TTS synthesizer [Cosi et al. 2001] modified with emotive/expressive capabilities: the APML/VSML mark up language [Carolis et al. 2004] for behavior specification permits to specify how to markup the verbal part of a dialog in order to modify the graphical and the speech parameters that an animated agent need to produce the required expressions. For the visual speech synthesis a data-driven procedure was utilized: visual data are physically extracted by an automatic opto-tracking movement analyzer for 3D kinematics data acquisition called ELITE [Ferrigno and Pedotti 1985]. The 3D data coordinates of some reflecting

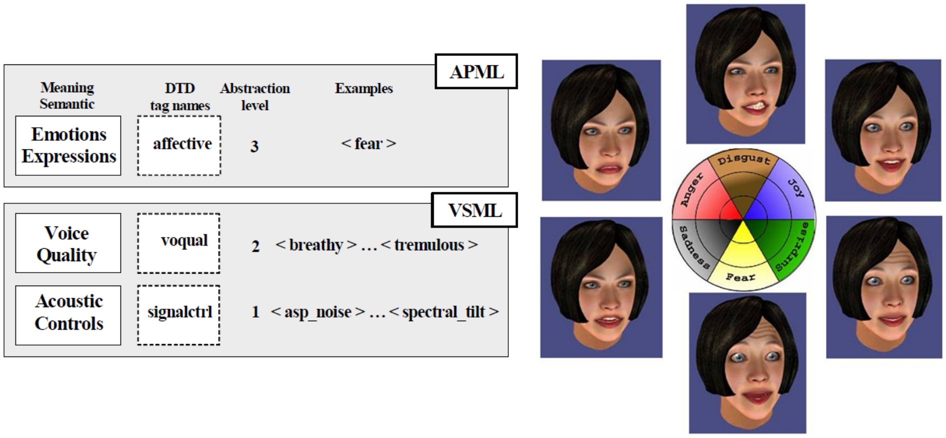

Figure 3: *On the lesft the APML/VSML mark-up language extensions for emotive audio/visual synthesis. On the right the Emotional Parrot Demo: clicking one of the six emotional states on the disc, the correspondent configuration is activated*

markers positioned on the actor face are recorded and collected, together with their velocity and acceleration, simultaneously with the co-produced speech. Using PRAAT [Boersma 1996], we obtain parameters that are quite significant in characterizing emotive/expressive speech [Drioli et al. 2003]. In order to simplify and automates many of the operation needed for building-up the 3D avatar from the motion-captured data we developed INTERFACE [Tisato et al. 2005], an integrated software designed and implemented in Matlab. To reproduce realistic facial animation in presence of co-articulation, we adopted a modified version of the Cohen-Massaro co-articulation model [Cosi and Perin 2002].

3 Emotional parrot demo

An extended version of the APML language has been included in the FESTIVAL speech synthesis environment, allowing the automatic generation of the extended phonation file from an APML tagged text with emotive tags. This module implements a three-level hierarchy in which the affective high level attributes (e.g. <anger>, <joy>, <fear>) are described in terms of medium-level voice quality attributes defining the phonation type (e.g., <modal>, <soft>, <pressed>, <breathy>, <whispery>, <creaky>). These medium-level attributes are in turn described by a set of low-level acoustic attributes defining the perceptual correlates of the sound (e.g. <spectral tilt>, <shimmer>, <jitter>). The low-level acoustic attributes correspond to the acoustic controls that the extended MBROLA synthesizer can render through the sound processing procedure described above. This descriptive scheme (left side of fig. 3) has been implemented within FESTIVAL as a set of mappings between high-level and low-level descriptors. The implementation includes the use of envelope generators to produce time curves of each parameter.

To show the capabilities of emotional synthesis of the avatar you can play with a demo called "Emotional Parrot": the avatar repeats any input text you enter in six different emotional ways: joy, surprise, fear, anger, sadness, disgust. For this classification we take inspiration from [Ruttkay et al. 2003]. The demo is perfectly fluid on desktop computer (about 60 fps) while it suffers of some frames skipping on low computational machine (7 fps on iPad 2)

4 Conclusion and future work

In this work we presented an MPEG-4 standard FAPs driven facial animation Italian talking head. It is a decoder compatible with the "Predictable Facial Animation Object Profile". It has a high quality 3D model and a fine co-articulation model, which is automatically trained by real data, used to animate the face. It runs on any WebGL compatible browser and now, with our succesful hacking, also on Apple iOS mobile devices (fig. 4). It reproduces six different emotional states of the input text in emotional parrot mode.

In the near future we will integrate speech recognition to have double input channels and we will test the performance of the Mary TTS synthesis engine [Schrder and Trouvain 2003] for the Italian language. We will work on dynamic mesh reduction t[illegible]ance the frame rate on slow computational hardware.

Acknowledgements

This work has been sponsored by WIKIMEMO.IT (The Portal of Italian Language and Culture, FIRB Project, RBNE078K93, Italian Ministry of University and Scientific Research)

References

BELL, G., PARISI, A., AND PESCE, M., 1995. The virtual reality modeling language.

BOERSMA, P. 1996. Praat, a system for doing phonetics by computer. *Glot International*, 341–345.

CAROLIS, B. D., PELACHAUD, C., POGGI, I., AND STEEDMAN, M. 2004. Apml, a mark-up language for believable behavior generation. *Life-Like Characters*, 65–85.

COSI, P., AND PERIN, G. 2002. Labial coarticulation modeling for realistic facial animation. In *Proceedings of ICMI 2002*, ICMI, 505–510.

COSI, P., TESSER, F., GRETTER, R., AND AVESANI, C. 2001. Festival speaks italian! In *Proceedings of Eurospeech 2001*, Eurospeech, 509–512.

DRIOLI, C., COSI, P., TESSER, F., AND TISATO, G. 2003. Emotions and voice quality: Experiments with sinusoidal modeling. In *Proceedings of Voqual 2003*, ISCA, 127–132.

EKMAN, P., AND FRIESEN, W. 1978. Facial action coding system. *Consulting Psychologist*.

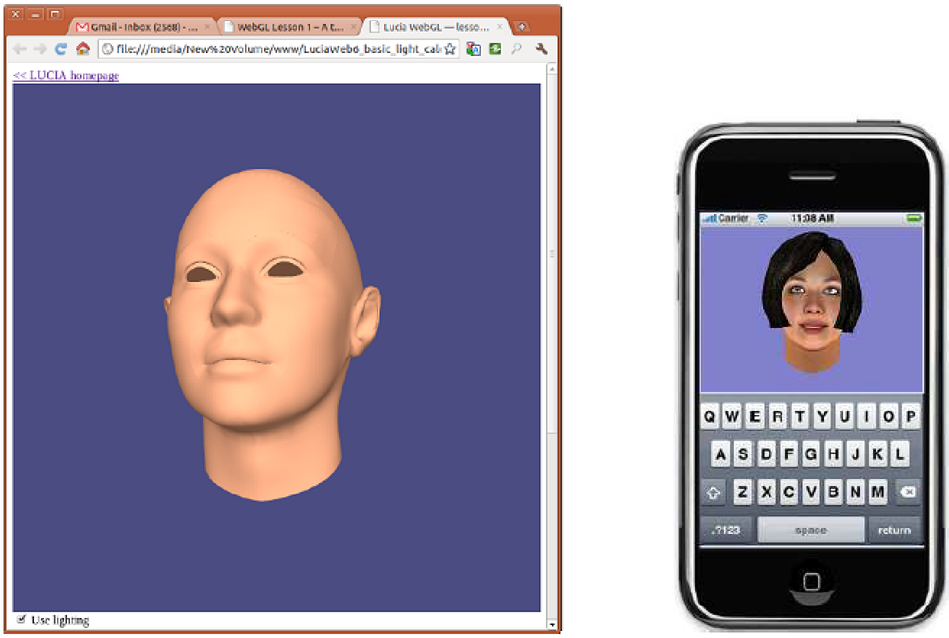

Figure 4: *On the left the skin element rendered in Chrome web-browser. Gouraud shading is adopted to obtain a very smooth and realistic curve face. On the right the Emotional Parrot Mode running on Apple iPhone*

FERRIGNO, G., AND PEDOTTI, A. 1985. Elite: A digital dedicated hardware system for movement analysis via real-time tv signal processing. *IEEE Transactions on Biomedical Engineering*, 943–950.

KHRONOS GROUP, 2012. Webgl - opengl es 2.0 for the web.

KOCHAN, S. 2003. *Programming in Objective-C*. Sams.

NETWORK WORKING GROUP, 2006. Javascript object notation.

PANDZIC, I. S., AND FORCHHEIMER, R., Eds. 2003. *MPEG-4 Facial Animation: The Standard, Implementation and Applications*. John Wiley & Sons, Inc., New York, NY, USA.

PARKE, F. I., AND WATERS, K. 1996. *Computer facial animation*. A. K. Peters, Ltd., Natick, MA, USA.

RUTTKAY, Z., NOOT, H., AND HAGEN, P. 2003. Emotion disc and emotion squares: tools to explore the facial expression space. In *Computer Graphics Forum*, 49–53.

SCHNEIDER, D., AND MARTIN-MICHIELLOT, S., 1998. Vrml primer and tutorial.

SCHRDER, M., AND TROUVAIN, J. 2003. The german text-to-speech synthesis system mary: A tool for research, development and teaching. In *Internation J. of Speech Technology*.

TISATO, G., DRIOLI, C., COSI, P., AND TESSER, F. 2005. Interface: a new tool for building emotive/expressive talking heads. In *Proceedings of INTERSPEECH 2005*, INTERSPEECH, 781–784.