



FP6-507609

SIMILAR

The European research taskforce creating
human-machine interfaces SIMILAR
to human-human communication

Network of excellence
FP6 - IST

Deliverable #93 **eNTERFACE 2006 Publications of proceedings**

Due date of deliverable: September, 2006
Actual submission date: January 15, 2007

Start date of project: 1st December 2003

Duration: 48 months

Organisation name of lead contractor for this deliverable: FPMs

Thierry Dutoit
FER - Zagreb University
Igor S. Pandzic

Revision 1

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	



eNTERFACE'06

The SIMILAR NoE Summer Workshop
on Multimodal Interfaces

PROCEEDINGS eNTERFACE'06

Summer Workshop
on Multimodal Interfaces

July 17- August 11, 2006
Centre for Advanced Academic Studies, Dubrovnik, Croatia

Prof. Igor S. Pandžić, Chair

Faculty of Electrical Engineering and Computing
Department of
Telecommunications

Unska 3, HR-10000 Zagreb

Ph.(+385) 1 6129 810 - Fax(+385) 1 6129832

igor.pandzic@tel.fer.hr

<http://www.tel.fer.hr/users/ipandzic/>





eNTERFACE Workshops

Actively building the European Research Area

What are eNTERFACE workshops?

The [eNTERFACE summer workshops](#), organized by the SIMILAR, European Network of Excellence, are a new type of European workshops. They aim at establishing a tradition of collaborative, localized research and development work by gathering, in a single place, a group of senior project leaders, researchers, and (undergraduate) students, working together on a pre-specified list of challenges, for 4 weeks. Participants are organized in teams, attached to specific projects related to multimodal interfaces, working on free software.

First eNTERFACE'05 was held at Faculté Polytechnique de Mons, Belgium, in July-August 2005. Next one followed in Dubrovnik, Croatia, and was organized by Faculty of Electrical Engineering and Computing from Zagreb. eNTERFACE'07 workshop will be organized in Istanbul, Turkey, also in July-August 2007 as the previous ones.

What do eNTERFACE workshops produce as results?

At the end of the workshop, a public presentation day is organized, in which the team leaders explain and demonstrate the results of their project. A press conference is also organized, to maximally publicize the event. All results, codes and data, are then made publicly available with an MIT-like open source license.

Last but not least, the workshop proceedings are produced 6 weeks after the end of the workshop, in which each team contributes a 15pp. paper on the project they had to study, the related state-of-the-art, the problems encountered and the solutions proposed and implemented.

But still more importantly, eNTERFACE workshops create a real transfer of know-how among participants, who continue to work together after the workshop has closed. They actively contribute to building the European Research Area, by establishing a tradition of localized collaborative research.



One working day at the eNTERFACE '06

The eINTERFACE funding model

No funding is provided by the organizers for researchers, but no registration fees are asked for either. Participants therefore have to pay for their travel, lodging, and catering expenses, using their SIMILAR finances or other EU, national, or regional funding. Catering and lodging is available from the University organizing the workshop, at minimal student rates. Some grants are available from scientific societies.

A limited number of undergraduate students (typ. 10) are also selected (based on their CVs and recommendations from professors), whose travel and accommodation expenses are funded by the organizers.



The Summer Workshop on Multimodal Interfaces July 17th - August 11th, Dubrovnik, Croatia

eINTERFACE '06, the second in the series of eINTERFACE workshops, was hosted by the Faculty of Electrical Engineering and Computing, University of Zagreb and held at the Centre for Advanced Academic Studies in Dubrovnik, Croatia, from July 17th to August 11th, 2006. Following the general organization process of eINTERFACE workshops, nine projects were first selected on the basis of an international call for projects. From this list of projects, a call for participation was then launched internationally, and participants were selected on the basis of their CVs and potential input in the projects. This call resulted in the selection of 63 researchers from 12 countries all around the world (but mostly Europe), organized in 9 teams.

Country	Researchers number	Country	Researchers number
Belgium	11	Japan	4
Canada	1	Spain	5
Croatia	10	Switzerland	2
Czech Republic	1	Turkey	7
Estonia	1	United Kingdom	1
France	11	Vietnam	1
Greece	8		

Each team worked for a complete month on one of the following 9 challenges:

1. **An Agent Based Multicultural User Interface in a Customer Service Application**

Coordinators: Hung-Hsuan Huang, *Prof. Toyoaki Nishida, Kyoto University, Japan*; Prof. Igor Pandzic, *Faculty of Electrical Engineering and Computing, Croatia*; Prof. Yukiko Nakano, *Tokyo University of Agriculture and Technology, Japan*

This project aimed to explore the possibility of rapidly building multicultural ECA interfaces for customer service applications with a common framework connecting their functional blocks.

2. **Multimodal tools and interfaces for the intercommunication between visually impaired and "deaf and mute" people**

Coordinator: Prof. Dimitrios Tzovaras, *Informatics and Telematics Institute, Greece*

The goal of this project was the development of a real time sign language tutoring tool related to a limited number of well defined gestures which associate hand gestures and head motion and facial expressions. This exhibited the feasibility of such a system which requires the fusion of three sources of information.

3. Sign Language Tutoring Tool

Coordinators: Prof. Lale Akarun, *Bogazici University, Turkey*; Profs. Alice Caplier & Michele Rombaut, *Universite de Grenoble, France*

The goal of this project was the development of a real time sign language tutoring tool related to a limited number of well defined gestures which associate hand gestures and head motion and facial expressions.

4. Multimodal Character Morphing

Coordinators: Prof. Yannis Stylianou, *University of Crete, Greece*; Thierry Dutoit, *Faculte Polytechnique de Mons (FPMs), Belgium*; Profs. Antonio Bonafante & Ferran Marques, *Universitat Politècnica de Catalunya (UPC), Spain*

This project aimed at performing high quality transformation on the multimodal recordings (audiovisual files) of a source speaker A. Using both voice conversion and video morphing, the result was the a set of audiovisual files with a target speaker B speaking with his/her own voice and acting like A does.

5. Introducing Network-Awareness for Networked Multimedia and Multi-modal Applications

Coordinators: Prof. Maja Matijasevic, *Faculty of Electrical Engineering and Computing, Croatia*; Miran Mosmondor, *Ericsson Nikola Tesla, Croatia*

The project objective was to create an application programming interface (API) which will enable multimodal application developers to create networked services for heterogeneous end-user devices, capable of requesting and adapting to network quality of service (QoS), but without the need to know the signalling protocol specifics.

6. An instrument of sound and visual creation driven by biological signals

Coordinators: Prof. B. Macq, Rémy Lehembre & Jean-Julien Filatriau, *TELE Lab, UCL Louvain-La- Neuve, Belgium*

Pursuing this first eINTERFACE workshop, this project aimed to use biophysical signals (EEG, EMG, ECG, EOG, etc...) analysis to drive digital musical instruments, enhanced with a rich visual feedback, and playable in real-time. This year improvement the interaction musician-instrument by expanding the mapping between biological signals and synthesis parameters was done.



7. Emotion Detection in the Loop from Brain Signals and Facial Images

Coordinators: Profs. Bulent Sankur & Lale Akarun, *Bogazici University, Turkey*; Profs. Alice Caplier & Michele Rombaut, *Universite de Grenoble, France*

This project aimed to develop techniques for multimodal emotion detection, one modality being brain signals via fNIRS, the other modality being face video and the third modality being the scalp EEG signal.

8. Realtime and Accurate Control of Expression in Singing Synthesis

Coordinator: D'Alessandro Nicolas, *Faculte Polytechnique de Mons (FPMs), Belgium*

The main purpose of this project was to develop a full computer-based system musical instrument allowing real-time synthesis of expressive singing voice. The expression results from the continuous action of an interpreter through a gesture controlled interface. Those gesture parameters influence the voice characteristics thanks to a particular mapping strategy.

9. Multimodal Driving Simulator

Coordinators: Prof. Alice Caplier & Laurent Bonnaud, *Universite de Grenoble, France*; Prof. Laurence Nigay, *Université Joseph Fourier, Grenoble, France*; Prof. Dimitrios Tzovaras, *Informatics and Telematics Institute, Greece*

Facing the sophisticated sensing and interaction technology available in modern cars, this project aimed at designing and developing a multimodal driving simulator that is based on both multimodal driver's focus of attention detection and driver's state detection (i.e., stress and fatigue) as well as multimodal interaction for enhancing a driving task.

The workshop finished on August 11th with the full day of project result presentations and demonstrations. Both the opening and the final day were covered by Croatian national television, so eINTERFACE featured on the national TV on three different occasions.

All project results, TV recordings, press releases, photo gallery and other materials are available from the workshop web site: <http://www.enterface.net/enterface06>



The eINTERFACE'06 Sponsors

We want to express our gratitude to all the organizations which made this event possible.



The eINTERFACE'06 Scientific Committee

Niels Ole Bernsen, *University of Southern Denmark - Odense, Denmark*
 Thierry Dutoit, *Faculté Polytechnique de Mons, Belgium*
 Christine Guillemot, *IRISA, Rennes, France*
 Richard Kitney, *University College of London, United Kingdom*
 Benoît Macq, *Université Catholique de Louvain, Louvain-la-Neuve, Belgium*
 Cornelius Malerczyk, *Zentrum für Graphische Datenverarbeitung e.V, Germany*
 Ferran Marques, *Univertat Politècnica de Catalunya PC, Spain*
 Laurence Nigay, *Université Joseph Fourier, Grenoble, France*
 Dimitrios Tzovaras, *Informatics and Telematics Intsitute, Greece*
 Jean-Philippe Thiran, *Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland*
 Jean Vanderdonckt, *Université Catholique de Louvain, Louvain-la-Neuve, Belgium*

The eINTERFACE'06 Local Organization

Coordination: Igor. S Pandzic
 Accomodation & Food: Goranka Zoric
 Social Activities : Aleksandra Cerekovic
 Web Management: Vjekoslav Levacic
 Technical Support : Ognjen Dobrijevic

eNTERFACE 2006

PROJECT REPORTS

Project 1:	An Agent Based Multicultural User Interface in a Customer Service Application	1
Project 2:	Multimodal tools and interfaces for the intercommunication between visually impaired and "deaf and mute" people	11
Project 3:	Sign Language Tutoring Tool	23
Project 4:	Multimodal Character Morphing	34
Project 5:	Introducing Network-Awareness for Networked Multimedia and Multi-modal Applications	46
Project 6:	An instrument of sound and visual creation driven by biological signals	59
Project 7:	Emotion Detection in the Loop from Brain Signals and Facial Images	69
Project 8:	Realtime and Accurate Control of Expression in Singing Synthesis	81
Project 9:	Multimodal Driving Simulator	91

An Agent Based Multicultural User Interface in a Customer Service Application

Hung-Hsuan Huang¹, Aleksandra Cerekovic², Kateryna Tarasenko¹, Vjekoslav Levacic², Goranka Zoric², Margus Treumuth⁴, Igor S. Pandzic², Yukiko Nakano³, and Toyoaki Nishida¹

¹Graduate School of Informatics, Kyoto University, Japan, ²Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, ³Department of Computer, Information and Communication Sciences, Tokyo University of Agriculture & Technology, Japan, ⁴Institute of Computer Science, University of Tartu, Estonia

Abstract—The advancement of traffic and computer networks makes the world more and more internationalized and increases the frequency of communications between people using different languages and expressing different nonverbal behaviors. To improve communication of embodied conversational agent (ECA) systems with their human users, the importance of their capability to cover the cultural differences emerged. Various excellent ECA systems are developed and proposed previously, however, the cross-culture communication issues are seldom addressed by researchers. This project aims to explore the possibility of rapidly building multicultural and multimodal ECA interfaces for customer service applications with a generic framework connecting their functional blocks.

Index Terms— embodied conversational agent, distributed system, blackboard, user interface, non-verbal interaction

I. PROJECT BACKGROUND

EMBODIED conversational agents (ECA) are computer generated humanlike characters that interact with human users in face-to-face conversation and possess the following abilities:

- Recognize and respond to verbal and nonverbal input
- Generate verbal and nonverbal output
- Perform conversational functions (e.g. utterance turn taking, feedback and repair mechanisms)
- Give signals that indicate the state of conversations as well as to contribute new propositions

To achieve these features, system assemblies such as natural language processing, sensor signal processing, verbal and nonverbal behavior understanding, facial expression recognition, dialogue management, personality modeling, emotional modeling, natural language generation, facial expression generation, gesture generation, and CG animator are required. These functions actually involve multiple disciplines like A.I., computer graphics, cognitive science, sociology, linguistics, psychology, etc. They are in so broad range of research disciplines such that virtually no single research group can cover all

aspects of a fully operating ECA system. Moreover, the software developed from individual research result is usually not meant to cooperate with each other and is designed for different purpose. Hence, if there is a common and generic backbone framework that connects a set of reusable modularized ECA software components, the rapid building of ECA systems will become possible and the redundant efforts and resource uses of ECA researches can be prevented. For these reasons, our group is developing such a generic ECA platform and researching the adequate communicative interfaces between ECA software blocks. As a result, a basic system model is developed with a prototype system and described in the next section.

On the other hand, the advancement of traffic and computer networks makes the world more and more internationalized and increases the frequency of communications between people using different languages and expressing different nonverbal behaviors. To improve the communication of ECA systems with their human users, the importance of their capability to cover the cultural differences emerged. Although various excellent agent interface systems are developed and proposed previously, the cross-culture communication issues are seldom addressed by researchers.

II. PROJECT OBJECTIVES

To explore the issues that may occur in multicultural communication, especially nonverbal communicative behaviors performed spontaneously by humans; we propose this project with the objective to develop a customer service application with an ECA interface which serves human users from different cultures based on the generic ECA framework. Based on the discussion among the team members prior to the workshop, the target application is decided to be a tour guide agent of Dubrovnik city where is specified as a UNESCO Worlds Heritage. Since most of the team members come from Japan and Croatia, it is most convenient to gather first-hand Japanese and Croatian cultural information where the differences are supposed to be fairly obvious. A guide agent dynamically changes its behaviors either in Japanese way or in Croatian way according to its

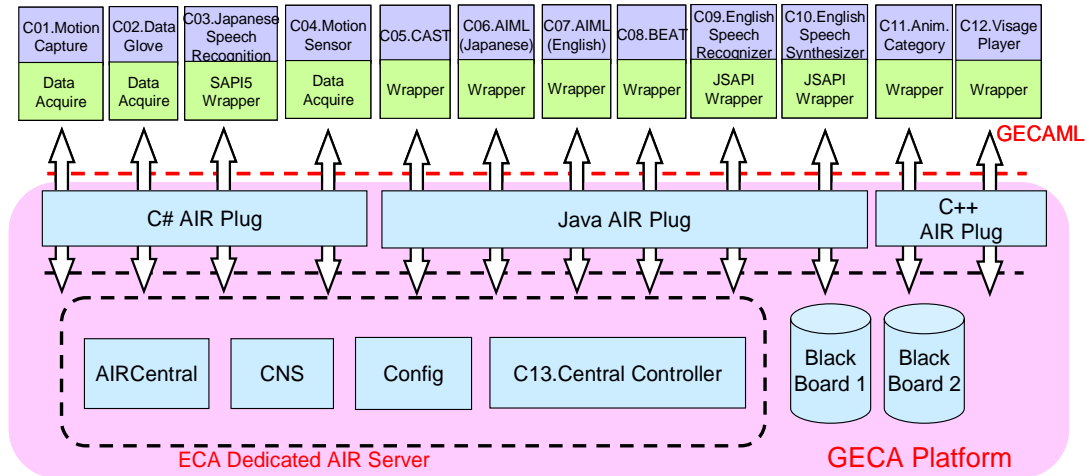


Fig. 1 The conceptual diagram of GECA framework and the configuration of the eNTERFACE06 agent

visitor was thus suggested. On the other hand, because of the lack of good-quality speech synthesizer/recognizer for Croatian, the guide agent will speak and listen to English in its Croatian mode.

In this system, the agent mediates the site seeing information of Dubrovnik to its visitors via verbal and non-verbal interactions. An example scenario is: when a visitor comes to the system, the system recognizes the visitor as a Japanese or Croatian from the combination of the speech recognizer's result and the non-verbal behaviors of the visitor such as bowing in greeting in Japanese culture. The agent then switches to its Japanese mode, that is, speaks Japanese and behaves like a Japanese to accept and answer the queries from the visitor while performing culture-dependent gestures according to predefined scenarios in that session. At the same time, the visitor can interact with the agent not only by natural language speaking but also by non-verbal behaviors such as pointing to an object on the background image or raising his (her) hand to indicate that he (she) wants to ask a question. Besides, to reduce the system complexity and prevent the drawbacks come from an ill-implemented 3D environment, in the prototype system that is going to be implemented during eNTERFACE'06, scene transitions are approximated by camerawork and the changes of realistic background photos instead of building a full 3D virtual world.

III. GENERIC ECA FRAMEWORK

To connect many heterogeneous functional components to an integral virtual human, the consistency of all communication channels and the timing synchronization of all components will be very important issues. Also, to handle nonverbal inputs from humans, the capability to handle streaming data from sensors in real-time is indispensable. Our platform is built upon a routing and communication protocol of cooperating A.I. programs, OpenAIR [1]. The platform mediates the information exchange of ECA software components with XML messages via shared memory mechanism (blackboard or white boards in OpenAIR's context) and will have the following advantages:

- Distributed computing model over network eases the integration of legacy systems
- Communication via XML messages eliminates the dependency on operating systems and programming languages
- Simple protocol using light weight messages reduces the computing and network traffic overhead
- Prioritized messages make quality of service control possible and facilitates real-time event processing (not implemented yet)
- Explicit timing management mechanism (partially implemented)
- Support discrete messages and streaming sensor data at the same time (partially implemented)
- The use of shared backbone blackboards flatten the component hierarchy, shorten the decision making path and can realize reflexive behaviors
- Possible to use multiple logically isolated blackboards rather than traditional single blackboard (not implemented yet)
- Components can communicate with each other directly or via blackboard(s) (not implemented yet)
- Easy to switch or replace components which have the same function if they understand and generate messages in the same type

Figure 1 shows the conceptual diagram of the GECA framework and the configuration of the planned Dubrovnik tour guide agent. Based on this framework, we are specifying an XML based high-level protocol for the data exchanges between the components plugged into the GECA platform. Every GECA message belongs to a message type, for example, "input.speech.text", "output.action.speak", etc. Each message type has a specified set of XML elements and attributes, for example, "intensity", "duration", "start_time", etc. The message flow works like the following scenario upon the platform, when a component starts; it registers its contact information (unique name, IP address, etc) to CNS (Central Naming Service) component and subscribes its interested message type(s) to the AIRCentral component. Then the mes-

sages in those types will be sent to the component from the specified blackboard (or a *whiteboard* in OpenAIR's terminology) which behaves like a shared memory between the components when some other component *published* the messages. That component then processes the data it got and publishes its own output to the shared blackboard in certain message type.

By utilizing the communicative functions provided by the Air Plug libraries (currently we have developed the C#, C++ version libraries and a customized Java reference implementation from mindmakers.org) which are a part of the platform, an ECA system builder needs to develop a small piece program called a *wrapper* in order to handle and convert the input/output of an existing software component to be GECAML (Generic ECA Markup Language) compliant. After doing this, the heterogeneous nature of components that provide the same capability (for example, both of a MS SAPI4 TTS and a JSAPI TTS provide the same capability of the agent, i.e. to speak out from text) can be hidden and behave identically to the other software components.

IV. DUBROVNIK TOUR GUIDE AGENT

During the eNTERFACE06 project's four-week period, the participants of this project cooperated to develop the tour guide agent described in section II. In this section, we discuss the GECA software components that are used in this project and the main tasks that were dealt during the project period.

A. Software Component Configuration

This agent was planned with the component configuration depicted in Figure 1. The follows are the brief descriptions of those software components.

- C01. Motion capture component. This component utilizes a simple motion capture device [2] using IR technology to roughly approximate a predefined set of human visitor's non-verbal behaviors.
- C02. Data glove component. This component acquires data from a data glove hardware device and reports recognized movements of the visitor's fingers to the other components.
- C03. Japanese speech recognition component. This component is a wrapped SAPI-5 Japanese recognition engine, Julius [3] and has been implemented.
- C04. Motion sensor component. This component acquires data from a 3 dimensional acceleration sensor [4] which is attached on the visitor's head to detect head shaking and nod movements. This component has been implemented.
- C05. Japanese spontaneous gesture generating component. This component is a wrapper of CAST [5] engine which generates the type and timing information of spontaneous gestures from Japanese utterance input string. This component has been implemented.
- C06. AIML interpreter components for Japanese. This component wraps a Java implementation [6] of AIML [7] interpreter. It reads one or more AIML scripts which specify the agent's verbal and nonverbal responses to certain input behaviors from the visitors. Therefore, this

component behaves like the brain of the agent and thus the current agent shows only reflexive behaviors with some context referencing capability comes with AIML and has no internal state. Besides, because the original AIML does not accept customized tags, a set of tags specifying visitor's non-verbal inputs and agent's non-verbal outputs must be encoded into the script. The wrapper of this component has been implemented but the scenario script(s) has to be defined during the eNTERFACE workshop.

- C07. AIML interpreter components for English. The same as above except this component handles English inputs / outputs.
- C08. English spontaneous gesture generating component. This component is a wrapper of BEAT [8] which generates the type and timing information of spontaneous gestures from English utterance input string. This component has not been implemented yet.
- C09. English speech recognition component. This component wraps a speech recognition engine to recognize English speaking of the visitor and from predefined grammar rule and sends the recognized result as a text string to the subscribed components. This component has not been completed yet.
- C10. English Text-To-Speech component. This component wraps an English Text-To-Speech (TTS) engine to generate the voice output of the agent and viseme events to drive the character animator to move the agent's lips. This component has not been completed yet.
- C11. Animation category component. This component is a database storing the number values of MPEG4 FBA parameters of a predefined set of animation / action to drive the character animation in real-time. This component has not been implemented yet.
- C12. Character animation player component. This component is a wrapped character animation player which is implemented in visage|SDK [9]. It accepts driving event messages from the animation category and speech synthesizer component and performs the specified character animation.
- C13. Central controlling component dedicated to ECA. This component is one part of the OpenAIR server and handles synchronization among the components, ensures integrity of all output modals, selects the actions to perform if there is some contradiction. A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

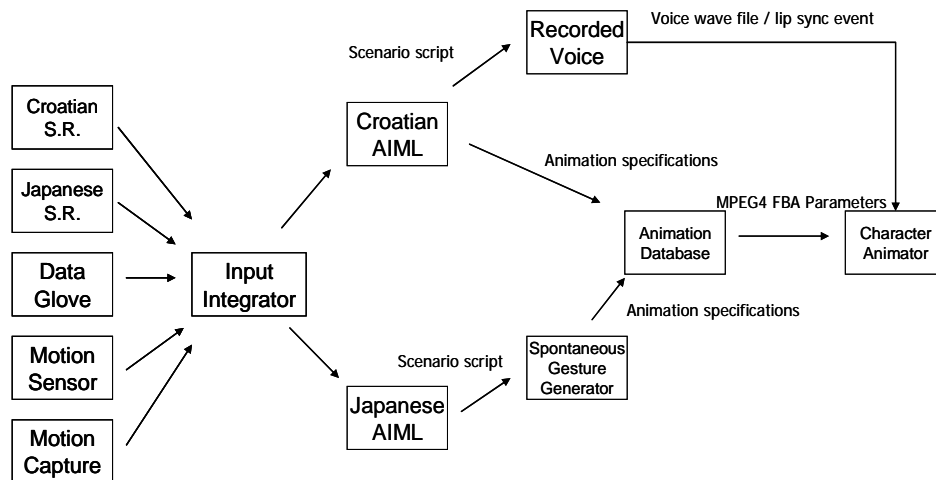


Fig. 2 The data flow of the Dubrovnik tour guide agent

However, during the development period, there were some modifications made to the original plan. Because the venue of the workshop is in Croatia, we decided to use pre-recorded Croatian voice instead of English TTS and use English speech recognition engine to recognize a limited range of Croatian vocabularies. Figure 2 shows the data flow among the components of the actually built Dubrovnik tour guide agent. The speech recognition component and sensor components gather and recognize the verbal and nonverbal inputs from the human user and send the results to the AIML component. The inputs from different modals are combined by the wrapper of AIML component and are matched with predefined scenario AIML scripts. The AIML then sends the matched response which may include utterance and action tags to speech synthesizer and animation category component. Depends on the design of the spontaneous gesture generating component, the speech synthesizer component may output the generated voice by itself and send the visime events to the animator to drive the agent's lips or leave these jobs to the other components. In either way, timing information is sent to the spontaneous gesture generator. The spontaneous gesture generator inserts action tags into the utterance according to the timing information from the speech synthesizer and its natural language tagging companion. The animation category listens to action queries from the spontaneous gesture generating component or the AIML component and sends FBA (Facial Body Animation) parameters to drive the character animator. The character animator listens to action and visime events and play them in real-time. Some character animators (e.g. visage) may also provide TTS support; in that case, it also listens to the utterance output of the spontaneous gesture generating component. Furthermore, shortcuts between the sensor components and the animator that bypass the pipeline are allowed and make reflexive behaviors of the agent possible, and this is one of the strengths of this framework over the other ECA architectures.

B. Non-verbal input recognition

To provide an immersive environment for the user to interact with the tour guide agent, a LCD projector with tilt-compensation function is used to project a large enough image of the agent on the screen. The user then stands in front of the screen and interact with the guide agent as (s)he is really in the virtual Dubrovnik space.

In the non-verbal input recognition issue, the aim is to detect the following behaviors from the user:

- Get the agent's attention
- Point to the interested objects shown on the display
- Show the willing to ask a question
- Interrupt the agent's utterance
- Shake head and nod to express positive and negative answers

Because of the nature of the eINTERFACE workshop, only small size and portable sensor devices are adopted in this project. These non-verbal behaviors are recognized by using the data from data gloves, infrared camera, and acceleration sensors.

Nissho Electronics Super Glove

This data glove is a simple wearable input device which user can put on his right hand to detect finger curvature. Ten sensors, two for each finger, are used to detect how fingers are bent. Prior to first use, user must calibrate the glove's sensor readings by putting the fingers into three different positions. Data glove is connected with a cable to the control box which is a power input device and a processing unit of the data collected from the sensors. Control box can be connected to the PC with a serial cable and a serial port reader can be used to read the glove data. Data from the glove is represented with thirty ASCII characters. Three ASCII characters are assigned to each sensor where "000" means that a finger is straight and "900" means that it is fully curved. In a program we developed, we assign a threshold value of when finger becomes bent, which means that we detect only two states of the finger. By mapping finger shapes into the gestures it is easy to detect different kinds of positions like

pointing or five fingers straight.

NaturalPoint OptiTrack FLEX3

Infrared reflecting materials are used to help detecting the approximate pointing direction of the user's right hand. Material in a shape of a strap is put around the wrist of the right hand and its spatial coordinates are detected by the OptiTrack infrared camera and projected into the 2D plane.

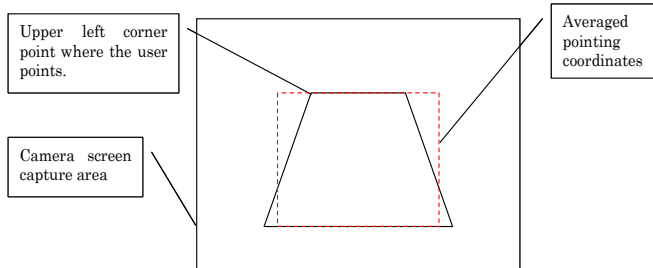


Fig. 3 The calibration of project 2D coordinates

During initial use, system must be calibrated. To calibrate the system, user stands in front of the projection screen and a camera and follows the software instructions to point to the each corner of the projection screen. Projected coordinates of the upper corners are bent inward and give overall projection shape resembling the trapezoid because camera stands closer to the floor. The concept is shown in figure 3. That inevitably leads to the curved projection of the hand movement. Nevertheless, it is assumed that interesting scenario objects will have perceivable size compared to the projection screen and that approximate pointing coordinates should be detectable.

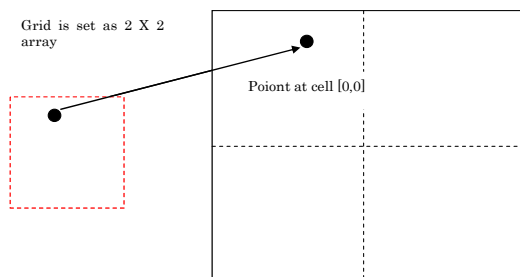


Fig. 4 Mapping between raw data and application coordinates

After the trapezoid's corner points are calculated we approximate the area of hand movement with a rectangle by averaging trapezoid's neighbor values. The projection screen is divided into the grid of arbitrary size. When user is pointing to the screen, his pointing coordinates are inside one of the grid's cell. Interesting scenario objects are mapped to distinct cells and therefore pointing at scenario object can be easily detected. This concept is shown in figure 4.

Camera's API gives us the information of where the detected object's center of mass is. If multiple infrared sources exist, more than one center of mass will be detected. This can especially occur when marker, due to the various reasons, has interruption in continuity of its area. To remove that problem, we use K-Means algorithm to group close centers. Software also

removes all centers which show spatial stability in time. Such an example is a LCD projector which is put in front of the camera and disturbs the detection by constantly emitting the infrared light.

Software detects hand stability and waving. To detect if user holds his hand in a stable position, we need a time buffer of marker coordinates. Buffer size is determined by the camera's frame rate. After each frame algorithm calculates the central point of all points in a buffer and distance from that central point to each other point in a buffer. If all distances are below the predefined percent of screen width, hand stability is alerted.

Swing is defined as a hand movement from one point to another, before hand changes movement direction. Software detects waving which goes from the elbow to the forearm, where the marker's waving is visible on the screen and recognizable by the system. Waving pattern is characterized with two specific features. First one is that a length of each swing is approximately the same as a previous one. Also, we tend to wave in a constant speed without variations in a hand movement. Therefore, gradient of a hand movement and movement length seem like a suitable features to detect the waving. To detect the hand movement, buffer is filled with the waving data. Each change in direction of the marker that is larger than some threshold is detected as a new swing. Pivot element, first element in a buffer, is taken as a reference point to calculate if waving is occurring. By comparing other waving elements found in a buffer with a pivot element, specifically their gradient, length and number of swings, we may detect and signalize the waving.

NEC/Tokin 3D Motion Sensor

It is a sensor that can detect the change of acceleration in three dimensions. This small-size sensor is attached on the top of the headset that is usually used for gathering speech inputs, and the data from it is used to detect the movement of the user's head. A component that detects two types of head movements, nodding and shaking was developed. It generates output message to represent positive (nodding) and negative (shaking) verbal answers. Therefore, the user can nod instead of saying "yes" or shake his (her) head instead of saying "no."

Data glove and hand movement detection programs work as a separate .NET applications. Each program has an OpenAir plug implementation, sending the data which *InputManager* component receives and combines into the new output (see Table 1). *InputManager* component acts as a fusion component of the different input modalities, and is made as a simple state machine. It sends the output only when new gesture is detected.

TABLE 1
THE USE OF SENSOR DATA

Component name	finger_shape		
Outputs	Type	Content	Description
in-put.shape.fingers		FiveUps	Five fingers straight
		Pointing	2 finger straight
		Victory	2,3 fingers straight
		Unknown	Any other combination
Component name	wrist_position		
Outputs	Type	Content	Description
in-put.position.wrist		UNKNOWN[X,Y or OUTSIDE], where the X,Y is the cell where the marker is detected	
		WAVING_DETECTED[X,Y OUTSIDE]	
		HAND_STABLE[X,Y OUTSIDE]	
		CALIBRATION[calibration message]	Sends the message in which status the calibration is
Component name	input_manager		
Outputs	Type	Content	Description
in-put.manager.fusion		POINTING[X,Y OUTSIDE]	Pointing + HAND_STABLE + coordinates
		WAVING	FiveUps + WAVING_DETECTED
		ATTENTION	FiveUps or Victory + HAND_STABLE
		UNKNOWN	

C. Croatian Speech Input / Output

As there is no Croatian speech recognizer it is decided that we will use English speech recognizer to recognize Croatian. Therefore simple rule grammar had to be created to recognize some words according to the agent scenario. The grammar for Croatian is defined by using English alphabet to approximate the pronunciation of Croatian even those words do not exist in English. CloudGarden [10] library is used to access SAPI5 compliant Speech-Recognition engine by standard Java Speech SAPI.

It was possible to create grammar according to the scenario to make English speech recognizer to recognize Croatian. We have made such grammar to recognize both only Croatian words and whole sentences. Both ways were quiet successful. However, several problems came out. Some Croatian words were impossible to write with English alphabet, therefore it was better to avoid them and use some other words instead. Also, if grammar contained several very similar words, they were sometimes mixed by recognizer, so it is better to choose words that are not so similar (since scenario was not that strict this was possible). And the last thing, although the recognition worked with all tested subjects, recognition with some was slightly

better.

Once we had Croatian scenario, a native Croatian speaker has recorded speech the agent was supposed to say in certain situation in the noise free room. By applying a lip sync application we have [11], which takes speech as input and gives animation (of the lips) as output, we have created animation from the prepared speech files.

Our automatic lip sync system determines the motion of the mouth and tongue during the speech by speech signal analysis. Neural networks are used to classify the speech into a sequence of visemes (visual representatives of phonemes). In order to obtain training data for the NNs, a training set with visemes was collected. The speech is first preprocessed. Input in NNs are MFCCs calculated from training data and output is different viseme classes. When correct viseme is chosen, it can be sent to animated face model. MPEG-4 standard is used for generating facial animation since facial animation can be generated for any parameterized face model if the visemes are known. The method is implemented in C++. The program reads speech from pre-recorded audio files and continuously performs spectral analysis of the speech. Suitable visemes are shown on the screen or saved in the FBA file.

At the end, we had a pair of speech-animation files for every situation according to the scenario.

D. Action Animation Database

By an animated action we mean a set of Face and Body animation parameters displayed within the same time interval to produce a visual agent action, such as nod or gesture. The queries for animated actions for the agent are stored in the AIML script. A single database query corresponds to an AIML category consisting of a *pattern* (typically, a human's action) and a *template*, the agent's reaction to the pattern. The description of an animation, which is to be started simultaneously with a particular part of the agent's utterance, is incorporated in the `<template>` tag using the "[" and "]" characters.

Below is a simple example of an AIML category with non-verbal input/output descriptions:

```
<category>
<pattern>What is this
[PointingAt Object="monastery"]
</pattern><template>This is the big Onofrio's Fountain
[Action Type="pointing" SubType="Null" Duration="2300"
Intensity="0" X="0" Y="0" Z="0" Direction="rightUp"
ActivationFunction="sinusoidal"/] built in 15th
century. The Fountain is a part of the town's water
supply system which Onofrio managed to create by
bringing the water from the spring located 20 km away
from town.</template></category>
```

Here, the non-verbal action "pointing" of the agent character is described. Its duration is specified by opening tag and closing tags that enclose a segment of an utterance and thus the actual value depends on the TTS (Text-To-Speech) synthesizer if it supports prior phoneme timing output or absolute values in milliseconds. The attribute `SubType` has the value of "Null", as there are no possible subtypes defined for it. The "Intensity" attribute is to have integer values, with "0" value meaning that

the intensity is not specified for the action in question. Other actions, for which the attribute “intensity” has sense, do have an intensity scale specified. For example, to distinguish between a slight bow used while greeting in European countries from the deep Japanese salutation bow, we introduce a scale of values for the “bow” action.

Further, we envisage the use of the coordinates (“X”, “Y”, “Z”) integer-valued attributes in the future. The meaning of these coordinates will be dependent on the action. For example, for the “pointing” action such this triad would mean the position on the background screen where the agent is supposed to point. At the moment the alternate attribute “Direction” is used.

The “ActivationFunction” attribute stands for the dynamics of the action. Possible values are “linear”, which uses a linear function to activate the corresponding MPEG4 Face and Body Animation parameters (FAPs), “sinusoidal”, which uses trigonometric functions to activate the FAPs, and “oscillation” function, which is used for the repeated actions, such as “Nodding” or “HeadShaking”. In addition to these attributes, the attribute “sync” with possible values “PauseSpeak”, “BeforeNext”, “WithNext” specifies the synchronization between non-verbal actions and speech synthesizer.

The action “pointing” is an invocation of one character action with the name “pointing” which is stored in a high-level action database. The database is currently implemented as one part of the visage animator and stores low-level MPEG4 FBA parameter specifications as well as run-time configurable parameters for high-level action invocations.

Visage operates with FBAPs – a large set of face and body animation parameters, specified by MPEG4. FBAPs are divided into FAPs (Face animation parameters) and BAPs (body animation parameters). The BAP parameters are the angles of rotation of body joints connecting different body parts, such as toe, ankle, knee, hip, spine joints, shoulder, clavicle, elbow, wrist, and the hand fingers. There are 64 low-level FAPs, which are closely related to muscle actions and represent a complete set of basic facial actions, and two high-level FAPs (expression and viseme).

Example: by analyzing the videos, we found that for the pointing gesture can be animated by manipulating such FAPs as head_yaw and head_pitch, and the following BAPs: l_shoulder_flexion, l_shoulder_abduct, l_shoulder_twisting, l_elbow_flexion. We ran a series of experiments to set the values of these parameters.

The analysis of videos to create gestures was pretty difficult because we did not use any tool that would translate a gesture in to a set of FBAPs. Without it took us from 5 to 30 experiments, depending on the complexity of the action, to adjust the parameters for an action. Needless to say, that such approach is rather time-consuming.

Then, for the most of the actions implemented in our system, we divide the duration of it into 3 states: attack, sustain and decay. For each of the intervals, depending on the activation function of the action in question, we define how the parameter value changes as a function of time. For example, in case of sinusoidal function (as is with the pointing gesture), in the

attack phase, the value of the parameters changes as a sinusoidal (increasing function) function of time, whereas in the sustain (i.e. peak) phase it changes as a constant. Finally, in the decay phase the corresponding function is cosinusoidal (decreasing). The character animation created for this application is listed in Table 2.

Cultural differences through the non-verbal behavior of the agent

We observed analyzed non-verbal behaviors of Japanese tour guides and took videos. As a result, we tried to implement these features in our agent. Also, some very typical emblem Japanese gestures, that are not inherent to the European culture, were implemented. For example, the so-called “handsCrossed” gesture. This gesture seems to be pretty unique, and normally draws attention of Western people who first come to Japan and are used to head shaking or simply verbal expression of prohibition (See Figure 5 and Figure 6). In Japanese culture, to show that something is prohibited, people tend to cross their hands before the chest. Sometimes, this action is accompanied with head shaking. Similarly, our agent uses this gesture when prohibiting in the Japanese mode, in contrast to the European mode, where only head shaking is envisaged.



Fig. 5 A Japanese emblem gesture to show prohibition



Fig. 6 Another example is the “Negation” gesture in the Japanese mode: waving with a hand while the arm is extended. In Japan, the negation is expressed by shaking one’s upright hand near one’s mouth with two thumbs closer to one’s face. Sometimes shaking head sideways is also added. When asking to wait, Japanese people usually show the palm of one hand to another person. At times, both hand maybe used.

TABLE 2
CHARACTER ACTION ANIMATION CREATED FOR THE DUBROVNIK GUIDE APPLICATION

Gesture/attributes	Values	Meaning
Pointing: - Direction	left, leftUp, right, rightUp, rightForward, leftForward, backH, backE	The agent points in the directions that semantically correspond to the values defined for this attribute. In future, we plan to use the coordinates on the screen to which the agent should point. Thus using direction instead of the coordinates is a temporarily solution. “backE” and “backH” values represent variations of gestures with the elbow bent.
Bow -Intensity	1-3	1 corresponds to a shallow bow, using only head; 2- is a deeper bow, very frequently used by Japanese people in a daily conversations, 3-corresponds to a very polite bow, showing a high respect to the listener
Invite -Subtype	Croatian, Japanese	The “invite” action of the “Croatian” subtype is waving upwards and then backwards with the left hand, a somewhat informal emblem gesture meaning inviting. The action of the subtype “Japanese” has not been implemented yet.
HandsCrossed		This is an emblem Japanese gesture, meaning that something is not allowed. The hands are crossed in front of the lower part of the chest
Nodding		The action meaning both in Croatian and Japanese agreement, consent.
ShakeHead		The action meaning both in Croatian and Japanese negation or disapproval.
Extend	[at the moment, “extend” means extending the right arm. In the future we might need extending the left arm as well. Thus, the subtype attribute might be introduced with the “left/right” as possible values.]	This action means right arm extended with the palm open and oriented upwards. The meaning in the Japanese culture is “wait please”
Wave	[at the moment, “wave” means waving with the right hand. In the future we might need waving with the left hand as well. Thus the subtype attribute might be introduced with the “left/right” as possible values.]	This action means oscillating right hand waving. Used in combination with the “extend” action as part of the Japanese gesture meaning “No. This is not true”.
Expression	“smile”	This value corresponds

-Subtype		facial expression “joy” defined in the Class SimpleFacialExpression
Walk -Direction	“right”, “left”, “back”	The agent walks in the directions that semantically correspond to the values defined for this attribute. In future, we plan to use the coordinates on the screen to which the agent should walk. Thus using direction instead of the coordinates is a temporarily solution.
Beat -Subtype	“a” – “e”	Waving spontaneous gestures with either one or both arms, used by the BEAT engine.
Contrast -Subtype	“a” – “c”	Waving spontaneous gestures with either one or both arms, used by the BEAT engine. NOT IMPLEMENTED YET
Warning		An emblem gesture meaning danger : the elbow bent and the hand raised. In future, the finger feature needs to be implemented, i.e. the pointing finger only pointing upwards.

In addition to the character action specification tags, animator controlling tags such as “Scene” are also defined. This tag informs the animator to switch scene settings while the scenario advances. Besides, the “PointingAt” tag is generated by a component which maps raw coordinate data to object names according to the information provided by motion capture device and scene changing messages.

E. Character Animator

During the workshop, the main improvement upon the character animator is the adoption of ARToolkit [12] to align the position of the character with the background images.

The ARToolKit 2.65 video tracking libraries capture real time input from real camera and detect presence of the marker in the input picture. If marker is detected, real camera position and orientation relative to physical markers in real time are calculated. Calculated parameters are used for rendering a virtual object on physical marker.

The main idea is to use ARToolkit libraries in one separate application that will recognize marker on the static picture (background picture). Marker will define the a position of the agent character on the picture. As output, this application will give calculated parameters that will be used in Visage player for rendering the agent character aligned with a same background, but without marker on the picture. In order to realize this, two pictures of the same background had to be taken, one with and another without marker.

At first, some modifications to existing *Simple* demo application, created by ARToolkit developers, had to be made. *Simple* application is programmed in Visual Studio 6.0 in C

program language. Application is considering video stream in input and during continuous reading of sequential pictures from frames it is recognizing marker in each picture, if present. Markers on the picture are distinguished by pattern objects that are previously saved by another application, called *Make pattern*. If the marker is recognized, parameters of the modelview matrix and projection matrix needed for drawing virtual object on a marker are calculated for every picture. After that, function for drawing virtual object on physical marker is called. As result of *Simple* application, an output window where input picture align with virtual object on the marker, if present, is displayed.

In order to read a static picture from specified location instead of real time input from camera, eight functions that are receiving video input in ARToolkit are changed to return NULL value. Size of the display window that was calculated automatically by video functions is increased by decreasing zoom value while displaying main window. As size of a picture, a constant value 1250x 937 is set. For reading static pictures into unsigned bytes is used Sourceforge DevIL library.

After these modifications several tests of a *Simple* application with various .jpg pictures are done. Pictures are made by camera and are differed in the position and rotation of the marker. Marker is put on the floor with a different distance relative to the camera and also on top of the tripod. These pictures are then used as input value to *Simple* application in order to check percentage of detection. When application is started, specified picture is read into unsigned bytes. After that, read bytes are checked if specified marker (saved by *Make pattern*) is present. If marker is detected, as output of modified *Simple* application, static picture with virtual object on the marker is displayed. In the same time, calculated parameters are saved in .txt file.

As a result of the tests, in output window only 20% of markers on various pictures are detected. Results of these tests were not satisfying.

In the all pictures that were detected in previous test, markers were put in the front of the camera on the floor and were lightened. Pictures with marker on the tripod had also good results, but they were not considered in later work because of the susceptibility of a marker position. If marker is slightly moved on the tripod, virtual object that is rendered on the picture is moved. Besides that, marker attached on the tripod was slightly rotated to the floor, so local coordinate system of a virtual object wasn't aligned to the floor. Conclusion of this test is to use a bigger marker and to lay it on the floor without presence of any shadow on the marker.

Further, different results of detection were also noticed in the two other things. In *Simple* application threshold value of an input picture can be changed manually. In some of the pictures that had bad results, marker was detected for changed threshold values. Second, new *Make pattern* application is made. Original ARToolkit version of the *Make pattern* application was used to save marker pattern from picture captured by USB camera. New *Make pattern* application that is created can detect and save presence of marker from a static picture. There-

fore, two different pattern objects for one marker can be used as input of *Simple* application.

Input value of final application is set of picture's names to be processed. After one picture is being read, presence of the marker on the picture is checked for different threshold values and both patterns made by *Make Pattern* application. This parameters are changed automatically in the application until marker on the picture is recognized. Output of the application is display window with static picture and virtual object and text file with parameters of the modelview matrix and projection matrix for each input picture. Output text file can be used by Visage player for positioning the agent character depending of the loaded background picture.

Output values, parameters of final background image alignment application give very good results in alignment of the agent character to the background image in Visage Player. However, this result can be used for only one picture separately. Later improvements of this system can include continuous scene transitions. The main idea is to generate continuous scenes while agent is walking, e.g. walking around circular fountain. In order to do realize this idea, detection of two different markers on the set of static pictures has to be included in *Simple* application. Set of static pictures has to be made while moving camera and continuously changing position of one marker after another, like character is walking. After this, output values of application will be two pairs of parameters for each picture. Later, these parameters can be used in Visage Player to calculate position of virtual character while moving from one picture to another.

V. PROJECT OUTCOME AND CONCLUSION

To the end of the workshop, we could not manage to achieve all of the planned project objectives. The individual non-verbal input components and Croatian speech contents are completed but are not integrated into the system. The final demonstration was done with a Dubrovnik guide agent running in two modes, English and Japanese modes. In Japanese mode, since there is a spontaneous gesture generating component, the agent's presentation looks more natural because the beating gestures performed at the timing generated by CAST engine. On the other hand, in English mode, the agent performs scripted gestures only.

However, this is our first time to apply the GECA framework to a larger and non-trivial testing application. We got some valuable experiments in component development and message specifications. Automatic character-background alignment application is developed and a suitable parameter set for dynamically configurable character animation is explored.

REFERENCES

- [1] OpenAIR protocol, <http://www.mindmakers.org/openair/airPage.jsp>
- [2] NaturalPoint OptiTrack Flex 3, <http://www.naturalpoint.com/optitrack/>
- [3] Julius Japanese speech recognition engine, <http://julius.sourceforge.jp/en/julius.html>
- [4] NEC/Tokin 3D motion sensor, <http://www.nec-tokin.com/english/product/3d/index.html>

eNTerFACE'06, July 17th – August 11th, Dubrovnik, Croatia — Final Project Report

- [5] Nakano, Y., Okamoto, M., Kawahara, D., Li Q., Nishida, T.: Converting Text into Agent Animations: Assigning Gestures to Text, in *The Proceedings of The Human Language Technology Conference (HLT-NAACL04)*, 2004.
- [6] Program D, http://www.aitools.org/Program_D
- [7] AIML (Artificial Intelligence Markup Language), <http://www.alicebot.org/>
- [8] Cassell, J., Vilhjalmsson, H., Bickmore, T.: BEAT: the Behavior Expression Animation Toolkit, in *The Proceedings of SIGGRAPH '01*, pp.477-486, 2001.
- [9] visage|SDK, visage technologies, <http://www.visagetechnologies.com/index.html>
- [10] TalkingJava SDK, CloudGarden.com, <http://www.cloudgarden.com/index.html>
- [11] Zoric, G., Pandzic, I.S.: A Real-time Language Independent Lip Synchronization Method Using a Genetic Algorithm, in the proceeding of ICME 2005, 6-8 July 2005.
- [12] ARToolKit, <http://artoolkit.sourceforge.net/>

Principal investigator:

Hung-Hsuan Huang graduated from the Computer Science Department of National Chen-Chi University, Taiwan in 1998 and obtained his master degree of computer science and information engineering from National Taiwan University, Taiwan in 2000. After a two-year military service where he was the political warfare director and the co-commander of an army company, he came to Japan. After the learning in a Japanese language school for one year, he entered the Ph.D. course of the Graduate School of Informatics of Kyoto University, Japan in 2003. His research interests include intelligent software agent, information visualization, photo management, gesture interface and is working on the generic ECA platform topic.

Project Advisors:

Prof. Toyooki Nishida received the B.E., the M.E., and the Doctor of Engineering degrees from Kyoto University in 1977, 1979, and 1984 respectively. In 1980, he joined Department of Information Science, Kyoto University as an Assistant Professor. In 1988, he was promoted as an associate professor. In 1993, he joined Graduate School of Information Science Nara Institute of Science and Technology as Professor. During 1998-2003, he led the Breakthrough 21 Nishida Project, which is a five-year project sponsored by Ministry of Posts and Telecommunications, Japan. In 1999, he moved to the University of Tokyo as Professor. In 2004, he moved to the current position at Kyoto University. His research area covers artificial intelligence in general. He has been working on natural language understanding, spatial reasoning and qualitative reasoning. His current research focusing on knowledge communication, including conversational knowledge process, knowledge sharing, and qualitative reasoning.

Asst. Prof. Igor S. Pandzic 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. He obtained his PhD from MIRALab, University of Geneva, Switzerland in 1998. In the same year he worked as a visiting scientist at AT&T Labs, USA. In 2001-2002 Igor was a visiting scientist in the Image Coding Group at the University of Linköping, Sweden. He is now an Assistant Professor at the Department of Telecommunications, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. 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. 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.

Asst. Prof. Yukiko Nakano received her bachelor degree in psychology from Tokyo Women's Christian University, Japan in 1988, one master degree in educational psychology from the University of Tokyo and the other one in media arts and sciences from Massachusetts Institute of Technology, USA in 1990 and 2002, respectively. She obtained her Ph.D. in information science and technology from the University of Tokyo, Japan in 2005. Yukiko was a researcher of NTT Research Laboratories from 1990 to 2000 and was a sub-leader researcher of Research Institute of Science and Technology for Society from

2002 to 2005. She moved to the Department of Computer, Information and Communication Sciences of Tokyo University of Agriculture and Technology as an associate professor in 2005. With her special interest in Embodied Conversational Agents (ECA), she has been studying human face-to-face communication in psychology and communication science, and creating multimodal conversational interfaces based on a model of human communication behaviors.

Team Members:

Kateryna Tarasenko graduated from the National Technical University of Ukraine "Kiev Polytechnic Institute" with a master degree in "Intelligence Systems for Information processing and Decision making". She entered the Graduate School of Informatics of Kyoto University, Japan as a research student in 2005. Her research interests include: embodied conversational agents, simulation of non-verbal communication behaviors.

Goranka Zoric received her master degree from the Faculty of Electrical Engineering and Computing, the University of Zagreb, Croatia in 2005 and is currently both a PhD student and a research associate there. Her main interest is in the field of facial animation and its application with Internet and mobile technologies and virtual environments with particular focus on automatic lip synchronization and gesturing of synthetic 3D avatars based only on the speech input.

Vjekoslav Levacic graduated from the Faculty of Electrical Engineering and Computing of University of Zagreb, Croatia in 2005 and is currently a graduate student of the same university. His research interests include multimedia, software architecture, web design, image processing, HCI and mobile networks. (Vjekoslav will stay in Dubrovnik only for the first two weeks)

Aleksandra Cerekovic is in her fifth and final year as an undergraduate student of the Faculty of Electrical Engineering and Computing of University of Zagreb, Croatia. She worked on the topic of lip synchronization for real-time speech recognition and synthesis. Besides that, she has the research interests on the topics of embodied conversational agent and virtual environment.

Margus Treumuth received his bachelor and master degrees in computer science from University of Tartu, Estonia, in 2002 and 2004 respectively. He is now a PhD student in University of Tartu and has research interests in computational linguistics and dialogue systems. (Margus will leave on 22/07)

Multimodal tools and interfaces for the intercommunication between visually impaired and “deaf and mute” people

Konstantinos Moustakas, Georgios Nikolakis, Dimitrios Tzovaras, Benoit Deville, Ioannis Marras and Jakov Pavlek

Abstract— The present paper presents the framework and the results of Project 2: “Multimodal tools and interfaces for the intercommunication between visually impaired and “deaf and mute” people”, which has been developed during the eNTERFACE-2006 summer workshop in the context of the SIMILAR NoE. The developed system aims to provide alternative tools and interfaces to blind and deaf-and-mute persons so as to enable their intercommunication as well as their interaction with the computer. All the involved technologies are integrated into a treasure hunting game application that is jointly played by the blind and deaf-and-mute user. The reason for choosing to integrate the multimodal interfaces into a game application is that it serves both as an entertainment and as a pleasant education tool to its users. The proposed application integrates haptics, audio, visual output as well as computer vision, sign language analysis and synthesis, speech recognition and synthesis, in order to provide an interactive environment where the blind and deaf and mute users can collaborate in order to play the treasure hunting game.

Index Terms—Multimodal interfaces, Rehabilitation technologies, Virtual Reality.

I. INTRODUCTION

DURING the latest years there has been an increasing interest in the Human-Computer Interaction society for multimodal interfaces. Since Sutherland's SketchPad in 1961 or Xerox' Alto in 1973, computer users have long been

acquainted with more than the traditional keyboard to interact with a system. More recently with the desire of increased productivity, of seamless interaction and immersion, of e-inclusion of people with disabilities, as well as with the progress in fields such as multimedia/multimodal signal analysis and human-computer interaction, multimodal interaction has emerged as a very active field of research (e.g. [1], [2]).

Multimodal interfaces are those encompassing more than the traditional keyboard and mouse. Natural input modes are put to use (e.g. [3], [4]), such as voice, gestures and body movement, haptic interaction, facial expressions and more recently physiological signals. As described in [5] multimodal interfaces should follow several guiding principles: multiple modalities that operate in different spaces need to share a common interaction space and to be synchronized; multimodal interaction should be predictable and not unnecessarily complex, and should degrade gracefully for instance by providing for modality switching; finally multimodal interfaces should adapt to user's needs, abilities, environment.

A key aspect in multimodal interfaces is also the integration of information from several different modalities in order to extract high-level information non-verbally conveyed by users. Such high-level information can be related to expressive, emotional content the user wants to communicate. In this framework, gesture has a relevant role as a primary non-verbal conveyor of expressive, emotional information. Research on gesture analysis, processing, and synthesis has received a growing interest from the scientific community in recent years and demonstrated its paramount importance for human machine interaction (see for example the Gesture Workshop series of conferences started in 1996 and since then continuously growing in number and quality of contributions; a selection of revised papers from the last workshop can be found in [6]).

The present work aims to make the first step in the development of efficient tools and interfaces for the generation of an integrated platform for the intercommunication of blind and deaf-mute persons. It is obvious that while multimodal signal processing is essential in such applications, specific issues like modality replacement and enhancement should be addressed in detail.

In the blind user's terminal the major modality to perceive a

K. Moustakas is with the Informatics and Telematics Institute Centre for Research and Technology Hellas, 1st Km Thermi-Panorama Str. 57001 Thermi-Thessaloniki, Greece and the Aristotle University of Thessaloniki (moustak@iti.gr)

G. Nikolakis is with the Informatics and Telematics Institute Centre for Research and Technology Hellas, 1st Km Thermi-Panorama Str. 57001 Thermi-Thessaloniki, Greece (gniko@iti.gr)

D. Tzovaras is with the Informatics and Telematics Institute Centre for Research and Technology Hellas, 1st Km Thermi-Panorama Str. 57001 Thermi-Thessaloniki, Greece (tzovaras@iti.gr)

B. Deville is with Computer Vision and Multimedia Lab, University of Geneva, Geneva, Switzerland (e-mail: Benoit.Deville@cui.unige.ch).

I. Marras is with the Aristotle University of Thessaloniki, Informatics School AIIA Lab. Thessaloniki Greece (e-mail: imarras@aia.csd.auth.gr)

J. Pavlek is with the Faculty of Electrical Engineering and Computing, Zagreb, CROATIA (e-mail: jakov.pavlek@fer.hr)

virtual environment is haptics while audio input is provided as supplementary side information. Force feedback interfaces allow blind and visually impaired users to access not only two-dimensional graphic information, but also information presented in 3D virtual reality environments (VEs) [7]. The greatest potential benefits from virtual environments can be found in applications concerning areas such as education, training, and communication of general ideas and concepts [8]. Several research projects have been conducted to assist visually impaired to understand 3D objects, scientific data and mathematical functions, by using force feedback devices [9].

PHANToMTM is one of the most commonly used force feedback device; it is regarded as one of the best on the market. Due its hardware design, only one point of contact at a time is supported. This is very different from the way that we usually interact with surroundings and thus, the amount of information that can be transmitted through this haptic channel at a given time is very limited. However, research has shown that this form of exploration, although time consuming, allows users to recognize simple 3D objects. The PHANToMTM device has the advantage to provide the sense of touch along with the feeling of force feedback at the fingertip. Another device that is often used in such cases is the CyberGrasp that combines a data glove (CyberGlove) with an exoskeletal structure so as to provide force feedback to each of the fingers of the user (5DoF force feedback, 1DoF for each finger). In the context of the present work we used the PHANToMTM desktop device to enable haptic interaction of the blind user with the virtual environment.

Deaf and mute users have visual access to 3D virtual environments; however their immersion is significantly reduced by the lack of audio feedback. Furthermore effort has been done to provide applications for the training of hearing impaired. Such applications include the visualization of the hand and body movements performed in order to produce words in sign language as well as applications based on computer vision techniques that aim to recognize such gestures in order to allow natural human machine interaction for the hearing impaired. In the context of the presented framework the deaf-mute terminal incorporates sign-language analysis and synthesis tools so as to allow physical interaction of the deaf-mute user and the virtual environment.

The paper is organized as follows. Section II describes the overall system architecture and the objectives, Sections III and IV describe the SeeCoLoR and the haptic interaction modules respectively. In Section V the sign synthesis and sign recognition systems are briefly described. In Sections VI and VII the automatic grooved map generation and partial matching algorithm are described, respectively. The entertainment scenario is described in Section VIII. Finally, in section IX the conclusions are drawn.

II. OVERALL SYSTEM DESCRIPTION

The basic development concept in multimodal interfaces for the disabled is the idea of *modality replacement*, which is the

use of information originating from various modalities to compensate for the missing input modality of the system or the users.

The main objective of the proposed system is the development of tools, algorithms and interfaces that will utilize modality replacement so as to allow the communication between blind or visually impaired and deaf-mute users. To achieve the desired result the proposed system combines the use of a set of different modules, such as

- Gesture recognition,
- Sign language analysis and synthesis,
- Speech analysis and synthesis,
- Haptics,

into an innovative multimodal interface available to disabled users. Modality replacement was used in order to enable information transition between the various modalities used and thus enable the communication between the involved users.

Figure 1 presents the architecture of the proposed system, including the communication between the various modules used for the integration of the system as well as intermediate stages used for replacement between the various modalities. The left part of the figure refers to the blind user's terminal, while the right refers to the deaf-mute user's terminal.

The different terminals of the treasure hunting game communicate through asynchronous TCP connection using TCP sockets. The following sockets are implemented in the context of the treasure hunting game.

- SeeColor terminal: Implements a server socket that receives queries for translating color into sound. The code word consists of the following bytes, “b;R;G;B”, where b is a boolean flag and R, G, B the color values.
- Blind user terminal: Implements three sockets.
 - A client socket that connects to the SeeColor terminal.
 - A server socket to receive messages from the deaf-mute user terminal
 - A client socket to send messages to the deaf-mute user terminal
- Deaf-mute user terminal: Implements two sockets
 - A server socket to receive messages from the blind user terminal
 - A client socket to send messages to the blind user terminal

Also file sharing is used to ensure consistency between the data used in the various applications.

III. SEECOLOR

SeeCoLoR is meant to be a mobility aid for visually impaired and blind people. Its main interest resides in the fact that colours will henceforth be accessible to these disabled people using their hearing. As you will see later, this is done using an association between colours and musical instruments. The project #2 being an implementation of different multimodal interfaces to make possible the communication

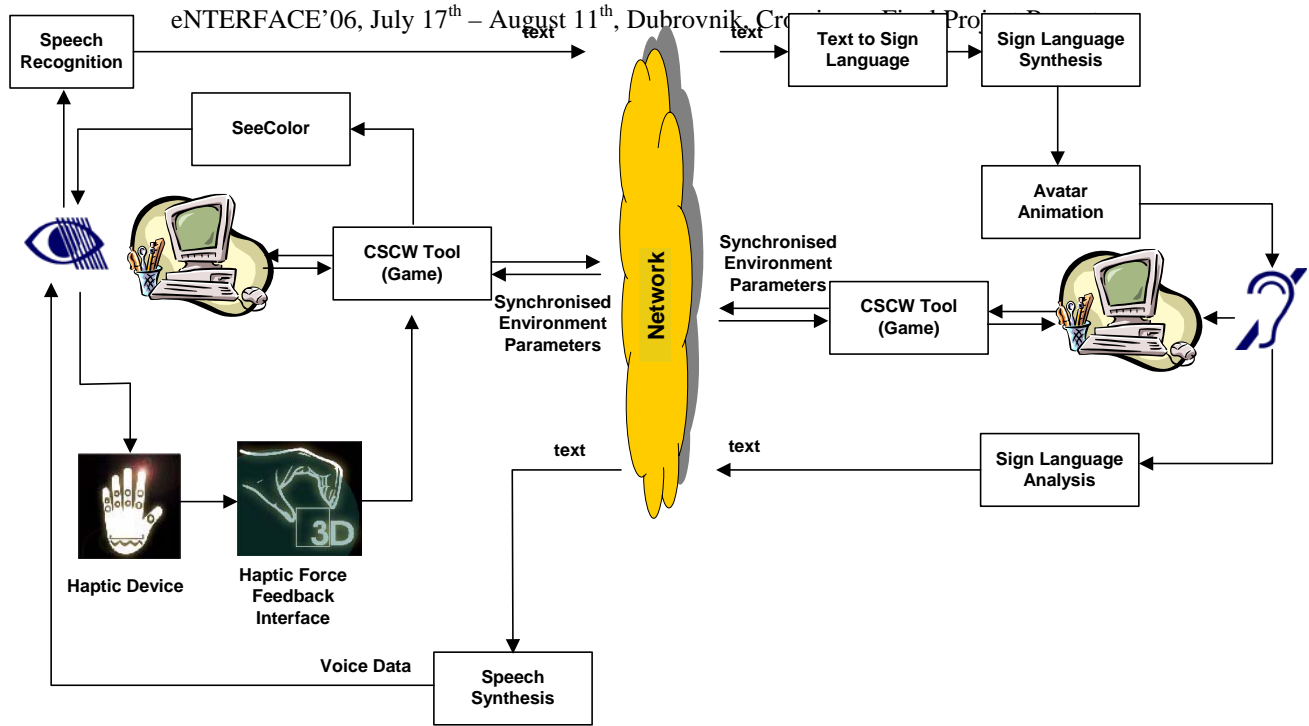


Figure 1. Block diagram showing the intercommunication between the various modalities used to provide communication between visually impaired (left part of the diagram) and deaf-mute users (right part).

between two different communities of disabled people, one of them being the visually impaired community, the interest of the integration of SeeCoLoR was obvious.

We will first describe the SeeCoLoR system from a global point of view, and its objectives. Then we will go into details, explaining how works the colour to instrument mapping. Finally, the way it has been integrated into the treasure hunt game for blind and deaf-and-mute people will be pictured, as for the adaptations we made to simplify and adapt it to the game.

A. SeeCoLoR system

Unlike other systems, SeeCoLoR's colour to sound mapping includes musical instruments. In fact, most of actualization systems use either artificial sounds [10], [11], [12], [13] or meaningful sounds [12], [14], i.e. particular sounds or texts that are supposed to describe an object. For example, a car could be represented by the sound of a motor engine, or a horn sound; toilets could be well described by a flush sound [12]; and the leaves moving in winds are easily understood as being trees.

Each approach has qualities and drawbacks. For example, it is easy to describe edges using artificial sounds, but constantly listening to them is tiring, even annoying. There is no evidence that these sounds should not be used in a sensory substitution system, but it was intuitively thought that such a system should only produce sounds that are common to users. These sounds should be pleasant to hear. This is why common musical instruments were chosen.

The meaningful sound approach is really useful for known environments, and searching particular objects. Although easier to learn, such a system is useless in a new environment,

or with new objects. This is a classification problem, and in an unknown situation, one never knows how the system would react. This is the reason why we prefer to let the human brain makes its own deductions, and only sonify colour information.

B. Colour-instrument mapping

The system uses the HSL (Hue - Saturation - Lightness) colour system to define colours. The hue represents the pigment of the colour. Its value is expressed in degrees, from 0° to 360°. The variations of saturation from 0 to 1 make the colour go from grey to really intense colour. The lightness expresses the quantity of light, from black to white, and it is also a decimal value in [0,1]. Figure 2 illustrates this colour system.

Each dimension of this colour space is mapped to an auditory dimension. First, the hue is quantified into seven colours: red, orange, yellow, green, cyan, blue, and magenta. Each colour is then mapped as a linear combination of two particular instruments, according to the table 1. A pure colour is of course played by only one instrument timbre.

The pitch of the played note depends on the colour saturation, which is divided into four intervals. The four notes are C, G, B ♭, and E, thus making a dominant seventh chord.

Finally, the luminance –or brightness, or lightness– of the colour is represented by two different instruments with varying pitch. Dark and clear colours will be played respectively by the double bass and synthesized human voice. Here the pitch depends on the lightness value. The lightness interval is divided into eight parts; the four darker ones are played by the double bass and the four other by the synthesized human voice. The notes are the same as for the saturation, therefore a dominant seventh chord.

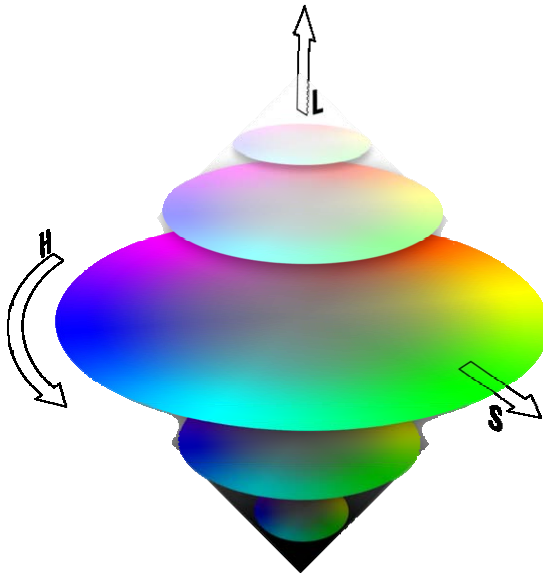


Figure. 2. HSL Double Cone.

TABLE 1
SEECOLOR'S COLOUR-TO-INSTRUMENT MAPPING

Colour	Instrument
red	oboe
orange	viola
yellow	violin (pizzicato)
green	flute
cyan	trumpet
blue	piano
magenta	Saxophone

This system has been tested on five sighted persons, and some useful conclusions have been made. First, some instruments are difficult to differentiate. Most people never clearly heard the viola, or had problems to distinguish all or part of wind instruments, i.e. saxophone, oboe, flute or trumpet. However, it was obvious that it was easy to say if the colour was either dark or bright. On the other hand, pitch modulations remained abstruse to them, both for saturation and lightness. All these observations will lead to the creation of a new orchestra in the near future.

C. Integration

In the context of the eNTERFACE'06 project, we decided to simplify the SeeColOr system. First, the haptic device gives us one pixel, not a window. Actually, SeeColOr's configuration computes pixel values over a 17 x 9 pixel window. This is done in order to sonify more colours at the same time, and to spatialize them in 3D using a virtual AmbiSonic approach, so that different sources of colour can be distinguished. Since this part has not been completely finished and validated, we have decided to leave it away.

We then have an adaptation of SeeColOr playing only one instrument at the same time. To achieve this, we decided to

quantize the hue wheel into the six primary colours –red, yellow, green, cyan, blue, and magenta–, and grey, from black to white. Thus, learning the correspondences will be easy for blind users. Indeed, too much instruments would be difficult to learn and recognize, since we only play on the timbre of the instrument. Furthermore, the choice of instruments was reduced, due to different causes.

TABLE 2
CONDITIONS LEADING THE GAME'S COLOUR-TO-INSTRUMENT MAPPING

Condition	Colour	Instrument
$S < 0.05$ or $L < 0.15$ or $L > 0.9$	grey	double bass
$H = [0^\circ; 15^\circ] \cup [325^\circ; 360^\circ]$	red	oboe
$H =]15^\circ; 70^\circ]$	yellow	violin (pizzicato)
$H =]70^\circ; 155^\circ]$	green	flute
$H =]155^\circ; 210^\circ]$	cyan	trumpet
$H =]210^\circ; 255^\circ]$	blue	piano
$H =]255^\circ; 325^\circ]$	magenta	saxophone

- The original database was limited to a certain amount of instruments.
- The recording quality was not good enough for some of the instruments.
- Some instruments were too close to be distinguished from one another.

Table 2 shows the choice we made for the colour to instrument mapping, and the condition for each mapping. The grey colour's condition is prior to any other one, because for some values of saturation and lightness (presented in this table), the hue is not important anymore, the visual feeling remains grey. Note that the values were chosen empirically, as for the choice of instruments. In fact, the colour to instrument mapping was made, imagining the feeling effect of colour and instrument. For example, the green reminds nature, and then, singing birds, which leads to flute. Cyan is a powerful colour, like the sound of the trumpet. In jazz, you can hear about the blue note, and piano is one of the most known instruments in jazz music. Experiments done during the last months showed that this approach should be reconsidered.

With this sonification system defined, we can now talk about the technical integration into the game. The sonifier is an independent server, which is waiting for the haptic device to send the colour to map. This has been implemented in C/C++ through a socket server, which works as follows.

- Launch server
- While not the end of the game do
 - Server listens to client
 - If Client connection then
 - string ← buffer
 - if string[0] = '1' then
 - R ← rrr part of string
 - G ← ggg part of string
 - B ← bbb part of string
 - hsl ← rgb2hsl(R,G,B)

- sound \leftarrow sonify(hsl)
- play(sound)

The pixel string send by the haptic device part is normalized according to this code: “n;rrr;ggg;bbb”. The first value, n, is 0 or 1, to say if this value has to be sonified or not. The rrr, ggg, and bbb are the red, green, and blue values respectively, coded on three characters, and are filled with zeros if necessary.

IV. HAPTIC INTERACTION

Haptic rendering is performed at every time step of the haptic loop using the extensively used spring dumper model. The force feedback calculation is performed using directly the GHOST SDK [15], [16] library for PHANToMTM device. PHANToM desktop has 6 DOF for input (provides position and orientation) and 3 DOF for output (provides force feedback along the three axes). In particular, the force fed onto the haptic device is evaluated through the following formula:

$$F = k_s d - k_d v$$

where k_s , k_d are the spring and dumping coefficients and d, v the penetrating distance of the haptic probe into the grooved line map and its velocity.

In order to provide realistic force feedback it is important to ensure that force feedback loop runs at frequency equal or higher than 1KHz. As a result simplified models of the 3D visual objects are used for the collision detection and the calculation of force feedback in the system.

V. SIGN LANGUAGE RECOGNITION AND SYNTHESIS

A. Sign Language Recognition

Sign language recognition used in the application has been developed in cooperation with Project 3 within the eNTERFACE'06 workshop and was integrated to the system.

Figure 3 illustrates the steps in sign recognition. The first step in hand gesture recognition is to detect and track both hands. This is a complex task because the hands may occlude each other and also come in front of other skin colored regions, such as the arms and the face. To make the detection problem easier, we have used differently colored gloves worn on two hands (see Figure 4).

Once the hands are detected, a complete hand gesture recognition system must be able to extract the hand shape, and the hand motion. We have extracted simple hand shape features and combined them with hand motion and position information to obtain a combined feature vector [17].

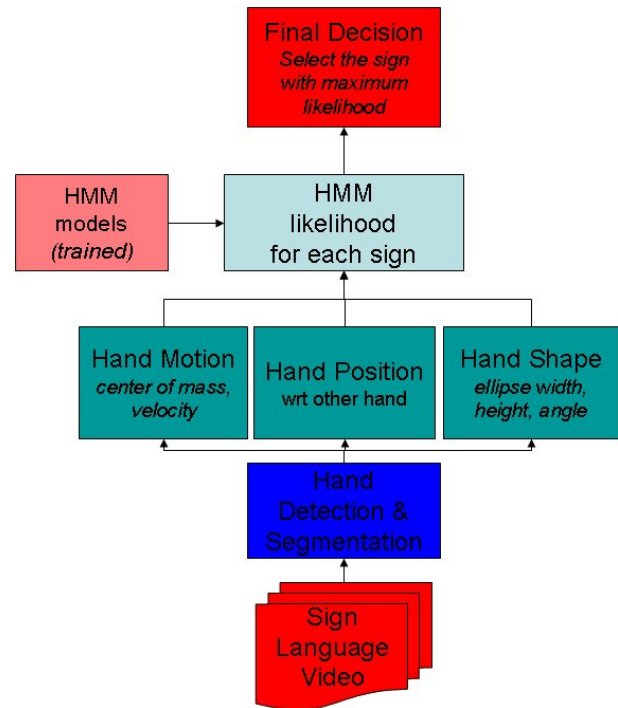


Figure 3. Sign language recognition system block diagram

Our sign database consists of five signs from ASL: *map*, *exit*, *start*, *town*, and *cave*. For each sign, we recorded 15 repetitions from two subjects. The video resolution is 640*480 pixels and the frame rate is 25 fps.

A left-to-right continuous HMM model with no state skips is trained for each sign in the database. For the final decision, likelihoods of HMM for each sign class are calculated and the sign class with the maximum likelihood is selected as the base decision.



Figure 4. The user wears colored gloves

B. Sign Language Synthesis

The system [18] utilizes H-ANIM models to provide the animation and creates the animations using as input Sign Writing Markup Language (SWML). The avatar used was provided by EPFL [19].

Currently, symbols from the 1995 version of the Sign Symbol Sequence (SSS-1995) are supported. This sequence comprises an "alphabet" of the SignWriting notation system, while true images (in gif format) of each symbol contained in this sequence are available in [20].

The input for the sign synthesis system consists of the SWML entries of the sign boxes to be visualized. For each sign box, the associated information corresponding to its symbols is parsed.

The proposed technique first converts all individual symbols found in each sign box to sequences of MPEG-4 Face and Body Animation Parameters. The resulting sequences are used to animate a H-anim-compliant VRML avatar using MPEG-4 SNHC BAP and FAP players, provided by EPFL. The system is able to convert all hand symbols as well as the associated movement, contact and movement dynamics symbols contained in any ASL sign-box. Manual (hand) gestures and facial animations are currently supported. The proposed technique has significant advantages:

- Allows almost real-time visualization of sign language notation, thus enabling interactive applications,
- Avatars can easily be included in any virtual environment created using VRML, which is useful for a number of envisaged applications, such as TV newscasts, automatic translation systems for the deaf, etc.

1) Generation of BAP key-frames

The shape number field of movement description symbols, which indicates the symbol shape, indicates the type of movement. First, the total number of key-frames to be produced is determined, based on the number and nature of the available movement, movement dynamics, contact, and synchronization symbols. More specifically, a look-up table is used to define an initial number k of key frames for each movement symbol. Furthermore, the fill parameter specifies whether the motion is slow, normal or fast. In addition, some symbols explicitly specify the movement duration. For this reason, a classification of such symbols into three categories has been defined and a different duration value D is defined for each category:

- Slow motion ($D=3$)
- Normal motion ($D=2$)
- Fast motion ($D=1$)

Synchronization (Movement Dynamics) symbols (180,181 and 182) are handled in a similar way as movement symbols. An exception is the "Un-even alternating" symbol, where first one hand moves, while the other hand is still and then the opposite. Thus, in this case, the total number of key frames is doubled ($N=2kDP$), since kDP frames are generated for the first hand, while the second hand remains still then and vice versa.

After the generation of the key-frames related to the available hand, contact, movement and synchronization symbols, it is checked whether more palm postures for the right or both of the hands exist. If there are more than one

palm symbols for one or both hands, additional key-frame(s) are generated containing the values of the BAPs, which represent the final palm position(s).

2) Use of Inverse Kinematics

Our technique is applied only to rotational joints, whose configuration is specified by one or more scalar values, describing the angle values (degrees of freedom) of a rotational joint. The complete configuration of the multibody is specified by n unknown scalars $\theta_1, \dots, \theta_n$ (joint angles) describing the configuration of all joints ([21]). The positions of the k end effectors are denoted by a vector $s=(s_1, \dots, s_k)$. The (desired) target positions are also expressed by a vector $t=(t_1, \dots, t_k)T$, where t_i is the target position for the i th end effector, and $e_i = t_i - s_i$ is the corresponding error. If $\theta = (\theta_1, \dots, \theta_n)$, T is the column vector of the unknown joint angles, the position of the j th end effector is given by a function $s_j(\theta)$,

$1 \leq j \leq k$ of the joint angles. In vector notation, this can be expressed as $s=s(\theta)$, where $s_i = s_i(\theta)$. According to the IK problem we seek values for the θ_j 's such that $t_i = s_i(\theta)$, for all i . These equations can be solved by iterative local search based on the $m \times n$ Jacobean matrix J whose elements are defined by: $J_{i,j} = \frac{\partial s_i}{\partial \theta_j}$. According to that iterative method,

the current values of θ , s and t are used for the computation of a value $\Delta\theta$ and the incrementing of the joint angles θ by $\Delta\theta$. Since $s' = J(\theta)\theta'$, the resulting change in end effector positions can be estimated as $\Delta s \approx J\Delta\theta$. The angle update may be performed either once per frame so that the end effectors only approximately follow the target positions, or iteratively until the end effectors are sufficiently close to the targets.

The entries in the Jacobean matrix are usually easy to calculate. If p_j is the position of the joint, v_j is a unit vector pointing along the current axis of rotation for the joint, and the i th end effector is affected by the joint, then the corresponding

entry in the Jacobean is $\frac{\partial s_i}{\partial \theta_j} = v_j \times (s_i - p_j)$, where angles are

measured in radians with the direction of rotation given by the right rule. If the i th end effector is not affected by the j th joint, then $\frac{\partial s_i}{\partial \theta_j} = 0$. The update value $\Delta\theta$ is computed using the

Selectively Damped Least Squares (SDLS) method ([21]), where the damping constants depend not only on the current configuration of the articulated multibody, but also on the relative positions of the end effector and the target position as well as on the difficulty of reaching the target rather than just the distance to the target.

The generation of the FAP frame sequence is performed after the generation of the BAP frame sequence, so that the total number of generated FAP frames is exactly the same as the total number of BAP frames. For each sign-box, the FAP

key-frames are determined, based on the existing facial expression/animation symbols, from predefined lookup tables for each symbol. The number of FAP key-frames, $N_{FAP_keyframes}$, is generally much smaller than the total number of BAP frames N_{BAP} that have been already generated using the procedures described in the previous Subsections. If $FAP(k), k = 0, \dots, N_{BAP} - 1$ denotes the vector of FAPs corresponding to frame k , the FAP keyframes are first positioned every $s = N_{BAP} / (N_{FAP_keyframes} - 1)$ frames:

$$FAP(i \cdot s) = FAP_keyframe(i), i = 0, \dots, N_{FAP_keyframes} - 1$$

Then, each of the remaining FAP frames is determined using linear interpolation between the two closest available FAP keyframes.

VI. GROOVED LINE MAP GENERATION

In the context of the treasure hunting game, a tool for generating grooved line maps out of interactively sketched 2D drawings is developed. A grooved line map is a 3D terrain that is grooved in specific areas that represent streets or other meaningful areas that the blind user is able to perceive through a haptic device. Recently, a system for converting conventional 2D maps to haptic representations for the blind has been developed by the ITI-CERTH team [22].

The method presented in [22] has been extended so as to convert drawings that are interactively sketched by the user into haptic representations. Figure 5 illustrates the sketched image, while Figure 6 depicts the 3D grooved line map. Since haptic rendering is a very sensitive process and demands in every time step to perform the computationally intensive procedure of collision detection that performs slower for larger 3D meshes, the grooved line map is further processed so as to generate a multiresolutional grooved line map as illustrated in Figure 7. It is obvious that this map is more detailed in the areas close to the path thus reducing the redundant complexity of the initial 3D map.

VII. HEIGHT FIELD BASED PARTIAL MATCHING

Another important aspect of developed framework is its partial matching utility using height fields. In particular, the user is initially drawing with the one hand 3D gestures that correspond to the shape of a 3D object. A stereo camera captures the gestures and generates a 3D point cloud.

The point cloud is subsequently filtered so as to remove the 3D visual tracking noise. To achieve that, a low-pass filter is used, that erases points with high value of deviation after checking their distribution in space.

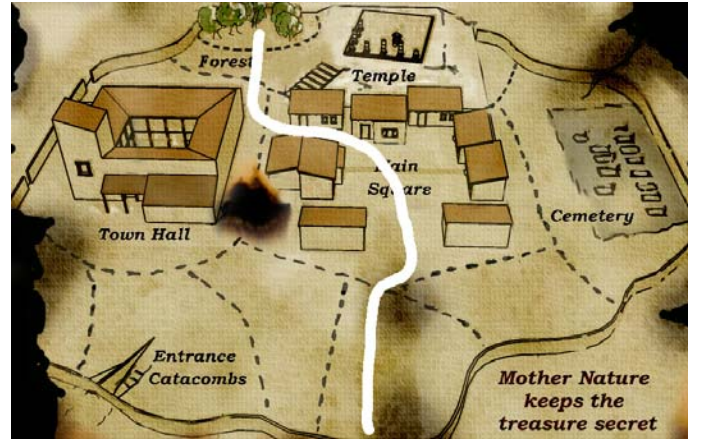


Figure 5. Example of sketched image map

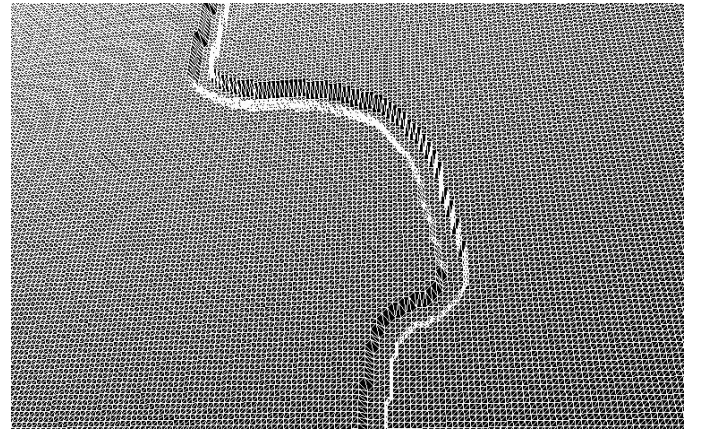


Figure 6. The 3D grooved line map.

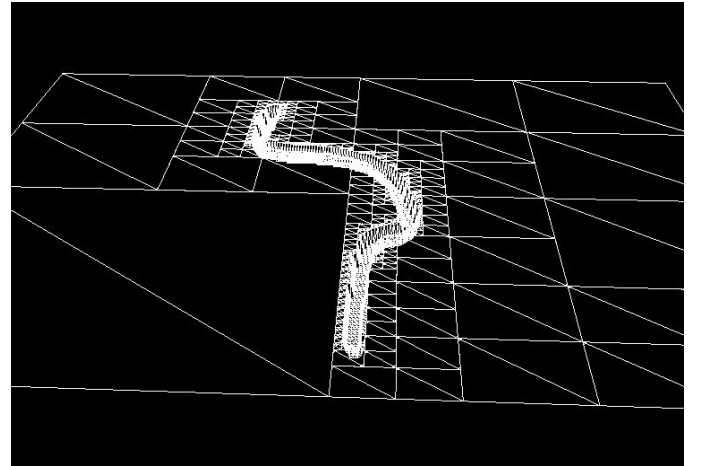


Figure 7. 3D grooved map after polygon reduction

A. Creating Height Fields

Height fields are fast, efficient structures that are generally used to create terrains or other raised surfaces out of hundreds of triangles in a mesh. A height field is, as implied, a 2D array that stores in its entries values about the height of the specific point in the 3D structure [23]. Graphics applications usually create a polygon mesh so as to render a height field [24], [25].

The values of the height field are normalized in the interval $[0,1]$. In the context of the present framework, we initially generate a height field that corresponds to the entire scene. During runtime, the user sketches a 3D point cloud that is also converted to a height field. The aim of the system is to identify in the scene the object that the user has drawn.

The first step is to generate the height field from a triangulated surface. In the case of the sketched point cloud, Delaunay triangulation [26], [27] is used in order to convert the point cloud to a 3D mesh. It is obvious that from a 3D triangulated mesh direct information about the shape's height is obtained only for the vertices of the mesh. Thus, in order to generate a dense height field with height values at every entry of the height map, proper interpolation procedures have to be implemented. Moreover, the generated dense height field is invariant on the sampling resolutions of the scene and the sketched object.

The dense height field is generated by performing interpolation on the triangulated mesh that it models. In the beginning, the user can determine the density of the resulting height field that influences its quality. The height field of the complete 3D scene of the game is shown in Figure 8. Also, the height field for the point cloud is shown in Figure 9. Brighter values correspond to higher areas in the scene.

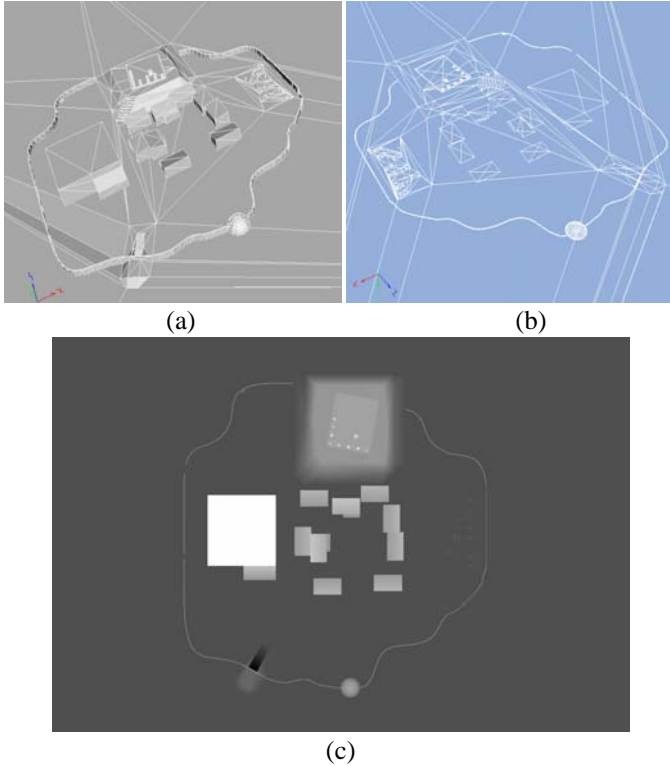


Figure 8: (a) The initial 3D scene, (b) 2D mapping of the 3D scene, (c) The produced height field.

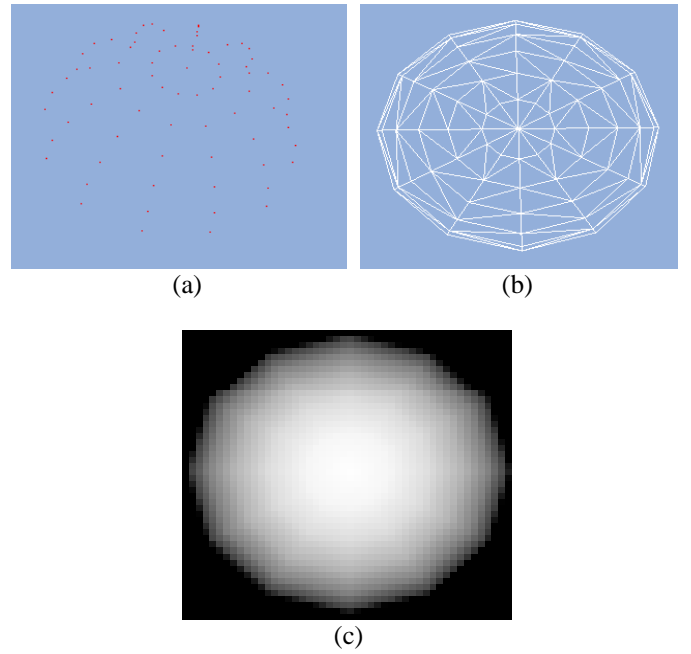


Figure 9: (a) The initial point cloud, (b) Triangulated 2D map of the point cloud, (c) The produced height field,.

A height field is in general an efficient way to describe a terrain or a landscape. Moreover, as a 2D structure, it can be processed much faster and state-of-the-art image processing techniques can be applied so as to obtain, e.g. shape descriptors, local features, etc.

On contrary, it features a serious limitation. In particular, in order to generate accurately a height field out of a 3D mesh, the mapping of the mesh onto the 2D height field surface has to be injective (one-to-one), i.e. each entry of the height field should correspond only to one element of the 3D mesh. This restriction inhibits the modeling of folded and other complex structures with height fields. Although, structures for modeling those kinds of models exist, e.g. vectorial height fields, their processing becomes more complex.

B. Height Field Matching

As mentioned before, the values of the entries of a normalized height field range in the interval $[0,1]$. When trying to match a height field with only one part of another, then a Z-scaling (scaling in the Z dimension) issue obviously arises. In the context of the present framework, this problem is dealt with by renormalizing only the part of the height field that is currently processed. As a result, assuming that the correct 2D part of the height field is chosen, the renormalization would solve a possible scaling mismatch of the source and target height fields.

The matching process proceeds as follows: The source height field is convolved with the target height field by moving the source height field window across the target height field. Notice that the size of the window is variable so as to also consider the 2D scaling mismatch. Moreover, correlation of the input height field and the target height field

part is also considered for different orientations [28], [29] so as to create a rotation invariant matching scheme. The correlation is calculated using the following equation:

$$C(x_0, y_0, a_k, \theta_l) = \sum_{i=x_0}^{x_0+a_k N} \sum_{j=y_0}^{y_0+a_k N} (\mathbf{H}_o(i, j) - \mathbf{H}_s(i - x_0, j - y_0, \theta_l))^2$$

$$\forall (x_0, y_0),$$

$$\forall a_k \in \{a_i, a_i + a_s, a_i + 2a_s, \dots, a_i\},$$

$$\forall \theta_l, \quad \theta_l = \frac{2\pi k}{N_r} \quad \text{for } k \in \{0, 1, \dots, N_r - 1\}$$

where (x_0, y_0) refers to the current processing position in the scene height field, a_k is the 2D scaling factor, N is the default size of the sketched height field, a_i and a_s the marginal values of the scaling factor, a_s the scaling space search step, θ_l the current rotation angle and N_r the rotation space resolution. Functions \mathbf{H}_o and \mathbf{H}_s correspond to the scene height field and the sketched object's height field respectively.

Finally, the algorithm outputs a variable vector that maximizes the correlation between the source height field and a target area in the scene height field. The variables in the variable vector refer to X position, Y position, 2D scaling and rotation angle.

VIII. APPLICATION SCENARIO

The aforementioned technologies were integrated in order to create an entertainment scenario. The Scenario consists of seven steps. In each step one of the users has to perform one or more actions in order to pass successfully to the next step.

The storyboard is about an ancient city that is under attack and citizens of the city try finding the designs in order to create high technology war machines.

A. 1st Step



Figure 10. House with the red closet. The white sphere corresponds to the position of the Phantom probe

The first step involves the blind user. The user receives

audio message informing him/her to find a red closed. The user starts from the initiating point at the entrance of the village and using the haptic device explores the village in order to find in one of the houses a red closet. In this step the blind user has to use the haptic device in order to explore the 3D Workspace.

Furthermore audio modality replaces color modality, using SeeCoLo module, and allows the blind user select the correct closet and thus receive an audio message. The audio message is then sent to the second step of the application.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input*: Audio message “Find the red closet”
- *Action*: Search for the red closet in one of the city houses
- *Output*: Audio message “Town hall”
- *Modality replacement*: Color is converted into sound using the SeeColor utility

B. 2nd Step

The second step involves the deaf and mute user. The user receives audio message. The message is converted from audio to text using the speech recognition tool and then to sign language using the sign synthesis tool. The user finally receives the message as a gesture through a 3D interactive avatar.

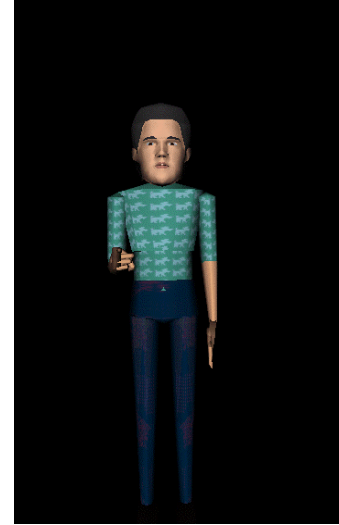


Figure 11. The 3D avatar is performing a sign language phrase.

The message guides the blind user to the town hall of the city where the mayor (Figure 12) of the city assigns them a task.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input*: Sign language synthesized phrase “Go to the town hall”
- *Action*: Go to the town hall and talk to the king
- *Output*: Audio message “Go to the temple ruins”
- *Modality replacement*: The input audio message is recognized and converted to text, which is finally

converted to the corresponding sign language phrase.



Figure 12. Town Hall.

C. 3^d Step

The third step involves the blind user, who hears the message said by the Mayor and goes to the temple ruins. In the temple ruins the blind user has to search for an object that has an inscription written on it.



Figure 13. The temple ruins and the inscription.

One of the columns in the destroyed temple has an inscription written on it that states, “The dead will save the city”. The blind user is informed by an audio message whenever he finds this column and the message is sent to the deaf-mute user’s terminal.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input:* Audio message “Go to the temple ruins”
- *Action:* Go to the temple ruins and find oracle inscription
- *Output:* Inscription (text) “The dead will save the city”

D. 4th Step

The fourth step involves again the deaf and mute user. The user receives the written text in sign language form. The text modality is translated to sign language symbols using the sign synthesis tool. Then the deaf and mute user has to understand the meaning of the inscription “The dead will save the city” and go to the cemetery using the mouse.



Figure 14. The cemetery scene.

There he/she should search for a key that lies in one of the graves. The word “Catacombs” is written on the key. The deaf and mute user has to perform a sign in sign language in order to allow the blind user understand that the key opens a box in the catacombs. The deaf user has to perform the sign “Cave”. This sign is recognized by the sign language recognition tool and the text result is sent to the next step of the scenario.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input:* Sign language synthesized phrase “The dead will save the city”
- *Action:* Go to the cemetery and find the key
- *Output:* Key labeled “Catacombs”
- *Modality replacement:* The input inscription text is converted to sign language and the deaf-mute user sketches the word “cave” that is recognized by the system and sent to the blind user’s terminal.

E. 5th Step

In this step the blind user receives the text, which is converted to audio using the text to speech tool. This step involves haptic and audio information. The user has to search for the catacombs enter in them and find the box that contains a map (Figure 15). The map is then sent to the next level.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input:* Audio message “Cave”
- *Action:* Go to the cave (catacombs) and search for a hidden map.
- *Output:* Map

- *Modality replacement*: Text is transformed to synthesized speech.



Figure 15. In the catacombs.

F. 6th Step

The deaf user receives the map, and has to draw the route to the area where the treasure is hidden (Figure 5). The route is drawn on the map and the map is converted to a grooved line map, which is send to for the last level to the blind user.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input*: Map with riddle (Figure 5)
- *Action*: Solve the riddle and sketch the path to the treasure area.
- *Output*: 2D sketch on the map
- *Modality replacement*: Visual information (sketched map) is transformed into haptic representation (grooved line map).

G. 7th Step

The blind user receives the grooved line map and has to find and follow the way to the forest where the treasure is hidden. Although the map is presented again as a 2D image the blind user can feel the 3D grooved map and follow the route to the forest. The 2D image and the 3D map are registered and this allows us to visualize the route that the blind user actually follows on the 2D image. The blind user is asked to press the key of the PHANToM device while he believes that the PHANTOM cursor lies in the path. Finally, after finding the forest he obtains a new grooved line map (Figure 16) where the blind user has to search for the final location of the treasure.

The input-output of this step as well as actions that should be performed are summarized in the following:

- *Input*: Two grooved line maps
- *Action*: Find the forest following the first grooved line map and finally explore the second grooved line map and

find the treasure.

- *Output*: Treasure



Figure 16. The forest grooved line map.

After searching in the forest streets the blind user should find the treasure that is illustrated in Figure 17.

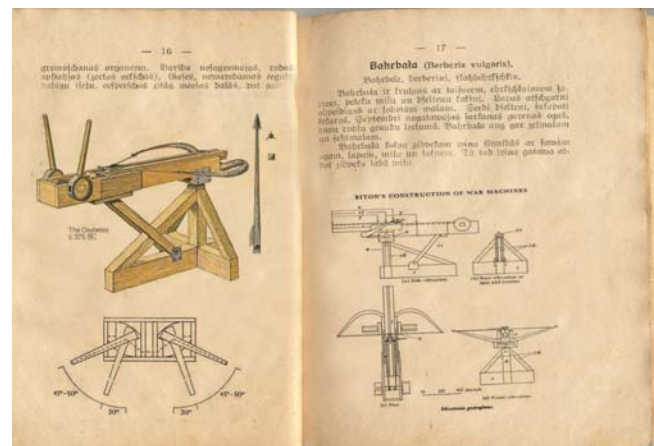


Figure 17. The treasure book.

IX. CONCLUSIONS

The initials tests performed within eINTERFACE have shown that the application is user friendly and is integrated into a feasible for the users scenario. Moreover, the tests the two users are asked to perform have various levels of difficulty.

The proposed system is actually the first attempt of generating a system for the intercommunication of blind and deaf-mute users. Although in the current version the users are somehow limited in their intercommunications, this initial implementation has shown that with proper user-centered design, the development of such a system is feasible.

Further a structured usability evaluation of the system involving both visually and hearing impaired users is a necessary in order to identify the weaknesses of the proposed

methodologies for the intercommunication between the blind and deaf mute users.

ACKNOWLEDGEMENTS:

We would like to thank the group 3 working on '**Sign Language Tutoring Tool**' for providing software, infrastructure and effort to integrate the sign language recognition tool to the developed system. This work was supported by the EU funded **SIMILAR** Network of Excellence.

REFERENCES

- [1] W3C Workshop on Multimodal Interaction, 19/20 July, 2004, Sophia Antipolis, France (<http://www.w3.org/2004/02/mmi-workshop-cfp.html>)
- [2] Special Issue: Interacting with emerging technologies, J. Strickon, Guest Ed., IEEE Computer Graphics and Applications, Jan-Feb 2004.
- [3] I. Marsic, A. Medl and J. Flanagan, "Natural communication with information systems", Proc. of the IEEE, pp. 1354-1366, vol. 88, no.8, August 2000.
- [4] J. Lumsden and S. A. Brewster, "A paradigm shift: Alternative interaction techniques for use with mobile & wearable devices", Proc. 13th Annual IBM Centers for Advanced Studies Conference CASCON'2003, pp. 97-100, Toronto, Canada, 2003.
- [5] T. V. Raman, Multimodal Interaction Design Principles For Multimodal Interaction, CHI 2003, pp. 5-10, Fort Lauderdale, USA, 2003.
- [6] A. Camurri and G. Volpe (Eds.), "Gesture-based Communication in Human-Computer Interaction", Lecture Notes in Artificial Intelligence, no. 2915, Springer Verlag, February 2004.
- [7] C. Colwell, H. Petrie, D. Kornbrot, A. Hardwick, and S. Furner, "Haptic Virtual Reality for Blind Computer Users", in Proc. of Annual ACM Conference on Assistive Technologies (ASSETS '98), pp 92-99, 1998.
- [8] C. Sjostrom, "Touch Access for People With Disabilities", Licentiate Thesis, in CERTEC Lund University, Sweden, 1999.
- [9] V. Scoy, I. Kawai, S. Darrah, F. Rash, "Haptic Display of Mathematical Functions for Teaching Mathematics to Students with Vision Disabilities", Haptic Human-Computer Interaction Workshop, 2000.
- [10] C. Capelle, C. Trullemans, P. Arno, and C. Veraart., "A real-time experimental prototype for enhancement of visionrehabilitation using auditory substitution", IEEE Transactions on Biomedical Engineering, 40(10), 1998.
- [11] G. Iannizzotto, C. Costanzo, P. Lanzafame, and F. La Rosa, "Badge3d for visually impaired", In CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops, pages 29-36, 2005.
- [12] Patrick Roth, "Représentation multimodale d'images digitales dans des systèmes informatiques multimédias pour utilisateurs non-voyants", PhD Thesis, Geneva, Switzerland, 2002.
- [13] L. Kay, "A sonar aid to enhance spatial perception of the blind: Engineering design and evaluation." The Radio and Electronic Engineer, 44, pages 605-627, 1974.
- [14] P.B.L. Meijer, "An experimental system for auditory image representations", Transactions on Biomedical Engineering, 39(2), pages 112-121, 1992.
- [15] Sensable Technologies Inc, "PHANToM™ Haptic Device", http://www.sensable.com/products/phantom_ghost/phantom.asp.GHOST
- [16] T. Massie and K. Salisbury, "The PHANToM Haptic Interface: A Device for Probing Virtual Objects", ASME Winter Annual Meeting, DSC-Vol. 55-1, ASME, New York, pp. 295-300, 1994.
- [17] O. Aran, L. Akarun "Recognizing two handed gestures with generative, discriminative and ensemble methods via Fisher kernels", International Workshop on Multimedia Content Representation, Classification and Security, (MRCS'06), Istanbul, September 2006.
- [18] M. Papadogiorgaki, N. Grammalidis, L. Makris, and M. G. Strintzis, "Gesture synthesis from sign language notation using MPEG-4 humanoid animation parameters and inverse kinematics", 2nd International Conference on Intelligent Environments (IE06), July 5-6, 2006, Athens, Greece
- [19] SNHC. ISO/IEC JTC1/SC29/WG11 N2802, <http://coven.lancs.ac.uk/mpeg4/>
- [20] Official Site of SWML, <http://swml.ucpel.tche.br/>
- [21] S.R. Buss (2004), 'Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods', University of California, San Diego, Typeset manuscript, available from <http://math.ucsd.edu/~sbuss/ResearchWeb>.
- [22] K. Moustakas, G. Nikolakis, K. Kostopoulos, D. Tzovaras and M.G. Strintzis, "The Force Field Haptic Rendering Method: Application in the Haptic Access to Visual Data for the Training of the Visually Impaired", IEEE Multimedia Magazine, accepted for publication.
- [23] Fabio Policarpo, Manuel M. Oliveira, "Relief Mapping of Non-Height-Field Surface Details".
- [24] J. Peng, D. Kristjansson and D. Zorin, "Interactive modeling of topologically complex geometric detail". ACM Transaction of Graphics - Proceedings of SIGGRAPH 2004 23, 3 (August), 635–643.
- [25] S. Pombrescu, B. Budge, L. Feng and K. Joy, "Shell maps", ACM Transaction of Graphics - Proceedings of SIGGRAPH2005 24, 3 (July), 626–633, 2005.
- [26] B. Delaunay, Sur la sphère vide, Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk, 7:793-800, 1934.
- [27] P.Cignoni, C.Montani, R.Perego and R.Scopigno. Parallel 3D Delaunay Triangulation. Eurographics 93: 129-142, 1993.
- [28] F. Policarpo, M.M. Oliveira and J. Comba, "Realtime relief mapping on arbitrary polygonal surfaces", In Proceedings of ACM Symposium on Interactive 3D Graphics and Games 2005, ACM Press, 155–162.
- [29] X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H.Y. Shum, "Generalized displacement maps", In Eurographics Symposium on Rendering 2004, 227–233.

Sign Language Tutoring Tool

Oya Aran¹, Ismail Ari¹, Alexandre Benoit², Ana Huerta Carrillo³, François-Xavier Fanard⁴,
Pavel Campr⁵, Lale Akarun¹, Alice Caplier², Michele Rombaut² and Bulent Sankur¹

¹Bogazici University, ²LIS_INPG, ³Technical University of Madrid, ⁴Universite Catholique de Louvain, ⁵ University of West Bohemia in Pilsen

Abstract—In this project, we have developed a sign language tutor that lets users learn isolated signs by watching recorded videos and by trying the same signs. The system records the user's video and analyses it. If the sign is recognized, both verbal and animated feedback is given to the user. The system is able to recognize complex signs that involve both hand gestures and head movements and expressions. Our performance tests yield a 99% recognition rate on signs involving only manual gestures and 85% recognition rate on signs that involve both manual and non manual components, such as head movement and facial expressions.

Index Terms—Gesture recognition, sign language recognition, head movement analysis, human body animation

I. INTRODUCTION

THE purpose of this project is to develop a Sign Language Tutoring Demonstrator that lets users practice demonstrated signs and get feedback about their performance. In a learning step, a video of a specific sign is demonstrated to the user and in the practice step, the user is asked to repeat the sign. An evaluation of produced gesture is given to the learner; together with a synthesized version of the sign that lets the user get visual feedback in a caricatured form.

The specificity of Sign Language is that the whole message is contained not only in hand gestures and shapes (manual signs) but also in facial expressions and head/shoulder motion (non-manual signs). As a consequence, the language is intrinsically multimodal. In order to solve the hand trajectory recognition problem, Hidden Markov Models have been used extensively for the last decade. Lee and Kim [1] propose a method for online gesture spotting using HMMs. Starner et al. [2] used HMMs for continuous American Sign Language recognition. The vocabulary contains 40 signs and the sentence structure to be recognized was constrained to personal pronoun, verb, noun, and adjective. In 1997, Vogler and Metaxas [3] proposed a system for both isolated and continuous ASL recognition sentences with a 53-sign vocabulary. In a later study [4] the same authors attacked the scalability problem and proposed a method for the parallel modeling of the phonemes within an HMM framework. Most

systems of Sign Language recognition concentrate on hand gesture analysis only. In , a survey on automatic sign language analysis is given and integrating non-manual signs with hand gestures is examined.

A preliminary version of the tutor we propose to develop, demonstrated at EUSIPCO, uses only hand trajectory based gesture recognition [6]. The signs selected were signs that could be recognized based on solely the trajectory of one hand. In this project, we aim at developing a tutoring system able to cope with two sources of information: hand gestures and head motion. The database contains complex signs that are performed with two hands and head gestures. Therefore, our Sign Language Recognition system fuses the data coming from two sources of information to recognize a performed sign: The shape and trajectory of the two hands and the head movements.

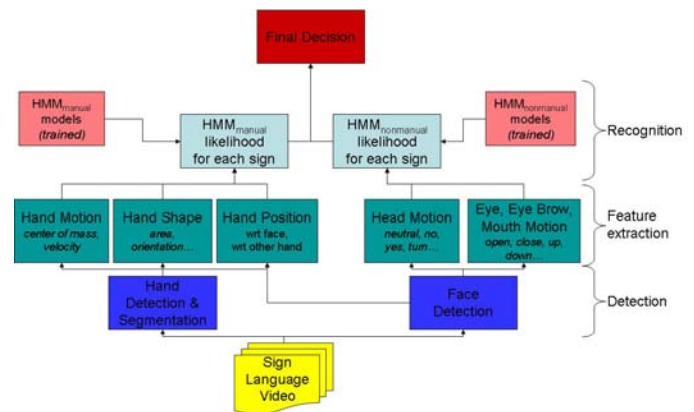


Fig. 1. Sign language recognition system block diagram

Fig. 1 illustrates the steps in sign recognition. The first step in hand gesture recognition is to detect and track both hands. This is a complex task because the hands may occlude each other and also overlap other skin colored regions, such as the arms and the face. To make the detection problem easier, markers on the hand and fingers are widely used in the literature. In this project, we have used differently colored gloves worn on the two hands. Once the hands are detected, a complete hand gesture recognition system must be able to extract the hand shape, and the hand motion. We have extracted simple hand shape features and combined them with hand motion and position information to obtain a combined feature vector. A left-to-right continuous HMM model with no

state skips is trained for each sign. These HMM models could be directly used for recognition if we were to recognize only the manual signs. However, some signs involve non-manual components. Thus further analysis of head movements and facial expressions must be done to recognize non-manual signs.

Head movement analysis works concurrently with hand gesture analysis. Following the face detection step, a method based on the human visual system is used to calculate the motion energy and the velocity of the head, eye, eyebrows and mouth. These features are combined into a single feature vector and HMM models for the non-manual signs are trained.

For the final decision, manual and non-manual HMM models are fused in a sequential manner. Decisions of the manual HMMs are used as the base for decision and non-manual HMMs take part to differentiate between the variants of the base sign.

Another new feature of the Sign Language Tutoring tool is that it uses synthesized head and arm motions based on the analysis of arm and head movements. This lets the user get accentuated feedback. Feedback, either TRUE or FALSE, is given for the manual component as well as for the non-manual one, separately.

In this project, we have first defined a limited number of signs that can be used for sign language tutoring. 19 signs have been selected so that head motions are crucial for their recognition: Some signs have identical hand motions but different head motions. After defining the dataset, we have collected data from eight subjects. Each subject performed all the signs five times.

The sign language tutor application was designed to show selected signs to the user and to let the user record his/her own sign using the webcam connected to the system. The application then runs the analysis, recognition, and synthesis subsystems. The recognized sign is identified by a text message and the synthesized animation is shown as feedback to the user. If the sign is not performed correctly, the user may repeat the test.

This report is organized as follows: In section II, we give details of the sign language tutor application, together with database details. Section III details the analysis: hand segmentation, hand motion feature extraction, hand shape feature extraction, and head motion feature extraction. Section IV describes the recognition by fusion of information from all sources. Section V describes the synthesis of head motion, facial expressions, hands and arms motion. Section VI gives results of the recognition tests and Section VII concludes the report and outlines future directions.

II. SIGN LANGUAGE TUTOR

A. Sign Language

The linguistic characteristics of sign language is different than that of spoken languages due to the existence of several components affecting the context such as the use of facial

expressions and the head movements in addition to the hand movements. The structure of spoken language makes use of words linearly i.e., one after another, whereas sign language makes use of several body movements in parallel in a completely different spatial and temporal sequence.

Language modeling enables to improve the performance of speech recognition systems. A language model for sign language is also required for the same purpose. Besides, the significance of co-articulation effects necessitates the continuous recognition of sign language instead of the recognition of isolated signs. These are complex problems to be tackled. For the present, we have focused on recognition of isolated words, or phrases, that involve manual and non-manual components.

There are many sign languages in the world. We have chosen signs from American Sign Language (ASL), since ASL is widely studied. However, our system is quite general and can be adapted to others.

B. Database

For our database, 19 signs from American Sign Language were used. The selected signs include non-manual signs and inflections in the signing of the same manual sign [5]. For each sign, we recorded five repetitions from eight subjects. The preferred video resolution was 640*480 pixels and the frame rate was 25 fps. Short descriptions about the signs we used in the database can be seen in TABLE I.

TABLE I. ASL SIGNS IN THE DATABASE

Sign	Head / Facial Expression	Hand
[smbdy] is here	Nod	Circular motion parallel to the ground with right hand.
Is [smbdy] here?	Brows up, Head forward	
[smbdy] is not here	Head shake	
Clean	-	Right palm facing down, left palm facing up. Sweep left hand with right.
Very clean	Lips closed, head turns from right to frontt, sharp motion	
Afraid	-	
Very afraid	Facial expression (lips open, eyes wide)	The same as “afraid”, but shake the hands at the middle
Fast	-	Hands start in front of body and motion towards the body. Fingers partially closed, thumb open
Very fast	Facial expression (lips open, eyes wide), and sharp motion	
To drink	Head motion (up and down)	Drinking motion, hand as holding a cup
Drink (noun)	-	Repetitive drinking motion, hand as holding a cup.

To open door	-	Palms facing to the front. One hand moves as if the door is opened; only once.
Open door (noun)	-	Palms facing to the front. One hand moves as if the door is opened. Repeat, with small hand motion
Study	-	Left hand palm facing upwards, right hand all fingers open, mainly finger motion (finger tilt)
Study continuously	Circular head motion accompanies hand motion	Left palm facing up, right hand all fingers open, finger tilt together with large and downward circular motion
Study regularly	Downward head motion accompanies hand motion	Left palm facing upwards, right hand all fingers open, downward/ upward sharp motion, no finger motion
Look at	-	Starting from the eyes, forward motion, two hands together.
Look at continuously	Circular head motion accompanies hand motion	Starting from the eyes, forward motion, two hands together. Larger and circular motion
Look at regularly	Downward head motion accompanies hand motion	Starting from the eyes, forward motion, two hands together. Sharp forward/ backward motion

C. Tutor Application

The sign language tutor application was designed to show selected signs to the user and to let the user record his/her own sign using the webcam connected to the system. The graphical user interface for the tutor can be observed in Fig. 2.



Fig. 2: Sign Language Tutoring Tool GUI

The graphical user interface consists of four panels: Training, Information, Practice and Synthesis. Training panel

involves the teacher videos, thus the user can watch the videos to learn the sign by pressing the Play button. The program captures the user's sign video after the Try button is pressed. Afterwards, information panel is used for informing the user about the results of his/her trial. There are three types of results: "ok" (the sign was confirmed), "false" (the sign was wrong) and "head is ok but hands are false". Possible errors are also shown in this field.

Users can watch the original captured video or the segmented video in this panel as shown in Fig. 3. Afterwards, if the user wants to see the synthesized video, he/she can use the synthesis panel.

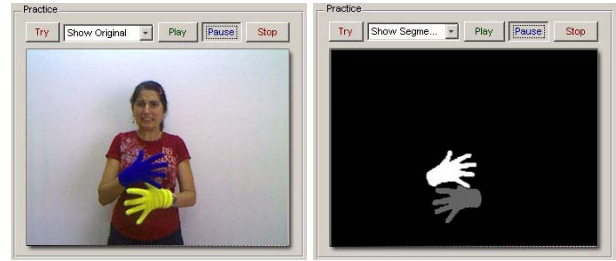


Fig. 3: A screenshot of original and segmented videos

III. SIGN LANGUAGE ANALYSIS

A. Hand segmentation

The user wears gloves with different colors when performing the signs. The two colored regions are detected and marked as separate components. Ideally, we expect an image with three components: the background, the right hand and the left hand.

For the classification, histogram approach is used as proposed in [7]. Double thresholding is used to ensure connectivity, and to avoid spikes in the binary image. We prefer HSV color space as Jayaram et al. [7] and Albiol et al. [8] propose. HSV is preferred because of its robustness to changing illumination conditions.

The scheme is composed of training the histogram and threshold values for future use. We took 135 random snapshot images from our training video database. For each snapshot, ground truth binary images were constructed for the true position of the hands. Using the ground truth images, we have constructed the histogram for the left and right hands, resulting in two different histograms. Finally, normalization is needed for each histogram such that the values lie in the interval [0,1].

The low and high threshold values for double thresholding are found in training period. When single thresholding is used, a threshold value is chosen according to the miss and false alarm rates. Since we use double thresholding, we use an iterative scheme to minimize total error. We iteratively search for the minimum total error. This search is done in the range $[\mu - \delta, \mu + \delta]$ to decrease the running time, where μ is the mean and δ is the standard deviation of the histogram.

After classification by using the scheme described above,

we observed that some confusing colors on the subject's clothing were classified as hand pixels. To avoid this, we selected the largest connected component of the classified regions into consideration. Thus we had only one component classified as hand for each color.

This classification approach can also be used for different colored gloves or skin after changing the ground truth images in the training period.

B. Hand motion analysis

The analysis of hand motion is done by tracking the center of mass (CoM) and calculating the velocity of each segmented hand. However, these hand trajectories are noisy due to noise introduced at the segmentation step. Thus, we use Kalman filters to smooth the obtained trajectories. The motion of each hand is approximated by a constant velocity motion model, in which the acceleration is neglected.

Two independent Kalman filters are used for each hand. The initialization of the Kalman Filter is done when the hand is first detected in the video. At each sequential frame, Kalman filter time update equations are calculated to predict the new hand position. The hand position found by the hand segmentation is used as measurements to correct the Kalman Filter parameters. Posterior states of each Kalman filter is defined as feature vectors for x, y coordinates of CoM and velocity. The hand can be lost due to occlusion or bad lighting in some frames. In that case, Kalman Filter prediction is directly used without correcting the Kalman Filter parameters. The hand is assumed to be out of the camera view if no hand can be detected for some number of (i.e. six) consecutive frames. Fig. 4 shows the extracted trajectories for each hand for the “fast” sign.

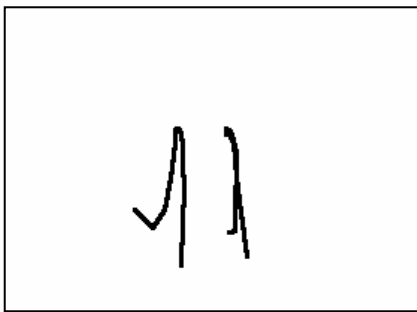


Fig. 4. Hand trajectories for sign “fast”

C. Hand shape analysis

Hand shape analysis is performed during sign recognition in order to increase the accuracy of recognition system and to differentiate between signs that differ only in hand shape. Each sign has a specific movement of the head, hands and hand postures. The extreme situation is when two signs have the same movements of head and hands and they differ only in hand postures. In this case, hand shape analysis is necessary to distinguish between them.

Another application can be in sign synthesis. If we analyze an unknown gesture and want to synthesize it with the same

movements to caricature the movements of the actor, then finger and palm movements may be synthesized by following these steps: 1) unknown hand shape is classified into one of predefined clusters, 2) hand posture synthesis of classified cluster is performed (synthesis is prepared for each cluster). This can be useful whenever it is difficult to analyze finger and palm positions directly from image, for example when only low resolution images are available. This was the case in this project – each hand shape image was smaller than 80x80 pixels.

1) Input – binary image

After the segmentation of the source image is done, two binary images (only two colours representing background and hand) of left and right hand are analyzed. The mirror reflection of the right hand is taken so we analyze both hands in the same geometry; with thumb to the right. There are several difficulties using these images:

1. Low resolution (max. 80 pixels wide in our case)
2. Segmentation errors due to blurring caused by fast movement (see Fig. 5b)
3. Two different hand postures can have the same binary image (see Fig. 5a; which can be left hand observed from top or right hand from bottom)



Fig.5. Two different hand segmentations: a. Hand shape 1; b. hand shape 2

2) Hand shape analysis – feature extraction

The binary image is converted into a set of numbers which describe hand shape, yielding the feature set. The aim is to have similar values of features for similar hand shapes and distant values for different shapes. It is also required to have scale invariant features so that images with the same hand shape but different size would have the same feature values. This is done by choosing features which are scale invariant. Our system uses only a single camera and our features do not have depth information; except for the foreshortening due to perspective. In order to keep this information about the z-coordinate (depth), five of the 19 features were not normalized. All 19 features are listed in TABLE II.

TABLE II. HAND SHAPE FEATURES

#	feature	invariant	
		scale	rotation
1	Best fitting ellipse width		✓
2	Best fitting ellipse height		✓

		<i>invariant</i>	
3	Compactness ($\text{perimeter}^2/\text{area}$)	✓	✓
4	Ratio of hand pixels outside / inside of ellipse	✓	✓
5	Ratio of hand / background pixels inside of ellipse	✓	✓
6	$\sin(2*\alpha)$ α = angle of ellipse major axis	✓	
7	$\cos(2*\alpha)$ α = angle of ellipse major axis	✓	
8	Elongation (ratio of ellipse major/minor axis length)	✓	✓
9	Percentage of NW (north-west) area filled by hand	✓	
10	Percentage of N area filled by hand	✓	
11	Percentage of NE area filled by hand	✓	
12	Percentage of E area filled by hand	✓	
13	Percentage of SE area filled by hand	✓	
14	Percentage of S area filled by hand	✓	
15	Percentage of SW area filled by hand	✓	
16	Percentage of W area filled by hand	✓	
17	Total area (pixels)		✓
18	Bounding box width		
19	Bounding box height		

An initial idea was to use “high level” knowledge about the shape such as finger count, but the problems listed previously caused us to use more low level features, which are robust to segmentation errors and work well with low resolution images.

Seven of the features (#1,2,4,5,6,7,8) are based on using the best fitting ellipse (in least-squares sense) to a binary image, as seen in Fig. 6a. The angle α is a value from 0° to 360° . However, only values from 0 to 180 are meaningful, because the ellipse has mirror symmetry. Hence only 0° to 180° interval is used. Another problem is the following: Consider 5° and 15° ellipses, which have similar angles and similar orientation. 5° and 175° ellipses have similar orientations as before, but the angles are completely different. In order to represent this difference, we use $\sin(2*\alpha)$ and $\cos(2*\alpha)$ as features.

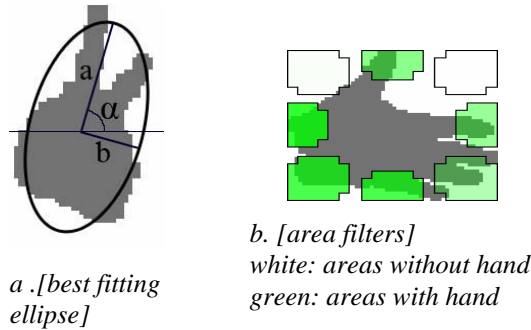


Fig. 6 a. Best fitting ellipse; b. Area filters

Features #9 to 16 are based on using “area filters”, as seen in Fig. 6a. The bounding box of the hand is divided into eight areas, in which percentage of hand pixels are calculated.

Other features in TABLE II, are perimeter, area and bounding box width and height.

3) Classification

Hand shape classification can be used for sign synthesis or to improve the recognition: The classified cluster can be used as new feature: We can use hand features for recognition only when the unknown hand shape is classified into a cluster (this means that the unknown hand shape is similar to a known one and not to a blurred shape which can have misleading features).

We have tried classification of hand shapes into 20 clusters (see Fig. 7 “clusters”). Each cluster is represented by approximately 15 templates. We use K-means algorithm ($K=4$) to classify unknown hand shape (represented by set of features described above). If the distance of unknown shape and each cluster is greater than 0.6 then this shape is declared as unclassified.

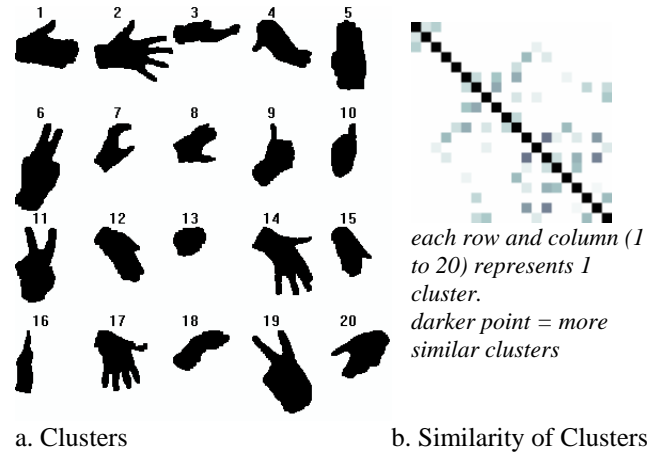


Fig. 7. a.The hand clusters; b.Similarity of clusters

As seen in Fig. 7b, some of the clusters are more similar than the others. For example, clusters 12, 14 and 19 are similar; so it is more difficult to correctly classify the unknown shape into one of these clusters.

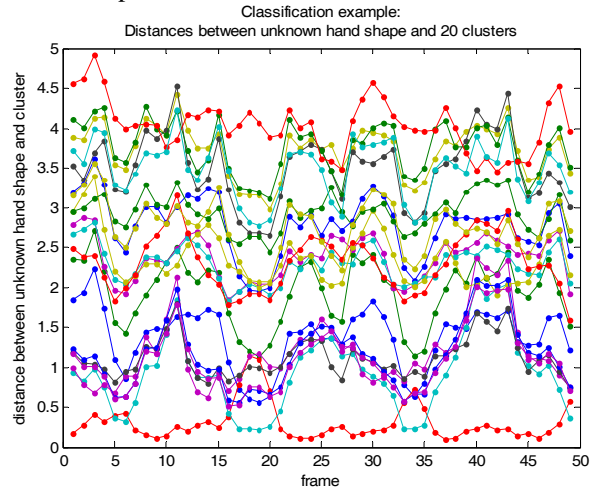


Fig. 8 Classification example: distances between an unknown hand shape and center of 20 clusters.

Classification of hand shapes is made in each frame of video sequence. It is reasonable to use information from previous frames, because hand shape cannot change so fast in each frame (1 frame = 40ms). Usually the classification is the same as in the previous frame, as seen in Fig. 8, where an unknown shape is classified into a cluster with the smallest distance.

To avoid fast variations of classifications we proposed a filter which smoothes these distances by weighted averaging Fig. 9 shows a classification example with filtering.

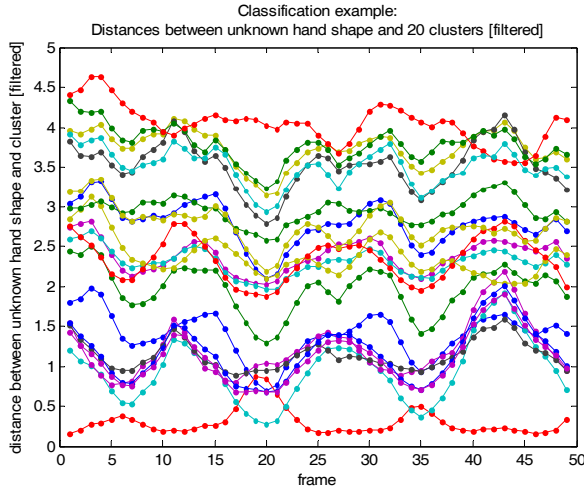


Fig. 9 Classification example: distances between an unknown hand shape and center of 20 clusters (filtered).

The new distance is calculated by the following equation:

$$D_{\text{new}}(t) = 0.34 \cdot D_{\text{old}}(t) + 0.25 \cdot D_{\text{old}}(t-1) + 0.18 \cdot D_{\text{old}}(t-2) + 0.12 \cdot D_{\text{old}}(t-3) + 0.07 \cdot D_{\text{old}}(t-4) + 0.04 \cdot D_{\text{old}}(t-5)$$

By comparing Fig. 8 and 9, one can see that this filter prevents fast changes in frames 5 and 6. This filter is designed to work in real-time applications. If used in offline application, it can easily be changed to use information from the future to increase the accuracy.

D. Head motion analysis

1) General Overview of the system

Once a bounding box around the sign language student's face has been detected, rigid head motions such as head rotations and head nods are detected by using an algorithm working in a way close to the human visual system. In a first step, a filter inspired by the modeling of the human retina is applied. This filter enhances moving contours and cancels static ones. In a second step, the fast fourier transform (FFT) of the filtered image is computed in the log polar domain as a model of the primary visual cortex (V1). This step allows extracting two types of features: the quantity of motion and motion event alerts. In parallel, an optic flow algorithm extracts both vertical and velocity information only on the motion events alerts provided by the visual cortex stage. Fig. 10 gives a general overview of the algorithm. This module provides three features per frame: the quantity of motion, horizontal velocity and vertical velocity.

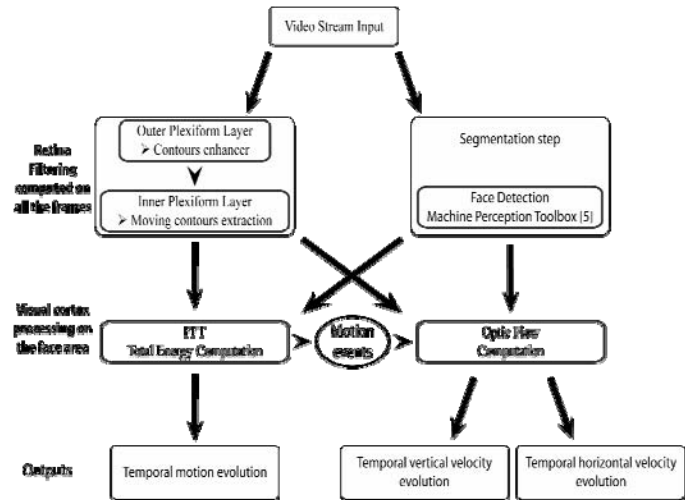


Fig. 10: Algorithm for rigid head motion data extraction

2) Description of the components

The first step consists in an efficient prefiltering [9]: the retina OPL (Outer Plexiform Layer) that enhances all contours by attenuating spatio-temporal noise, correcting luminance and whitening the spectrum (see Fig. 2). The IPL filter (Inner Plexiform Layer) [9] removes the static contours and extracts moving ones. This prefiltering is essential for data enhancement and allows minimizing the common problems of video acquisition such as luminance variations and noise.

The second step consists in a frequency analysis of the IPL filter output around the face whose response is presented on Fig.11. By computing the total energy of the amplitude spectrum of this output, as described in [10], we have information that depends linearly on the motion. The temporal evolution of this signal is the first data that is used in the sign language analyzer.



Fig. 11: Retina preprocessing outputs: extraction of enhanced contours (OPL) and moving contours (IPL)

In order to estimate the rigid head rotations [10], the proposed method analyses the spectrum of the IPL filter output in the log polar domain. It first detects head motion events [11] and is also able to extract its orientation. Then, in order to complete the description of the velocity, we propose to use features based on neuromorphic optical flow filters [12] which are oriented filters able to compute the velocity of the global head. Finally, optical flow is computed only when motion alerts are provided and its orientation is compared to the result given by the spectrum analysis. If the information is

redundant, then we extract the velocity value at each frame, either horizontal or vertical in order to simplify the system.

3) Extracted data sample

In the end, the head analyzer is able to provide three signals per frame, information related to the quantity of motion and the vertical and horizontal velocity values. Fig. 12 shows two examples of the evolution of these signals, first in the case of a sequence in which the person expresses an affirmative « Here », second in the case of the expression of the sign « Very clean ». For the first sign, the head motion is a sequence of vertical head nods. Then, the quantity of motion indicator shows a periodic variation of its values with high amplitude for maximum velocity. The vertical velocity presents non zero values only during motion and also exhibits a periodic variation. On the contrary, the horizontal velocity indicator remains at zero. The « Very clean » sign consists of two opposite horizontal head motions. The quantity of motion indicator exhibits them. This time, the horizontal motion reports the velocity sign and amplitude variations while the vertical velocity indicator remains at zero. On this last sequence, we can see that some false alarms can be generated at the velocity output level: For example, at frame 68, a false horizontal motion is detected, but since the value of the quantity of motion is low, this velocity should not be taken into account. This is the advantage of using two detection signals: the cortex analysis model helps the velocity analyzer.

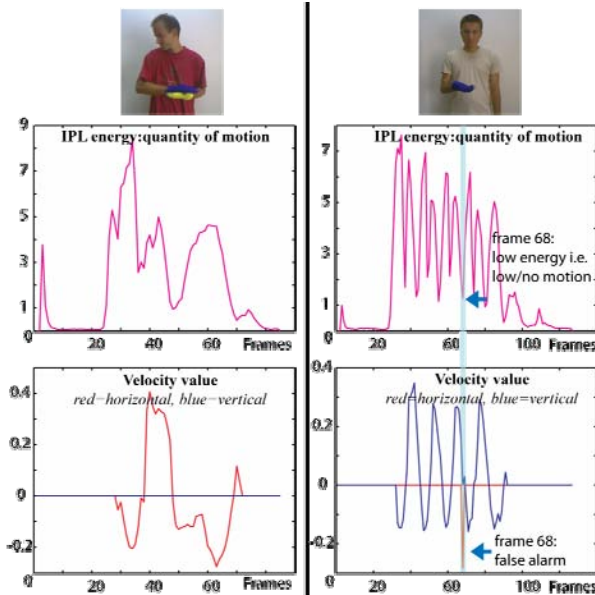


Fig. 12: Data extracted by the head analyzer

IV. SIGN LANGUAGE RECOGNITION

A. Preprocessing of sign sequences

The sequences obtained from the videos contain parts where the signer is not performing the sign (start and end parts) and some parts that can be considered as transition frames. These frames of the sequence are eliminated by looking at the result of the segmentation step:

- All the frames at the beginning of the sequence are eliminated until the hand is detected.
- If the hand can not be detected at the middle of the sequence for less than N frames, the shape information is copied from the last frame where there is detection.
- If the hand can not be detected for more than N consequent frames, the sign is assumed to be finished. Rests of the frames including the last N frames are eliminated.
- After indicating the start and end of the sequence and eliminating the unnecessary frames the transition frames can be eliminated by deleting T frames from the start and end of the sequence.

B. Sign features and normalization issues

1) Hand motion features

The trajectories must be further normalized to obtain translation and scale invariance. We use a similar normalization strategy as in [13]. The normalized trajectory coordinates are calculated with the following formulas:

Let $(\langle x_1; y_1 \rangle; \dots; \langle x_t; y_t \rangle; \dots; \langle x_N; y_N \rangle)$ be the hand trajectory where N is the sequence length. For translation normalization, define x_m and y_m :

$$x_m = (x_{\max} + x_{\min}) / 2$$

$$y_m = (y_{\max} + y_{\min}) / 2$$

where x_m and y_m are the mid-points of the range in x and y coordinates respectively. For scale normalization, define d_x and d_y :

$$d_x = (x_{\max} - x_{\min}) / 2$$

$$d_y = (y_{\max} - y_{\min}) / 2$$

where d_x and d_y are the amount of spread in x and y coordinates respectively. The scaling factor is selected to be the maximum of the spread in x and y coordinates, since scaling with different factors disturbs the shape.

$$d = \max(d_x; d_y)$$

The normalized trajectory coordinates, $(\langle x'_1; y'_1 \rangle; \dots; \langle x'_t; y'_t \rangle; \dots; \langle x'_N; y'_N \rangle)$ such that $0 \leq x'_t, y'_t \leq 1$, are then calculated as follows:

$$x'_t = 0.5 + 0.5 (x_t - x_m) / d$$

$$y'_t = 0.5 + 0.5 (y_t - y_m) / d$$

Since the signs can be also two handed, both hand trajectories must be normalized. However, normalizing the trajectory of the two hands independently may result in a possible loss of data. To solve this problem, the midpoints and the scaling factor of left and right hand trajectories are calculated jointly. Following this normalization step, the left and right hand trajectories are translated such that their starting position is (0,0).

2) Hand position features

In sign language, the position of the hand with respect to the body location is also important. We integrated position information by calculating the distance of the CoM of each hand to the face CoM. The distance at x and y coordinates are normalized by the face width and height respectively.

3) Hand shape features

All 19 hand shape features are normalized into values between 0 and 1. Features calculated as percentage (0 to 100%) are just divided by 100. The rest of features is normalized by using this equation:

$$F_{\text{normalized}} = (F - \min) / (\max - \min)$$

where \min is minimal value of feature (in training dataset) and \max is maximum value. In case smaller or greater value occurs, $F_{\text{normalized}}$ is truncated to stay in $<0,1>$.

4) Head motion features

Head motion analysis provides three features that can be used in the recognition: motion energy of the head, horizontal and vertical velocity of the head. However these features are not invariant to differences that can exist between different performances of the same sign. Moreover, the head motion is not directly synchronized with the hand motion. To handle inter and intra personal differences, adaptive smoothing is applied to head motion features where α is used as 0.5:

$$F_i = \alpha F_i + (1 - \alpha) F_{i-1}$$

This smoothing has an effect of cancelling the noise between different performances of a sign and creating a smoother pattern.

C. HMM modeling

After sequence pre-processing and normalization, HMM models are trained for each sign, using Baum-Welch algorithm. We have trained 3 different HMMs for comparison purposes:

- HMM_{manual} uses only hand information. Since hands form the basis of the signs, these models are expected to be very powerful in classification. However, absence of the head motion information prohibits a correct classification when the only difference of two signs is related to the head motion (i.e. here, ishere and not here)
- $HMM_{\text{manual\&nonmanual}}$ uses hand and head information. Since there is not a direct synchronization between hand and head motions, these models are not expected to have much better performance than HMM_{manual} . However using head information results in a slight increase in the performance.
- $HMM_{\text{nonmanual}}$ uses only head information. The head motion is complementary of the sign thus it can not be used alone to classify the signs. A data fusion methodology is needed to utilize these models together with models of manual components.

D. Fusion of different modalities of sign language

We have used a sequential score fusion strategy for combining manual and non-manual parts of the sign. We want our system to be as general as possible and capable of extending the sign set without changing the recognition system. Thus, we do not use any prior knowledge about the sign classes. For example, we know that here, *ishere* and *nothere* have exactly the same hand information but the head information differs. Using this prior information as a part of the recognition system increases the performance however the system loses its extendibility for upcoming signs since each sign will require a similar prior information. Instead we choose to extract the cluster information as a part of the recognition system.

Base decision is given by an HMM which uses both hand and head features in the same feature vector. However, the decision of these models is not totally correct since the head information is not utilized well. We used the likelihoods of $HMM_{\text{nonmanual}}$ to give the final decision.

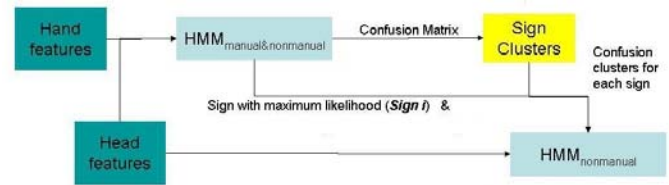


Fig. 13. Sequential fusion strategy

1) Training

- During training, models for each sign class are trained for $HMM_{\text{manual\&nonmanual}}$ and $HMM_{\text{nonmanual}}$.
- The cluster information for each sign is extracted from the confusion matrix of $HMM_{\text{manual\&nonmanual}}$. In the confusion matrix of the validation set the misclassifications are investigated. If all examples of a sign class are classified correctly, the cluster of that sign class only contains itself. For each misclassification, we add that sign class to the cluster.

2) Testing

The fusion strategy (Fig. 13) for an unseen test example is as follows:

- Likelihoods of $HMM_{\text{manual\&nonmanual}}$ for each sign class are calculated and the sign class with the maximum likelihood is selected as the base decision.
- Selected sign and its cluster information are sent to $HMM_{\text{nonmanual}}$.
- $HMM_{\text{nonmanual}}$ likelihood of the selected sign is calculated as well as the likelihoods of the signs in its cluster.
- Among these likelihoods, the sign class with the maximum $HMM_{\text{nonmanual}}$ likelihood is selected as the final decision.

V. SYNTHESIS AND ANIMATION

A. Head motion and facial expression synthesis

The head synthesis performed in the present project is based on the MPEG-4 Facial Animation Standard [14], [15]. In order to ease the synthesis of a virtual face, the MPEG-4 Facial Animation (FA) defines two sets of parameters in a standardized way. The first set of parameters, the Facial Definition Parameter (FDP) set, is used to define 84 Feature Points (FP), located on morphological places of the neutral head, as depicted in Fig. 14 (black points). The feature points serve as anchors for 3D face deformable meshes, represented by a set of 3D vertices.

The second set defined by the MPEG-4 Standard is the Facial Animation Parameter (FAP) set. The Facial Animation Parameters (FAPs) represent a complete set of basic facial actions closely related to muscle movements and therefore allow the representation of facial expressions by modifying the positions of the previously defined feature points (FP). They consist of a set of 2 high-level (visemes and 6 archetypal emotions) and 66 low-level parameters (depicted as white filled points on Fig. 14). In this project, we only use the low-level parameters which are basic deformations applied to specific morphological places of the face, like the top middle outer-lip, the bottom right eyelid, etc...

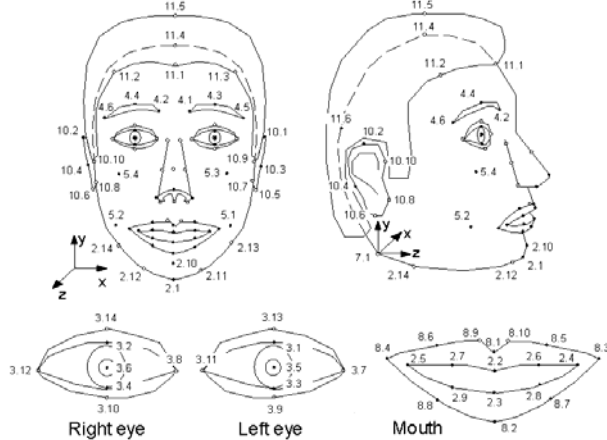


Fig. 14. The 3D feature points of the FDP set

The head synthesis system architecture is depicted on Fig. 15. As input, we receive the detected gesture (one data per sequence), the IPL energy, and the vertical and horizontal velocity of the head motion (as much data as frames in the sequence). We then filter and normalize these data in order to compute the head motion during the considered sequence. The result of the processing is expressed in terms of FAPs so that we can output a FAP file. The FAP file for the considered sequence is fed into the animation player. The animation player we used is an MPEG-4 compliant 3D talking head animation player developed by [16], part of an open source tools set available at [17]. Once rendered, we finally output an avi file containing the head synthesis sequence. Fig. 16 shows an example rendering with respect to the input data.

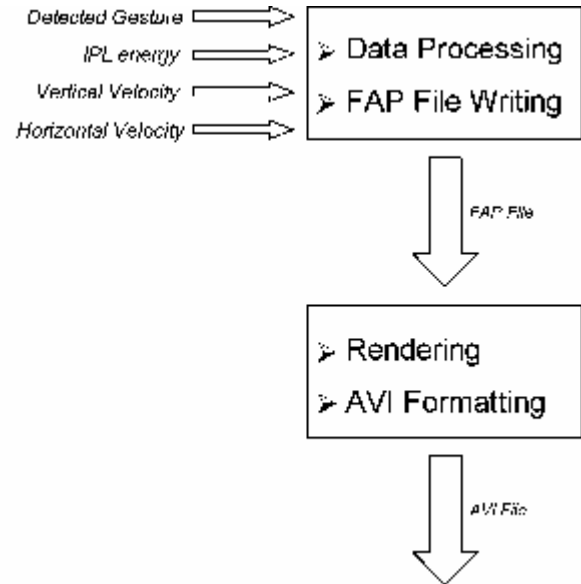


Fig. 15 : Head synthesis system architecture

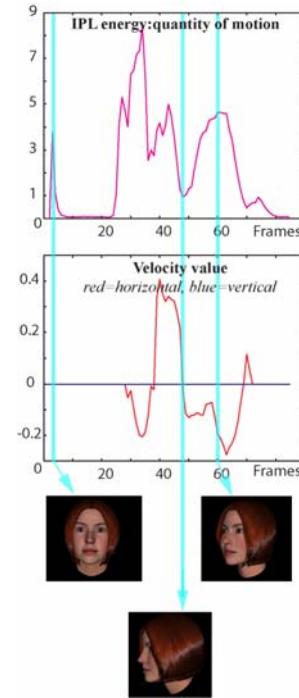


Fig. 16. Rendered head with different input values

B. Hands and arms synthesis

For the hands and arms synthesis, we use a library of predefined positions for each gesture. Each animation has key positions (the positions that define the gesture) and interpolated positions. To create an animation we only need to set the key positions in the correct frames depending on the speed we want to get, and then interpolate the rest of the frames.

For our system, we need to do two different adaptations: speed adaptation, and position adaptation. The gesture detected is supposed to be closer to its predefined one, so we

can define a set of steps that will be the same for each synthesis.

To create an animation from the features communicated by the analysis module, we have to follow the following steps:

1. *Physical features extraction*: from the input file, we extract the information about the head position, the length of the arms, and the maximal and minimal values of the hands coordinates (x,y). These will be used to normalize the information in order to adapt the system to our avatar features.

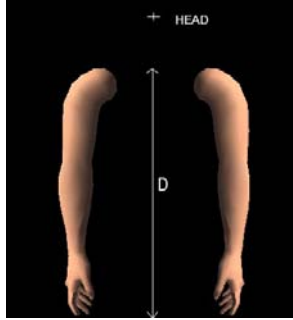


Fig. 17 : Arms system and main features.

2. *Speed features extraction*: For each gesture, we need to find the frames where the speed changes the most (border frames), because these frames will define our key positions. We find these features by differencing the coordinates of each frame and the next one. If the variation is smaller than a threshold we have set, it is supposed to be the same position than the last frame. It is necessary to know how much time we have to hold a position. With this extraction we have defined the frames where we will set the key positions.

3. *Physical adaptation and position definition*: We can adapt the parameters of our avatar according to the analysis results. Depending on the minimal and maximal values, we have extracted for x and y , we choose two predefined positions for each border frame, and then we interpolate these positions to get a new one. All positions we get here will be our key positions.

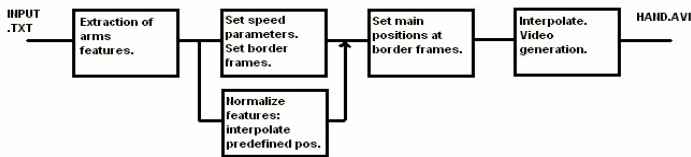


Fig. 18 : System structure for hands and arms synthesis.

4. *Set positions depending on the speed features*: To set the positions we have created, we take the border frames defined in step 2. We set the key positions for the border frames, holding them if necessary. After this, we only need to interpolate the rest of the frames, to get the final animation. From the final animation we will generate the video output that will be represented with the head result to show the complete avatar playing the gesture that the person in front of the camera did.

VI. RECOGNITION RESULTS

We have used 70% of the signs in the database for training and the rest for testing. The distributions of sign classes are equal both in training and test sets. Confusion matrices and performance results are reported on the test set.

The confusion matrix of HMMs that are trained by using only hand information is shown in TABLE III. The total recognition rate is 67%. However, it can be seen that most of the misclassifications are between the sign groups where the hand information is the same or similar and the main difference is in the head information, which is not utilized in this scheme. When sign clusters are taken into account, there are only five misclassifications out of 228; resulting in a 97.8% recognition rate.

TABLE III. CONFUSION MATRIX. (ONLY HAND INFORMATION)

Only Hand	door	to open	drink (noun)	to drink	here	is here?	not here	look at	look at cont.	look at reg.	study	study cont.	study reg.	afraid	very afraid	clean	very clean	fast	very fast
door	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
to open	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
drink (noun)	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
to drink	0	0	1	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
here	0	0	0	0	4	3	5	0	0	0	0	0	0	0	0	0	0	0	0
is here?	0	0	0	0	0	5	7	0	0	0	0	0	0	0	0	0	0	0	0
not here	0	0	0	0	0	5	7	0	0	0	0	0	0	0	0	0	0	0	0
look at	0	0	0	0	0	0	0	7	1	4	0	0	0	0	0	0	0	0	0
look at cont.	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0
look at reg.	3	1	0	0	0	0	0	0	1	7	0	0	0	0	0	0	0	0	0
study	0	0	0	0	0	0	0	0	0	0	4	4	4	0	0	0	0	0	0
study cont.	0	0	0	0	0	0	0	0	0	0	0	8	4	0	0	0	0	0	0
study reg.	0	0	0	0	0	0	0	0	0	0	1	1	10	0	0	0	0	0	0
afraid	0	0	0	0	0	0	0	0	0	0	0	0	0	2	10	0	0	0	0
very afraid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0
clean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	6	0	0
very clean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	10	0	0
fast	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	3	8
very fast	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	11

The confusion matrix of HMMs that are trained by the combined feature vector of hand and head information is shown in TABLE IV. The total performance is 77%. All misclassifications, except one, are between the sign groups. Although there is a slight increase in the performance, this fusion method does not utilize the head information effectively. Therefore, we have adopted the sequential fusion strategy described in Section IVD.

The confusion matrix of the sequential fusion methodology is shown in TABLE IV. The total performance is 85.5%. The misclassifications between the sign groups are very few except for the study and look at sign groups. The reason of these misclassifications can be related to the deficiency of vision hardware or to the misleading feature values:

- The *study* sign: The confusion between *study regularly* and *study continuously* can stem from a deficiency of the 2D capture system. These two signs differ mainly in the third dimension, which we cannot capture. The confusion between *study* and *study regularly* can be a result of over-smoothing the trajectory.
- For the *look at* sign, the hands can be in front of the head for many of the frames. For those frames, the face detector may fail to detect the face and may provide wrong feature values which can mislead the recognizer.

Manual sign classification performance is 99.5%, which means only one sign is misclassified out of 228.

TABLE IV. CONFUSION MATRIX. FEATURE LEVEL FUSION

Hand Head feature fusion	door	to open	drink (noun)	to drink	here	is here?	not here	look at	look at cont.	look at reg.	study	study cont.	study reg.	afraid	very afraid	clean	very clean	fast	very fast
door	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
to open	1	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
drink (noun)	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
to drink	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
here	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0	0	0
is here?	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0
not here	0	0	0	0	0	2	10	0	0	0	0	0	0	0	0	0	0	0	0
look at	0	0	0	0	0	0	0	7	1	4	0	0	0	0	0	0	0	0	0
look at cont.	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0
look at reg.	1	0	0	0	0	0	0	0	4	7	0	0	0	0	0	0	0	0	0
study	0	0	0	0	0	0	0	0	0	0	3	0	9	0	0	0	0	0	0
study cont.	0	0	0	0	0	0	0	0	0	0	0	8	4	0	0	0	0	0	0
study reg.	0	0	0	0	0	0	0	0	0	0	0	2	10	0	0	0	0	0	0
afraid	0	0	0	0	0	0	0	0	0	0	0	0	0	3	9	0	0	0	0
very afraid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0
clean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	1	0	0
very clean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	10	0	0
fast	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	6
very fast	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12

TABLE 3. CONFUSION MATRIX. SEQUENTIAL FUSION

Hand Head sequential fusion	door	to open	drink (noun)	to drink	here	is here?	not here	look at	look at cont.	look at reg.	study	study cont.	study reg.	afraid	very afraid	clean	very clean	fast	very fast
door	11	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
to open	1	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
drink (noun)	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
to drink	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
here	0	0	0	0	11	0	1	0	0	0	0	0	0	0	0	0	0	0	0
is here?	0	0	0	0	0	11	1	0	0	0	0	0	0	0	0	0	0	0	0
not here	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0
look at	0	0	0	0	0	0	0	10	0	2	0	0	0	0	0	0	0	0	0
look at cont.	0	0	0	0	0	0	0	2	9	1	0	0	0	0	0	0	0	0	0
look at reg.	1	0	0	0	0	0	0	5	4	2	0	0	0	0	0	0	0	0	0
study	0	0	0	0	0	0	0	0	0	0	9	3	0	0	0	0	0	0	0
study cont.	0	0	0	0	0	0	0	0	0	0	0	9	3	0	0	0	0	0	0
study reg.	0	0	0	0	0	0	0	0	0	0	0	5	7	0	0	0	0	0	0
afraid	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0
very afraid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0
clean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0
very clean	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0
fast	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0
very fast	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	10

VII. CONCLUSIONS AND FUTURE WORK

In this project, we have developed a sign tutor application that lets users learn and practice signs from a predefined library. The tutor application records the practiced signs; analyses the hand shapes and movements as well as the head movements, classifies the sign, and gives feedback to the user. The feedback consists of both text information and synthesized video, which shows the user a caricaturized version of his movements when the sign is correctly classified. Our performance tests yield a 99% recognition rate on signs involving manual gestures and 85% recognition rate on signs that involve both manual and non manual components, such as head movement and facial expressions.

ACKNOWLEDGMENT

We thank Jakov Pavlek and Vjekoslav Levacic, who have volunteered to be in the sign database.

REFERENCES

- [1] H-K. Lee, J-H Kim, "Gesture spotting from continuous hand motion" in *Pattern Recognition Letters*, 19(5-6), pp. 513-520, 1998.
- [2] T. Starner and A. Pentland. "Realtime american sign language recognition from video using hidden markov models". Technical report, MIT Media Laboratory, 1996.
- [3] C. Vogler and D. Metaxas. "Adapting hidden markov models for asl recognition by using three-dimensional computer vision methods". In *Conference on Systems, Man and Cybernetics (SMC'97)*, Orlando, FL, pages 156–161, 1997.
- [4] C. Vogler and D. Metaxas. "ASL recognition based on a coupling between HMMs and 3D motion analysis". In *International Conference on Computer Vision (ICCV'98)*, Mumbai, India, 1998.
- [5] Ong, S. Ranganath. "Automatic Sign Language Analysis: A survey and the Future beyond Lexical Meaning", *IEEE Transactions on PAMI*, vol.27, no.6, pp.873-891, June 2005.
- [6] O. Aran, C. Keskin, L. Akarun, "Sign language tutoring tool", in *Proceedings EUSIPCO'05*, September 2005.
- [7] Jayaram S., S. Schumge, M. C. Shin, and L. V. Tsap, "Effect of Colorspace Transformation, the Illuminance Component, and Color Modeling on Skin Detection," *cvpr*, pp. 813-818, 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04) - Volume 2, 2004.
- [8] Albiol, A., L. Torres, and E. J. Delp, "Optimum color spaces for skin detection", *Proceedings of IEEE International Conference on Image Processing*, Vol. 1, 122--124., 2001.
- [9] Baudot W., "The neural information processing in the vertebrate retina: A melting pot of ideas for artificial vision", PhD Thesis in Computer Science, INPG (France) december 1994.
- [10] Benoit A., Caplier A. "Head Nods Analysis: Interpretation of Non Verbal Communication Gestures" *IEEE, ICIP 2005*, Genova, Italy
- [11] Benoit A., Caplier A. "Hypovigilance Analysis: Open or Closed Eye or Mouth ? Blinking or Yawning Frequency ?" *IEEE, AVSS 2005*, Como, Italy
- [12] Torralba A. B., Hervault J. (1999). "An efficient neuromorphic analog network for motion estimation." *IEEE Transactions on Circuits and Systems-I: Special Issue on Bio-Inspired Processors and CNNs for Vision*. Vol 46, No. 2, February 1999.
- [13] Oya Aran, Lale Akarun "Recognizing two handed gestures with generative, discriminative and ensemble methods via Fisher kernels", *International Workshop on Multimedia Content Representation, Classification and Security, (MRC'S'06)*, Istanbul, September 2006.
- [14] M. Tekalp, Face and 2D mesh animation in MPEG-4, Tutorial Issue On The MPEG-4 Standard, Image Communication Journal, Elsevier, 1999.
- [15] I.S. Pandzic & R. Forchheimer, MPEG-4 Facial Animation: The standard, implementation and applications, Wiley, 2002.
- [16] K. Balci, XfaceEd: authoring tool for embodied conversational agents, 7th International Conference on Multimodal Interfaces (ICMI '05), 2005.
- [17] <http://xface.itec.it/index.html>
- [18] Machine Perception Toolbox (MPT) <http://mplab.ucsd.edu/grants/project1/free-software/MPTWebSite/API/>.
- [19] Kevin Murphy, "Bayes Net Toolbox for MATLAB,," <http://bnt.sourceforge.net/>
- [20] Intel Open Source Computer Vision Library, <http://opencvlibrary.sourceforge.net/>

APPENDIX : SOFTWARE NOTES

Since the individual parts in this project were coded in C, C++ and MATLAB, we preferred MATLAB to combine them for the tutor. MATLAB GUI was used to prepare the user interface.

We used the "Machine Perception Toolbox" [18] for head analysis. For HMM training, HMM routines in [19] are used. We also used "Intel Open Source Computer Vision Library" [20] routines in our project.

Multimodal Speaker Conversion

— his master’s voice...and face —

T. Dutoit*, A. Holzapfel†, M. Jottrand*, F. Marqués‡, A. Moinet* F. Ofli§, J. Pérez‡ and Y. Stylianou†
 *Faculte Polytechnique de Mons - BELGIUM †University of Crete - GREECE ‡Universitat Politècnica de Catalunya - SPAIN §Koc University - TURKEY

Abstract—The goal of this project is to convert a given speaker’s speech (the Source speaker) into another identified voice (the Target speaker) as well as analysing the face animation of the source to animate a 3D avatar imitating the source facial movements. We assume we have at our disposal a large amount of speech samples from the source and target voices with a reasonable amount of parallel data. Speech and video are processed separately and recombined at the end.

Voice conversion is obtained in two steps: a voice mapping step followed by a speech synthesis step. In the speech synthesis step, we specifically propose to select speech frames directly from the large target speech corpus, in a way that recall the unit-selection principle used in state-of-the-art text-to-speech systems.

The output of this four weeks work can be summarized as: a tailored source database, a set of open-source MATLAB and C files and finally audio and video files obtained by our conversion method. Experimental results show that we cannot aim to reach the target with our LPC synthesis method; further work is required to enhance the quality of the speech.

Index Terms—voice conversion, speech-to-speech conversion, speaker mapping, face tracking, cloning, morphing, avatar control.

I. INTRODUCTION

THIS project aims at converting a given speaker speech and facial movements into those of another (identified) speaker. More precisely, it focuses on controlling a 3D talking face (the avatar of the target speaker) using the most natural interfaces available : the speech and facial movements of a human speaker (the source speaker). It also assumes that the target will generally be different from the source, so that the project goes much further than the design of a state-of-the-art avatar controlled by the original speaker (i.e., the same as the one whose face was used to create the avatar). As a matter of fact, two additional problems are encountered here:

- That of voice conversion, in order to make the target talking face speak with the target’s voice, producing the source’s words.
- That of facial movement conversion, in order to adapt the movement excursions of the source face to match the movement excursions of the target face.

The multimodal conversion problem we consider here, however, is limited to signal-level modification, as opposed to semantic conversion. The latter would enable, for instance, filtering out some (most often paralinguistic) communication

acts (be them speech and/or facial movements, such as tics) of the source which the target never produces, or conversely adding target movements not present in the source but usually in the target.

Moreover, we constraint the conversion process to maintain some large-sense synchronicity between source and target: we do not aim at adapting speech rate at the phoneme level, but rather simplifying it to a possible overall speech rate adaptation. Similarly, we do not consider a possible syllable-level F0 conversion from source to target, but rather aim at a possible overall F0 adaptation ratio.

It will be assumed that a large amount of studio-quality speech data is available from the source *and* from the target. This is not a usual assumption for systems which try to put new words in the (virtual) mouth of most VIP characters (whom cannot be easily forced to attend a formal recording session in a studio). The assumption, however, remains realistic in the case of a source speaker driving a famous target speaker whose voice has been recorded in large amounts but who is simply no longer (or not always) available. It is also assumed that this speech data is available in the form of parallel speech corpora (i.e., recordings of the same sentences by both the source and the target).

A typical application of this project is therefore that of a human actor controlling the speech and facial movements of a 3D character whose voice is well-known to the audience. Another possible use is for psychologists talking to children through an avatar whose voice should be kept unchanged among sessions, even though the psychologist may change. If we reduce this project to its speech component, a typical application is that of a tool for producing natural sounding voice prompts with the voice of a voice talent, based on natural speech (prosody and spectral features) produced by a human speaker.

Last but not least, a side constraint of this work is that we aim at using and producing open-source code, as required by the eINTERFACE workshop organization.

Figure 1 shows the necessary steps involved in our project: speech/face analysis, voice/facial movements mapping, and speech/face synthesis. This report is therefore organized as follows. Section II browses the state-of-the-art in speech analysis, mapping, and synthesis for voice conversion, and examines the approach we have followed in this project. Section III examines facial movement analysis, mapping and synthesis for avatar face animation, and gives details on the algorithms we have used. This is followed in section IV by

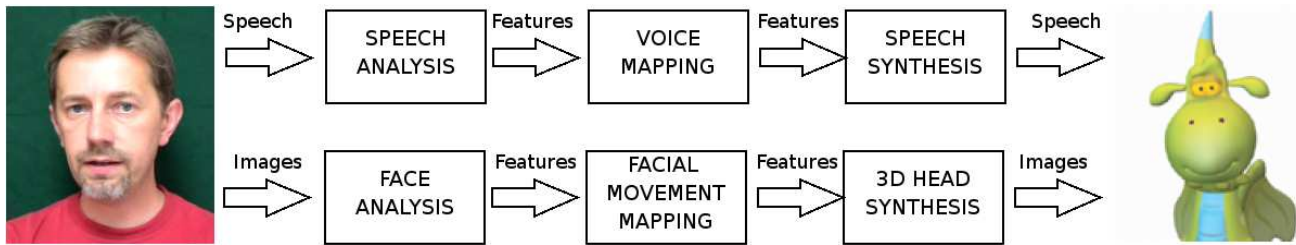


Fig. 1. An example of 3D cartoon avatar control using source speech and facial movements, and mapping them to target speech and face.

the experiments we did, using a specially designed database, and by some informal assesment of the quality we reached at the end of the 4-weeks project. The paper is concluded in Section V by perspectives for further developments.

II. SPEECH ANALYSIS, MAPPING, AND SYNTHESIS FOR VOICE CONVERSION

We understand the Voice Conversion (VC) process as a modification of a speaker voice (*source speaker*) so that it resembles that of another given speaker (*target speaker*). Our goal is to obtain a transformation that makes the source speech sound as if it were spoken by the target speaker.

In this work, we have approached the speech conversion task from x (source speaker) to y (target speaker) in two independent blocks:

- mapping from x to y' (a first approximate of y) using the (reduced) parallel corpus, and
- speech-to-speech (S2S) synthesis from y' to y'' (a second –and more accurate– approximate of y), using the (full) target corpus.

The first block involves aligning the data on a frame by frame basis (section II-A) and building a mapping function (either using Gaussian Mixture Models or Conditional Vector Quantization, as will be seen in section II-B). In the second block, for the successive frames of y' , we select new frames from a large database of y voice, in such a way that we guarantee both maximum similarity to the input frames, and maximum continuity (the details are given in section II-D.3). This can be seen as a smoothing step, made necessary by the fact that the reduced size of the parallel corpus resulted in large, non-acceptable discontinuities in the synthetic speech.

It is worth mentioning that since a large amount of data for the target is available for this particular application (as it is also the case for the design of a state-of-the-art text-to-speech (TTS) system), the challenge of this project is to be able to produce more natural sounding speech than that of a TTS system, trained with the same amount of data, and used to synthesize the phonemes obtained by automatic phonetic segmentation (speech recognition) of the input source waveform. As a matter of fact, the input of the voice conversion system is itself natural speech, which can hopefully be put to profit to deliver natural-sounding output speech. The intonation of the source speech, for instance, can readily be used (possibly modified) to produce the intonation of the target speech, and thus obtain an improvement in synthesis quality over standard TTS.

Furthermore, it is then possible to establish upper and lower bounds to the synthesis quality we can obtain with our frame-based voice conversion system. The lower bound would be that obtained with a TTS backend. In this case, we would directly use the prosody of the source waveform and the phonemes, obtained with automatic speech recognition. The TTS would then only need to perform unit selection and concatenation based on this requirements. This way we expect to obtain a higher similarity and naturality than when using only the source (recognized) text as input. The upper bound would of course be the natural target speech.

A. Data alignment

Although the corpus used for the voice mapping part of +this project (see Section IV) consists of parallel utterances, some timing differences are unavoidable due to different speaker characteristics (pause durations, speech rate, etc.). Since the training of the voice mapping block of fig. 1 requires parallel data vectors, the utterances of the source and target speakers have been aligned using a dynamic time warping (DTW) procedure. For this project, we used the DTW algorithm implemented by Dan Ellis¹ and released under GPL. The *local match* measure is computed as the cosine distance (angle of the vectors) between the Short-Time Fourier Transform (STFT) magnitudes. The left part of fig. 2 represents the local match scores matrix, with darker zones indicating high similarity values (ideally we would have a dark stripe down the leading diagonal). In the right part of the figure we can see the minimum-cost-to-this-point matrix (lighter color indicates lower cost). In both subfigures, the red line shows the lowest-cost path obtained by the DTW algorithm.

The DTW algorithm introduces some unavoidable errors due to the coexistence of intrinsic spectral differences between the two speakers. An iterative procedure can be applied to improve the alignment, by reapplying the DTW method between the converted and target envelopes [1]. After each iteration, a new mapping function can be estimated between the newly aligned original source and target data. In this project, convergence was reached after three iterations.

B. Voice mapping

When converting the voice of the source to the voice of a the target speaker we assume that these two voices are defined by their spectral spaces \mathcal{X} and \mathcal{Y} respectively. Our problem

¹<http://labrosa.ee.columbia.edu/matlab/dtw/>

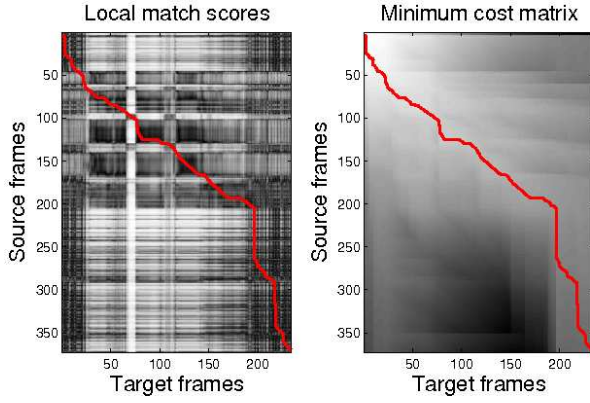


Fig. 2. Left: Graphical representation of the local match scores matrix (darker colors represent a closer match). Right: Minimum cost matrix (lighter colors indicate a lower cost). The red line shows the optimal alignment path as found by the DTW algorithm.

in voice conversion is two-fold: at first we have to find a way to model these spaces and then we have to find a way to map a previously unknown example from the source space to the target space. In order to be able to find such a mapping we assume that there is aligned training data available. This means that we have two sets of spectral vectors \mathbf{x}_t and \mathbf{y}_t that describe spectral envelopes from source and target speakers respectively. The two sets of vectors $\{\mathbf{x}_t, t = 1, \dots, N\}$ and $\{\mathbf{y}_t, t = 1, \dots, N\}$ have the same length N and are supposed to describe sentences uttered in parallel by source and target. What is desired is a function $\mathcal{F}()$ such that the transformed envelope $\mathcal{F}(\mathbf{x}_t)$ best matches the target envelope \mathbf{y}_t , for all envelopes in the learning set ($t = 1, \dots, N$).

In our project we tried two different approaches in order to achieve the goal of conversion: Gaussian Mixture Models and Conditional Vector Quantization.

a) Gaussian Mixture Models: The first approach has been described by Stylianou *et al.* [1] and is based on a description of the source space using Gaussian Mixture Models:

$$p(\mathbf{x}) = \sum_{i=1}^M \alpha_i \mathcal{N}(\mathbf{x}; \mu_i, \Sigma_i) \quad (1)$$

where M is the number of Gaussians, μ_i, Σ_i are the mean vector and the covariance matrix of the i -th Gaussian component, and α_i are the weights used to combine the different components. These M Gaussian components can be regarded as classes within the spectral space of the source and a vector \mathbf{x}_t can be classified to one of the classes using maximum likelihood. The mapping to the target space is done by using these parameters in the conversion function

$$\mathcal{F}(\mathbf{x}_t) = \sum_{i=1}^M P(C_i | \mathbf{x}_t) [\nu_i + \Gamma_i \Sigma_i^{-1} (\mathbf{x}_t - \mu_i)] \quad (2)$$

where ν and Γ are related to the mean target and the cross-covariance matrix of the source and target vectors. The parameters of the conversion function are determined by minimization of the total quadratic spectral distortion between

the converted envelopes and the target envelopes:

$$\epsilon = \sum_{t=1}^N \|\mathbf{y}_t - \mathcal{F}(\mathbf{x}_t)\|^2. \quad (3)$$

For details on the minimization see [1].

b) Conditional Vector Quantization: The second method for voice conversion applies a Conditional Vector Quantization as presented in [2]. In contrast to the first method we get hard cluster boundaries by using a standard LBG clustering for the source space giving us a codebook $C_x \equiv \{\hat{\mathbf{x}}_i, i = 1 \dots m\}$. Then the mapping function finds for each of these clusters a different codebook C_y with k entries for each source space cluster. The criterion function minimized in this case is the approximation to the average distortion D given by

$$D \approx \sum_{m=1}^M \left[\frac{1}{N} \sum_{n=1}^N p(\hat{\mathbf{x}}_m | \mathbf{x}_n) \sum_{k=1}^K p(\hat{\mathbf{y}}_{m,k} | \hat{\mathbf{x}}_m, \mathbf{y}_n) d(\mathbf{y}_n, \hat{\mathbf{y}}_{m,k}) \right] \quad (4)$$

The conditional probability $p(\hat{\mathbf{x}}_m | \mathbf{x}_n)$ is the association probability relating the input vector \mathbf{x}_n with codevector $\hat{\mathbf{x}}_m$, while the association probability $p(\hat{\mathbf{y}}_{m,k} | \hat{\mathbf{x}}_m, \mathbf{y}_n)$ relates the output vector \mathbf{y}_n with the codevector $\hat{\mathbf{y}}_{m,k}$ of the m -th subcodebook of C_y .

The mapping for a source feature vector \mathbf{x}_i is done by choosing the nearest source space cluster using Euclidean distance. This provides us with a subcodebook of C_y with K entries. For an utterance of length N we construct a lattice of $K \times N$ elements and we find a minimum weight path using again Euclidean distance from frame to frame providing us with a sequence of N vectors in \mathcal{Y} .

C. Frame selection algorithm

Once the features of the frames of the original speaker have been converted using either the GMM mapping or the CVQ-based conversion, the converted features are used as inputs of the unit-selection algorithm.

This algorithm is basically working like any TTS unit-selection system. However, state-of-the-art TTS systems based on unit-selection usually deal with diphones, phones or parts of phones [3] while our algorithm uses smaller units : 32 ms frames (with a constant shift of 8 ms between each frame).

For this part of the system, we use the complete target speech database (as opposed to the mapping system, which only used the aligned sub-part of it; see Section IV for details on the databases). Among all the frames in the target database (cmu_us_aws-arctic '95), we select a sequence of frames $\hat{\mathbf{Y}} = [\hat{y}^{(1)} \dots \hat{y}^{(t)} \dots \hat{y}^{(T)}]$ that best matches the sequence of frames output by the mapping function: $\hat{\mathbf{Y}} = [\hat{y}^{(1)} \dots \hat{y}^{(t)} \dots \hat{y}^{(T)}]$. This selection is made using the Viterbi algorithm [4], [5] to minimize the global distance between $\hat{\mathbf{Y}}$ and $\hat{\mathbf{Y}}$. This global distance is a combination of target and concatenation distances which are detailed in the next three subsections. This approach is very similar to that developed in Suenderman *et al.* [6], with the difference that in our approach the target sequence for the frame selection algorithm is the mapped sequence $\hat{\mathbf{Y}}$, while Suenderman *et al.* use the input sequence X as target."

1) *Clustering, clusters and n-best selection*: In order to reduce the computation time, the database has been divided into 256 clusters using the LBG method [7]. When we use only the parallel recordings as a database, there are about 300 frames per cluster and each cluster has a centroid which is a vector of the mean values of the features of the frames inside the cluster. Therefore, for each set of features $\dot{y}^{(t)}$, the algorithm first selects the cluster with the closest centroid. The closeness of the centroid is measured using a weighted euclidean distance

$$\text{closest centroid} = \underset{c=1, \dots, C}{\operatorname{argmin}} \sum_{i=1}^N w_i \cdot \left(\dot{y}_i^{(t)} - \ddot{y}_i^{(c)} \right)^2, \quad (5)$$

where N is the dimension of the feature vectors, $\dot{y}_i^{(t)}$ is the i^{th} component of the feature vector produced by the mapping function at time t , $\ddot{y}_i^{(c)}$ is the i^{th} component of the c^{th} centroid of the database and w_i is the weighting factor associated to that i^{th} component.

Then, for each frame ($1, \dots, m_c, \dots, M_c$) in the chosen cluster c , the weighted euclidean distance between the feature vectors $\dot{y}^{(t)}$ and $\ddot{y}^{(m_c)}$ is computed. This distance will be used as the target distance t_{dist} by the Viterbi algorithm:

$$t_{\text{dist}}(t, m_c) = \sum_{i=1}^N w_i \cdot \left(\dot{y}_i^{(t)} - \ddot{y}_i^{(m_c)} \right)^2 \quad (6)$$

Finally, if the n-best option is activated, only the NBEST closest $\ddot{y}^{(m_c)}$ to the $\dot{y}^{(t)}$ are selected as candidates for Viterbi.

2) *Concatenation distance*: During the induction step, the Viterbi algorithm has to compute the cost of all the transitions from each feature vector $\ddot{y}^{(m_{c(t)})}$, selected at time instant t , to each vector $\ddot{y}^{(m_{c(t+1)})}$, selected at time $t+1$. Again, these concatenation costs (c_{dist}) are measured using a weighted euclidean distance:

$$c_{\text{dist}}(m_{c(t)}, m_{c(t+1)}) = \sum_{i=1}^N w_i \cdot \left(\ddot{y}_i^{(m_{c(t+1)})} - \ddot{y}_i^{(m_{c(t)})} \right)^2 \quad (7)$$

This step is the most time-consuming part of the Viterbi algorithm and is the reason why clusters and n-best selections are so important. They reduce the transition possibilities but in return they dramatically increase the search speed without actually nor notably damaging the final output.

At this point, one important remark has to be done : the concatenation distance should advantage the selection of neighbour frames to reduce discontinuities during the synthesis of speech. Therefore, before computing the distance between two feature vectors, the process checks whether the corresponding frames are consecutive in a wav file of the database. In case they are, the distance is automatically set to zero.

3) *Implementation*: The very first tests were made using Matlab, however it early appeared that this was very time-consuming. In order to reduce the duration of the Viterbi algorithm, the whole program has been rewritten in C so that it could be compiled in a mex-file. Mex-files are pre-compiled libraries used by Matlab as any other Matlab script. The difference with usual scripts is that the functions implemented in these libraries are way faster (see Annex V-G in p. 11).

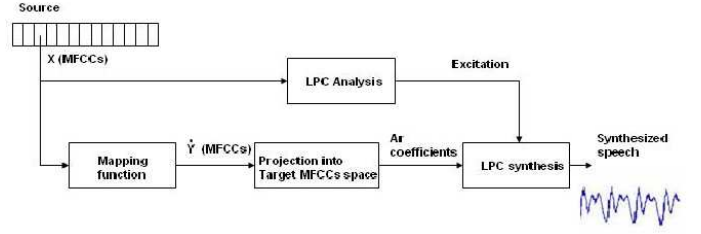


Fig. 3. Block diagram of the selection method using a projection of the mapped source MFCCs vectors into the target MFCCs space.

D. Speech synthesis

As a result of the previous steps (mapping, possibly followed by a frame-selection algorithm as described in II-C), we obtained a sequence of converted (target) frames that best represent the voice transformation of a given source utterance. The next step is to generate the converted speech resulting from this sequence. Although it is possible to regenerate the speech from the MFCC parameters [8], the resulting quality is not acceptable for the task, since the parametrization is not a lossless procedure. In particular, the phase information is completely lost and must be estimated to provide the synthetic voice with a higher degree of naturalness. In this subsection, we first produce $\ddot{y}(n)$ using the source speaker excitation as input of LPC filters modelling the target speech. We then examine the production of $\ddot{y}(n)$ by overlap-adding speech frames (i.e., samples) extracted directly from the target database.

1) *LP-based speech synthesis, without Viterbi*: This method is illustrated in fig. 3. The speech signal is segmented into overlapping frames and MFCCs are computed for each frame (vectors X in the figure). The mapping function (continuous probabilistic mapping function based on GMM and described earlier) computes transformed MFCCs vectors (\hat{Y}). Each \hat{Y} is then compared to all target database MFCC vectors in order to get the closest match \tilde{Y} . For each target MFCC vector, we also have memorized the corresponding auto-regressive coefficients. Thus, for each frame of the source speaker, we can get the corresponding autoregressive coefficients for the target speaker.

In parallel, for each source frame, we run a LPC analysis to extract the LP residual of the source. Finally, to synthesize converted speech, we use this residual as input excitation for the LPC synthesis filter with the auto-regressive coefficients obtained.

We know that to keep the excitation of the source is not an ideal solution since it still keeps information from the source. To approach a bit more the target voice, a pitch modification can be done on the converted speech by analysing the pitch of the source and of the target.

2) *LP-based speech synthesis, with Viterbi*: The voice conversion system by LPC analysis and synthesis using the Viterbi algorithm is detailed in fig. 4. As in the previous method, the mapping function transforms MFCCs from the source speech signal (X) into \hat{Y} MFCC vectors. For each frame of the processed sentence, the Viterbi algorithm (that has been previously described) gives as output an MFCC vector

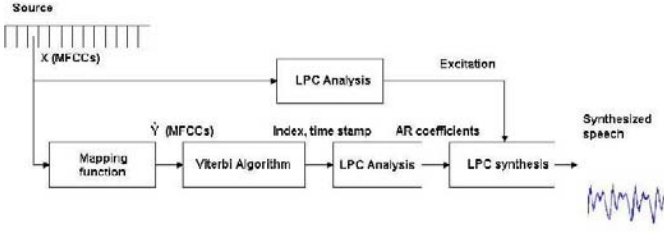


Fig. 4. Block diagram of the selection method using the Viterbi algorithm.

taken directly from the MFCC vectors of the target speech database. It also provides the index of the sentence the frame chosen comes from and the time stamp of the centre of the frame. From this information, one extracts the frame from its wave file and computes an LPC analysis to get the new AR coefficients that will be used for the synthesis.

In comparison with the previous method, one can expect an improvement in the choice of the frames in the target database, since the Viterbi algorithm takes the concatenation cost into account (and not only the target cost).

3) *Speech to Speech synthesis*: Instead of trying to produce the converted speech samples by LPC analysis-synthesis, as in the previous subsection, it is also possible to deliver speech by overlap-adding speech frames taken directly from the target speech files.

Here we use the natural target speech frames associated to each MFCC vector. The problem then reduces to combining these frames in order to achieve the highest quality possible. We use a simple OverLap-and-Add (OLA) procedure on the sequence of frames. However, there may be important problems associated to the discontinuities between the frames. To partially overcome this drawback, we apply a correction on the frame positions, based on the local maximization of the correlation between the already generated synthetic speech and each new frame, as used in WSOLA [9].

III. FACIAL EXPRESSION ANALYSIS, MAPPING, AND SYNTHESIS FOR FACE ANIMATION

Facial expression analysis and synthesis techniques have received increasing interest in recent years. Numerous new applications can be found, for instance in the areas of low bit-rate communication or in the film industry for animating 3D characters.

In this project, we are interested in head-and-shoulder video sequences, which we use to control the lip, eye, eyebrow, and head movements of a 3D talking head. First a 3D head model of the target is created (basically any 3D head will do, including a cartoon-like head). This step typically involves the extraction of Facial Definition Parameters (FDPs) from photographs of the target speaker. The system then analyzes the source video sequences and estimates 3D motion and facial expressions using the 3D head model information. The expressions are represented by a set of facial animation parameters (FAPs) which are part of the MPEG4 standard [10] (just as FDPs). The target 3D head model is then deformed according to the FAPs of the source, and new frames are synthesized using computer graphics techniques. Good examples of such

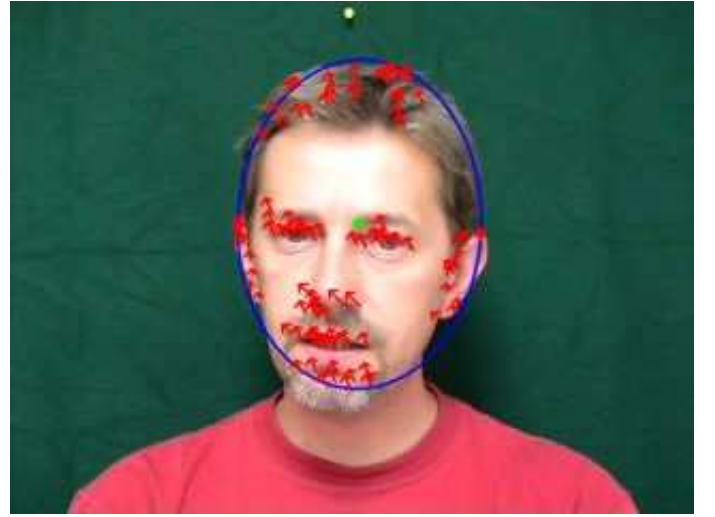


Fig. 5. An ellipse is fitted to skin blob and optical flow vectors are calculated in this region

video-based avatar control can be found in the work of Eisert [11].

A. Face Analysis

Face analysis process mainly consists of three tasks: detecting and tracking face region, extracting features for facial gestures from the detected face region, and tracking these features throughout the source video. In this framework, automatic face detection and computation of 8 quadratic global head motion parameters is done according to [12]. Figure 5 shows an ellipse fitted to the detected face region and optical vectors calculated between two consecutive frames that will be used in the process of computing quadratic global head motion parameters.

The global head motion parameters are used to

- estimate the position of head in the next frame,
- approximate the 3D movement of the head and
- calculate the canonical displacement vectors of facial feature points over the face region.

Once the face position is known, a set of useful feature points for face components, which are lips, eyes, and eyebrows in this scenario, are defined and tracked. For this purpose, *Active Appearance Models* (AAM) approach that was introduced by Cootes, Edwards and Taylor, is used as a means of modeling and tracking face components [13], [14], [15], [16].

AAM is, in essence, a combination of ideas from Eigen-face Models [17], Point Distribution Models [18], and Active Shape Models [19] and has its roots in model-based approaches towards image interpretation named *deformable template models* where a deformable template model can be characterized as a model which, under an implicit or explicit optimization criterion, deforms a shape to match a known object in a given image [20]. Consequently, AAMs establish a compact parametrization of shape and texture, as learned from a representative training set. Thus, AAMs require model training phase before they are used to process unseen images. In this training phase, the AAM learns a linear model of the

correlation between parameter displacements and the induced residuals.

During search for a face in a new frame, residuals are iteratively measured and used to correct the current parameters with respect to the main model, leading to a better fit. After a few iterations, a good overall match is obtained [15]. Applying this search method to each video frame using the model obtained after training will constitute the tracking part of the task. Figure 6 demonstrates a sequence of consecutive frames as a result of the tracking task. The output of the tracking process will be the set of 2D positions of key points of the face components for each frame which will be input to the facial movement mapping module.

B. Facial Movement Mapping

Since the previous module tracks pixel locations of the key feature points of the face components and the next module will synthesize a 3D head animation using an avatar model based on MPEG-4 parameters, the mapping function will compute MPEG-4 parameters from a set of 2D image locations.

MPEG-4 Facial Animation defines two sets of parameters: the Facial Definition Parameter (FDP) set and the Facial Animation Parameter (FAP) set [21], [22]. These two sets provide a common framework for animating a 3D face deformable mesh model with the help of high-level and low-level facial actions, closely related to facial muscle movements.

The first set of parameters, FDPs, is used to define feature points that are basic components in 3D face deformable meshes, represented by a 3D set of vertices. The movements of these vertices drive the deformations to be applied to the model to animate the desired facial expressions. 84 feature points on morphological places of the neutral head model are defined by MPEG-4 Facial Animation, as shown in figure 7.

The FDPs mainly serve for specifying how the face mesh will deform according to the transformation parameters (FAPs).

The second set of parameters, FAPs, on the other hand, consists of a collection of animation parameters that modify the positions of the previously defined feature points and, thus, can be used to create or change the model for each of the desired facial expressions. There are 68 FAPs that can be grouped in two categories: high-level and low-level parameters. The number of high-level parameters is only two. The first one is visemes, which are the visual equivalents to phonemes in speech. This parameter defines the mouth shapes produced by the different possible phonemes. The second high-level parameter corresponds to facial expressions and can take 6 values, one for each of the 6 archetypal emotions (anger, disgust, fear, joy, sadness and surprise). The remaining 66 low-level parameters are used for basic deformations applied to specific morphological points on the face, like the top middle outer-lip, the bottom right eyelid, etc... Because FAPs are universal parameters and independent from the head model geometry, MPEG-4 Facial Animation defines a set of 6 units, the Facial Animation Parameter Units (FAPU), to normalize the FAPs and make them independent of the overall face geometry. Prior to animation of a virtual face, the FAPs have to

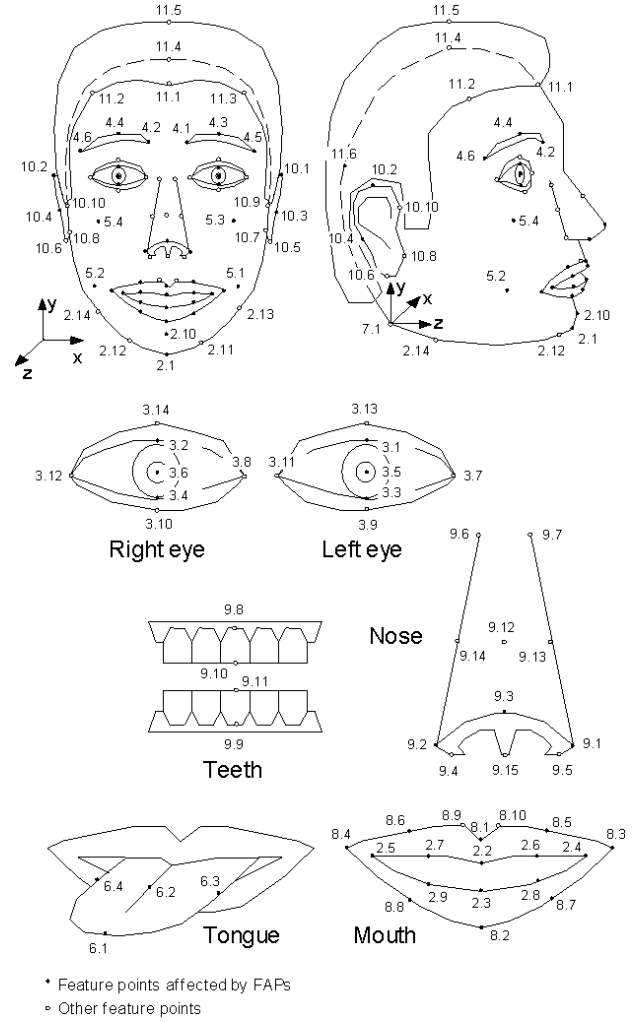


Fig. 7. Feature points are used to define the shape of a face model and the facial animation parameters are defined by motion of some of these feature points

be normalized by their corresponding FAPUs. As depicted in figure below, the FAPUs are defined as fractions of distances between key facial features (i.e. eye-to-eye distance, angular unit, etc.).

At this point, 8 quadratic global motion parameters are used to separate the local movements of key feature points from the global movement of head, since input value for each FAP is independent from other FAPs.

One can think about this scenario: if head rotates around its horizontal axis by 45 degrees, mouth will also rotate with head. In the meantime, mouth just opens a little bit, meaning that upper and lower lips move away from each other. And now, if one looks at the big picture, middle point of upper lip will seem to be moving neither just horizontally nor just vertically, but along an inclined line in between. If the deformation value for the middle point of the upper lip is calculated directly by taking the difference of the 2D pixel locations between two consecutive frames, the final value will obviously be a wrong input for the animation. Instead of this, having $(x(t), y(t))$ in frame t , for frame $t + 1$, one can estimate $(\hat{x}(t + 1), \hat{y}(t + 1))$ using global motion parameters (a_1, \dots, a_8) and $(x(t), y(t))$.

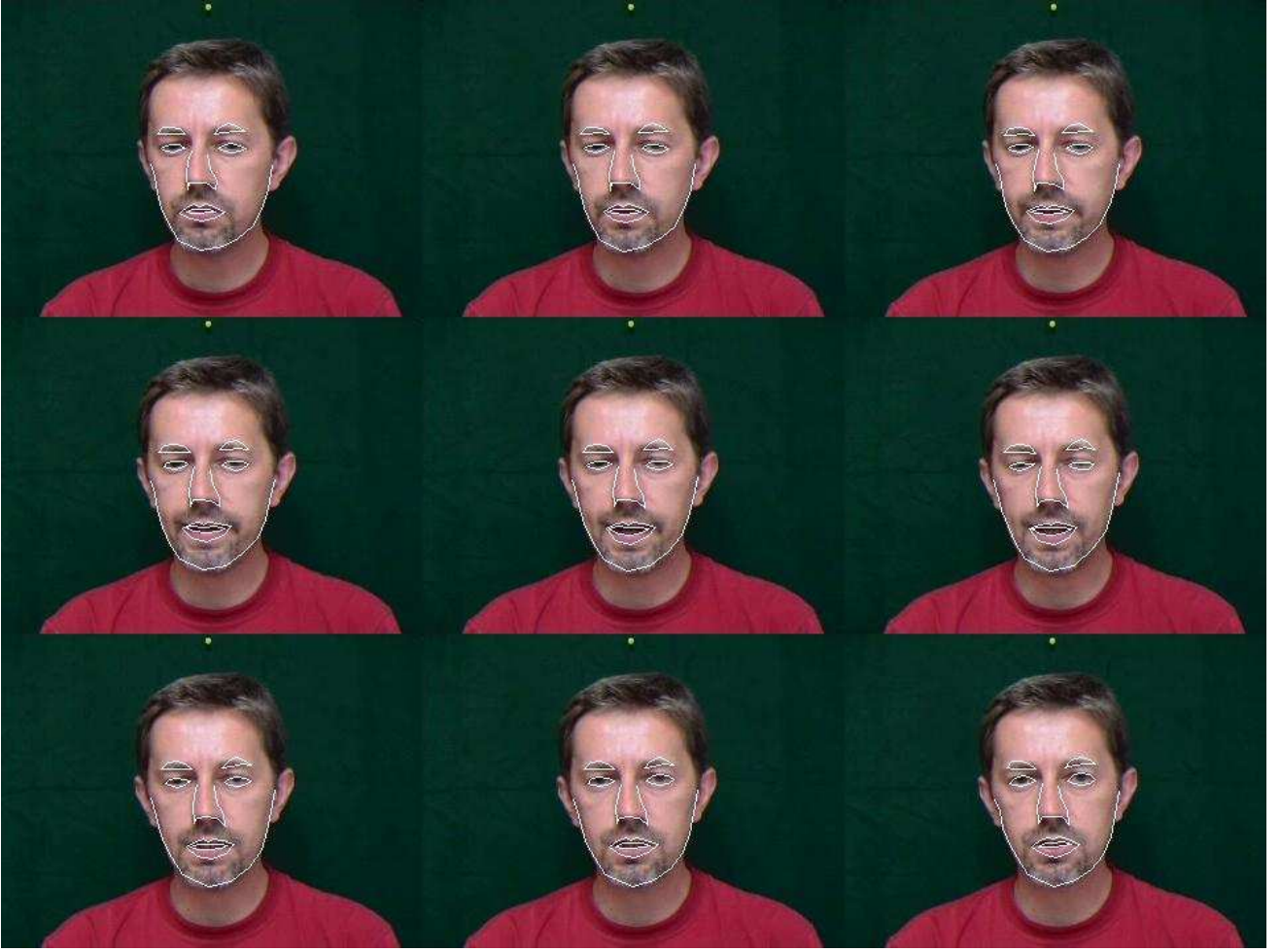


Fig. 6. Example image sequence that demonstrates the performance of tracking by AAMs

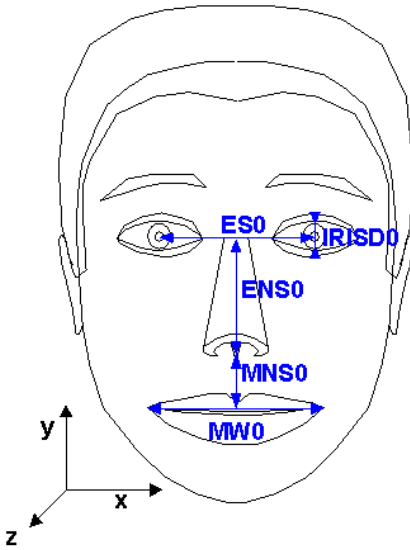


Fig. 8. Some of the feature points are used to define FAP units when the face model is in its neutral state. Fractions of distances between the marked key features are used to define FAPUs

Then, the difference between the measurement $(x(t+1), y(t+1))$ and estimate $(\hat{x}(t+1), \hat{y}(t+1))$ can be considered as the actual deformation of the model for the lips. Likewise, all the values for the used FAPs can be calculated in a similar fashion.

Besides calculating the FAP values for different face components, FAP values for the head movement itself have to be computed as three rotation angles around the three axes of the head. This is not an easy task since going from 2D world to the 3D world with only one angle of view requires sophisticated approximations, where there is no chance for accurate results at all. There are model-based pose estimation approaches using ellipsoidal models [23], or downhill simplex optimization method and the combination of motion and texture features [24], [25].

However, even though it is not as accurate as other methods, it is also possible to simply approximate the rotation angles by assuming that the head center moves around over the surface of a sphere that is centered at the top of the spinal cord. When the displacement of head is projected onto this sphere, the angles of head rotation can be estimated approximately.

IV. EXPERIMENTS AND RESULTS

In order to design the application depicted in Fig. 1, we needed to choose a source and target speaker, make sure we

could have access to a large amount of speech from the target (for the speech synthesis module), of a reasonable amount of aligned speech data for source and target (for the voice mapping module), and of some test speech+video data from the source (in order to test the complete system). We chose to use the CMU-ARCTIC databases as target speech [26]. This database was designed specifically for building unit-selection-based TTS systems. It is composed of eight voices, speaking 1150 sentences each (the same sentences, chosen to provide a phonetically balanced corpus). We thus decided to record an audiovisual complementary database, for use as the source data.

The eINTERFACE06_ARCTIC database we have created is composed of 199 sentences, spoken by one male speaker, and uniformly sampled from the CMU_ARCTIC database [26]. For each sentence, an .txt, a .avi, and a .wav files are available. The .avi file contains images with 320x240 pixels, 30 frames per second, of the speaker pronouncing the sentence ($F_s=44100$ Hz). The .wav file contains the same sound recording as in the .avi file, but resampled to 16 kHz.

The database was recorded using a standard mini-DV digital video camera. The recording of the speech signal was realized through the use of a high-quality microphone, specially conceived for speech recordings. The microphone was positioned roughly 30cm below the subject's mouth, outside the camera view.

The background consisted of a monochromatic dark green panel that covered the entire area behind the subject, to allow easier face detection and tracking. Natural lighting was used, so that some slight illumination variation can be encountered among the files (Fig. 2).

The recordings were made using the NannyRecord tool provided by UPC Barcelona, which makes it possible for the speaker to hear the sentence it has to pronounce twice before recording it. The source speaker used for the recordings were the “awb” speaker of CMU_ARCTIC. The eINTERFACE06_ARCTIC speaker was asked to keep the prosody (timing, pitch movements) of the source, while using his own acoustic realization of phonemes, and of course, his voice (i.e., not trying to imitate the target voice). This particular setting has made it possible for the eINTERFACE_ARCTIC recordings to be pretty much aligned with the corresponding CMU_ARCTIC recordings.

Following the standard approach, the parallel database was further divided into three subsets:

- *development* set, consisting of 188 sentences (of the total 198), used during the training phase of the different algorithms (alignment, voice mapping, etc.),
- *validation* set, used to avoid overfitting during the refinement of the model parameters (number of clusters, GMMs, etc.),
- *evaluation* set, used to obtain the results of the multi-modal conversion.

It is worth mentioning that this last subset (evaluation) was not present in any of the stages of the training. The results we have obtained can therefore be expected to generalize smoothly to any new data.

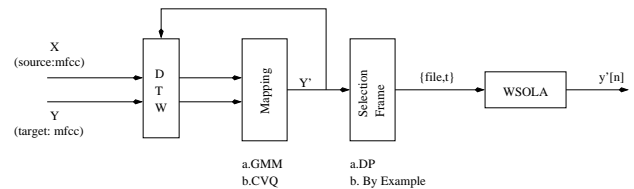


Fig. 10. Block diagram show the different alternatives for the mapping estimation and for the frame selection.

The features computed from the signals were either 13 or 20 MFCC's and their first and second derivatives; the frame rate was chosen to be 8ms. For the computation of the derivatives we relied on an implementation by Dan Ellis² and the other signal processing steps were taken from the MA toolbox by Elias Pampalk³. We also computed estimations of the fundamental frequency by using functionalities provided by the Praat software⁴. For fundamental frequency and for signal energy we can also provide first and second derivatives so that for each frame the full set of features was: 20 MFCC's, 20 Δ MFCC's, 20 $\Delta\Delta$ MFCC's, f_0 , Δf_0 , $\Delta\Delta f_0$, energy, Δ energy and $\Delta\Delta$ energy resulting in a vector of 66 dimensions.

In figure 10, we can see the different alternatives we have implemented for each of the blocks.

A. Alignment and voice mapping

After the first alignment the Euclidean distance (L_2 norm) between the source and the target MFCCs was: 1210.23. Then the GMM-based mapping was applied and the same norm was measured on the Transformed data and the Target Data: 604.35 (improvement: 50.08%). Then the Source data were transformed and a new alignment was performed. Using the new aligned data, a new mapping function was estimated and again the Source data were transformed, and again the L_2 norm between the transformed data and the Target data was measured (397.63). The process was repeated and a new measurement of the performance of the mapping function was measured using the L_2 norm (378.18).

Without iterations, we achieve a 50% reduction of distance between the target and the source data. This percentage should be higher, and provides also an information on the difference between the two speakers, and/or of the differences in the recording conditions. After some iteration we arrive to stable mapping function, with an improvement over the initial distance between the source and the target data of: 68.76%. Figure 11 shows the reduction of the distortion due to the iteration of the algorithm.

B. On clustering parameters

The incremental alignment procedure used a Gaussian Mixture Model of the source space with 128 components. This parameter remained fixed throughout the experiments. It had influence on the construction of the aligned data as well as on the mapping from source to target space, as this mapping is

²<http://labrosa.ee.columbia.edu/matlab/rastamat/deltas.m>

³<http://www.ofai.at/~elias.pampalk/ma/>

⁴<http://www.fon.hum.uva.nl/praat/>



Fig. 9. Facial excerpts from the eINTERFACE06_ARCTIC database.

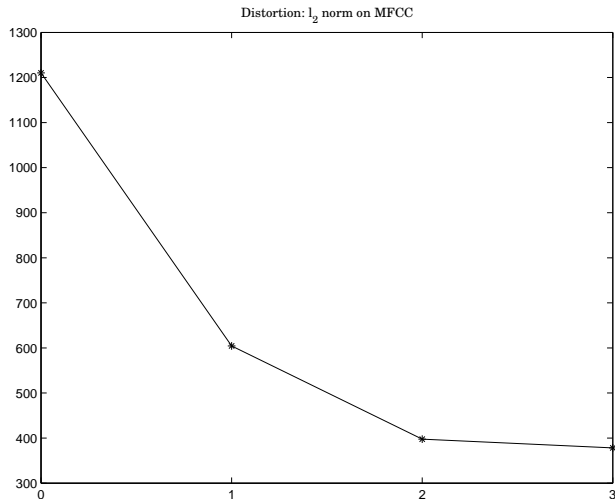


Fig. 11. Euclidean distance between the transformed data and the target data. Stability is reached after three iterations, with an overall improvement of 86%.

TABLE I
MEAN SIGNAL TO NOISE RATIO FOR VALIDATION FILE FRAMES MAPPED
TO SOURCE SPACE CLUSTERS

Number of Clusters	SNR
256	11.58 dB
512	12.04 dB
1024	12.46 dB

following equation 2 and so depends as well on the number of mixture components.

For the computation of the CVQ-based mapping the source space had to be clustered. Table I shows the average logarithmic signal to noise ratios for our five validation files. We can see that increasing the number of clusters brings improvements of about 1 dB. For the $Y|X$ codebook size eight was chosen, so that we had eight candidates for the \hat{Y} to choose from using the Viterbi algorithm.

C. Face animation

For our specific scenario, each frame in the training set has been manually labeled with 72 landmarks by using the AAM-API [27] software (freely available for non-commercial use such as research and education). The image in figure 12 shows an example of an annotated image from the training set.

In order to obtain an accurate mapping from the 2D pixel locations to the MPEG-4 parameters, the annotation of the



Fig. 12. Example of a training image labelled with 72 landmark points

images in the training set should closely match the MPEG-4 FDPs. In this particular task, the mapping process includes calculation of 6 facial animation parameters units (FAPUs), besides 44 low-level FAPs. Figure 13 shows the face model used in this work, already available in the XFace project.

We have found it necessary to smooth the calculated values for animation, since the measurements in the tracking process are noisy and small scale differences in the parameters for the simulation process may have large effects in the resulting animation. Kalman Filter was used for this purpose with a state model consisting of positions and velocities of all the key feature points on the face.

2cm

As a result of using the techniques described above, several video files have been produced from the evaluation subset of the database. These results are available for tests in the archive of our project on the eINTERFACE06 website. We created parallel videos showing source speaker and target avatar side-by-side in order to evaluate the face detection and tracking algorithm. As it can be seen in the videos, both algorithms are able to provide accurate results (although they are very sensitive to the tuning of the parameters, and in some cases they result in unreliable estimations).

Utterances using the three speech generation techniques previously explained (sec. II-D) have been generated. For comparison purposes, we have also generated an English voice for the Festival Speech Synthesis System using the full CMU ARCTIC database (except the evaluation subset). The



Fig. 13. The face model available in the XFace is used to create the desired animation

phonetic segmentation was performed automatically, and thus the resulting voice contains errors that would require manual correction. Informal tests show that the LPC-based methods (secs. II-D.1 and II-D.2) produce more natural, continuous sound speech, than the speech to speech synthesis method (sec. II-D.3). However, the speaker identity of the converted speech is closer in the later case to the target speaker. The highest identification score was obtained by the Festival voice, although the discontinuities due to the automatic segmentation seriously affect the quality of the synthetic speech.

V. CONCLUSIONS

In this paper we described a multimodal speaker conversion system, based on the simultaneous use of facial animation detection, 3D talking face synthesis, and voice conversion. A typical and important feature of the problem we have treated is that a large amount of speech data is assumed to be available for the target speaker.

During this work, two techniques have been studied for the mapping of the source voice to the target voice: Gaussian Mixture Models (GMMs) and Condition Vector Quantization (CVQ). Several alternatives for the synthesis block have been implemented, without achieving acceptable quality. Preliminary analysis of the results indicate that discontinuities in the phase are causing major distortion in the synthetic speech. Possible solutions and directions for future research are given below.

A. Enhanced multimodal approach

Of the two conversion problems mentioned in the introduction of this paper (voice and facial movements), we have

mostly focused on the first, and assumed the second merely reduced to a scaling of source to target movements. Clearly, each human face has its own ways of producing speech, so that the facial movement conversion step could still be widely enhanced by submitting it to a more complex mapping stage.

One immediately notices that the approach followed here is only weakly multimodal: it actually uses late multimodal fusion. One of the obvious ideas that could be exploited in the future is that of using speech for face movement detection, and possibly face movements for helping voice mapping. A further step could then be to study simultaneous face and voice conversion. This would require having video data for the target (which we did not have here). The speech-to-speech component of our project could then be made multimodal in essence, by using facial movements in the definition of target and concatenation costs, for instance.

B. Weighted Euclidean distance

Until now, we have not actually used the weighted Euclidean distance, instead we simply used the basic Euclidean distance. However methods exist that allow the computation of optimal weights for such a distance. A first future improvement of our results could be to apply one of those methods.

Another way to improve this part of the system could be to compute the target and concatenation cost with other measurements such as Mahalanobis, Kullback-Leibler or Itakura distance.

C. F_0 mapping and continuity

Presently, the target speech is synthesized using the residual excitation from the source speaker. Therefore the utterance has the same prosody than the source. This is a major drawback because we think this is the main reason why the final output sounds as a third speaker situated between the source and the target.

The next step in the development of an efficient open source voice conversion system should be to create a mapping function between the two speaker's prosodies.

D. Phase continuity

In order to reduce the influence of the source speaker's voice in the final result, we would like not to use his residual excitation anymore. Indeed, this residual still contains a lot of information about him.

The OLA solution proposed in section II-D.3 can be a solution. However, this will introduce a lot of phase discontinuity (aside from the energy and pitch discontinuities which can be handled more easily). We have found no elegant solution to this problem, which requires further study

E. Pitch synchronous processing

A different approach could be to use a PSOLA algorithm to achieve the resynthesis. If PSOLA is used in the synthesis step, then the absolute value of the pitch is of lesser importance as a target for the Viterbi alignment, however its delta and delta-delta are still important. Indeed PSOLA can change the overall pitch easily, but delivers lower quality speech when the shape of the pitch curve is modified.

F. 3D Head Synthesis

While we have completed the face detection, movement tracking, and 2D to 3D mapping of this project, we did not have time to work on the 3D head synthesis aspects. In this part, the main task is to convert the set of parameters into visual animations. Animations are created using XFace interface which is an MPEG-4 based open source toolkit for 3D facial animation, developed by Balci [28]. This is a straightforward process: once the FAP values have been computed, the XFace editor (or any other suitable player) can directly synthesize them (given that the parameters file format is correct).

G. Software notes

As a result of this project, the following resources are available:

- eNTERFACE06_arctic database (video and audio). 199 sentences sampled from the CMU_ARCTIC database⁵. The video part is recorded in *avi* files (320x240 pixels), 30 frames per second, unencoded. The audio part is recorded in *wav* files, 16kHz, 16 bits per sample, unencoded.
- Conditional Vector Quantization algorithm (Matlab).
- Clustering via *K*-means algorithm.
- Viterbi algorithm (Matlab and C++)
- OLA implementations, with optional correlation-based correction of the window positions.
- Automatic calculation of global head motion parameters.

ACKNOWLEDGMENT

We want to acknowledge the CMU-ARCTIC team at CMU, for having made large speech corpora available to the research community.

We are also grateful to Prof. Antonio Bonafonte, who collaborated to the initial steps in the project design, but could finally not attend eNTERFACE'06.

We also want to thank François-Xavier Fanard, who was part of Project 3, and helped us in the design of the 3D avatar. Our thanks also to Yannis Agiomyrgiannakis for kindly providing the CVQ code and technical support.

We are also grateful to Prof. Alice Caplier (INPG Grenoble) for letting us access the facial tracking system developed at INPG, even though we have not used it in the final version of our speaker conversion system.

REFERENCES

- [1] Y. Stylianou, O. Cappé, and E. Moulines, "Continuous probabilistic transform for voice conversion," *IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING*, vol. 6, no. 2, pp. 131–142, 1998.
- [2] Y. Agiomyrgiannakis and Y. Stylianou, "Conditional vector quantization for speech coding, to be published," Institute of Computer Science, Foundation of Research & Technology Hellas, Tech. Rep., 2006.
- [3] T. Dutoit, *An Introduction to Text-To-Speech Synthesis*. Dordrecht: Kluwer Academic Publishers, 1997.
- [4] L. Rabiner, "A tutorial on hmm and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, February 1989.
- [5] T. Dutoit and M. Cernak, "Ttsbox: A matlab toolbox for teaching text-to-speech synthesis," in *Proc. ICASSP'05*, Philadelphia, USA, 2005.
- [6] D. Suendermann, H. Hooge, A. Bonafonte, H. Ney, A. Black, and S. Narayanan, "Text-independent voice conversion based on unit selection," in *Proc. ICASSP06*, Toulouse, 2006, pp. 81–84.
- [7] A. Gersho and R. Gray, *Vector quantization and signal compression*. Norwell, Massachusetts: Kluwer Academic Publishers, 1992.
- [8] D. Chazan, R. Hoory, Z. Kons, D. Silberstein, and A. Sorin, "Reducing the footprint of the ibm trainable speech synthesis system," in *ICSLP*, 2002.
- [9] W. Verhelst and M. Roelands, "An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech," in *Proceedings of ICASSP-93*, vol. 2, 1993, pp. 554–557.
- [10] *ISO/IEC IS 14496-2 Visual*, 1999.
- [11] P. Eisert, "MPEG-4 facial animation in video analysis and synthesis," *International Journal of Imaging Systems and Technology*, Springer, vol. 13, no. 5, pp. 245–256, March 2003, see also: http://ip.hhi.de/comvision_G2/expressions.htm.
- [12] M. E. Sargin, F. Ofli, Y. Yasinnik, O. Aran, A. Karpov, S. Wilson, E. Erzin, Y. Yemez, , and A. M. Tekalp, "Combined gesture-speech analysis and synthesis," in *Proc. of the eNTERFACE'05 The SIMILAR Workshop on Multimodal Interfaces*, August 2005.
- [13] T. F. Cootes, G. J. Edwards, , and C. J. Taylor, "Active appearance models," in *Proc. European Conf. on Computer Vision*, vol. 2, 1998, pp. 484–498.
- [14] —, "Active appearance models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 681–685, 2001.
- [15] G. J. Edwards, C. J. Taylor, , and T. F. Cootes, "Interpreting face images using active appearance models," in *Proc. 3rd IEEE Int. Conf. on Automatic Face and Gesture Recognition*. IEEE Comput. Soc., 1998, pp. 300–305.
- [16] T. F. Cootes and C. J. Taylor, "Statistical models of appearance for medical image analysis and computer vision," in *Proc. SPIE Medical Imaging*, San Diego, CA, 2001, pp. 236–248.
- [17] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. 1991 IEEE Com. Soc. Conf. on CVPR*. IEEE Com. Soc. Press, 1991, pp. 586–591.
- [18] T. F. Cootes and C. J. Taylor, "Active shape models smart snakes," in *Proc. British Machine Vision Conf., BMVC92*, 1992, pp. 266–275.
- [19] T. F. Cootes, C. J. Taylor, D. Cooper, , and J. Graham, "Training models of shape from sets of examples," in *Proc. British Machine Vision Conf., BMVC92*, 1992, pp. 9–18.
- [20] R. Fisker, "Making deformable template models operational," Ph.D. dissertation, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, 2000.
- [21] M. Tekalp, "Face and 2d mesh animation in mpeg-4," *Tutorial Issue On The MPEG-4 Standard, Image Communication Journal*, Elsevier, 1999.
- [22] I. Pandzic and R. Forchheimer, *MPEG-4 Facial Animation : The standard, implementation and applications*. Wiley, 2002.
- [23] S. Basu, I. Essa, and A. Pentland, "Motion regularization for model-based head tracking," in *Intl. Conf. on Pattern Recognition (ICPR '96)*, Vienna, Austria, 1996.
- [24] F. Preteux and M. Malciu, "Model-based head tracking and 3d pose estimation," in *Proceedings SPIE Conference on Mathematical Modeling and Estimation Techniques in Computer Vision*, vol. 3457, San Diego, CA., July 1998, pp. 94–110.
- [25] M. Malciu and F. Preteux, "A robust model-based approach for 3d head tracking in video sequences," in *Proc. of 4th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000)*, Grenoble, France, March 2000, pp. 26–30.
- [26] J. Kominek and A. W. Black, "CMU ARCTIC databases for speech synthesis," Carnegie Mellon University, Tech. Rep., 2003, <http://festvox.org/cmu-arctic/>.
- [27] M. B. Stegmann, B. K. Ersboll, and R. Larsen, "Fame - a flexible appearance modeling environment," in *IEEE Transactions on Medical Imaging*, vol. 22, no. 10, October 2003, pp. 1319–1331.
- [28] K. Balci, "Xface: Mpeg-4 based open source toolkit for 3d facial animation," in *Proc. Advance Visual Interfaces*, 2004, pp. 399–402.

⁵Full database is available at: <http://www.festvox.org>



Thierry Dutoit graduated as an electrical engineer and Ph.D. in 1988 and 1993 from the Faculté Polytechnique de Mons, Belgium, where he is now a full professor.

He spent 16 months as a consultant for AT&T Labs Research in Murray Hill and Florham Park, NJ, from July, 1996 to September, 1998. He is the author of two books on speech processing and text-to-speech synthesis, and the coordinator of the MBROLA project for free multilingual speech synthesis.

T. Dutoit was an Associate Editor of the IEEE Transactions on Speech and Audio Processing (2004-2006) and is a member of the INTERSPEECH'07 organization committee.



André Holzapfel graduated as an electrical engineer in 2004 from the University of Applied Sciences Duesseldorf, Germany, and will complete work on his Master Thesis at the Graduate School University of Crete in September.

He has served Creamware Datentechnik GmbH from 2002 until 2003 to develop dynamic models for electronic tubes on DSP platforms. His special interest is information retrieval from musical signals.



Matthieu Jottrand holds an Electrical Engineering degree from the Faculté Polytechnique de Mons since June 2005. He did his master's thesis in the Image Coding Group of Linköping Institute of Technology. Matthieu is a researcher from TCTS lab (FPMs) since September 2005.

He is currently working in the field of ASR for the IRMA project (development of a multimodal interface to search into indexed audiovisual documents) and just started a PhD thesis under the supervision of Thierry Dutoit.



Ferran Marqués is Professor at Technical University of Catalonia (UPC), Department of Signal Theory and Communications, Barcelona, Spain. He received a degree on Electrical Engineering in 1988, and the Ph.D. degree in 1992, both from the UPC.

From 1989 to 1990, he joined the Digital Image Sequence Processing and Coding Group at the Signal Processing lab. of the Swiss Federal Institute of Technology (EPFL). In June 1990, he joined the Department of Signal Theory and Communications of the Technical University of Catalunya (UPC).

From June 1991 to September 1991, he was with the Signal and Image Processing Institute (SIPI) at the University of Southern California (USC). In 1993 he joined the Technical University of Catalonia where he is currently Professor.

He has been President of the European Association for Signal, Speech and Image Processing (EURASIP) in the term 2002-2004. He served as Associate Editor of the Journal of Electronic Imaging (SPIE) in the area of Image Communications (1996-2000), as member of the EURASIP Journal of Applied Signal Processing Editorial Board (2001-2003) and of the Hindawi International Journal of Image and Video Processing (2006-).



Alexis Moinet holds an Electrical Engineering degree from the FPMs (2005). He did his master thesis at the T.J. Watson research Center of IBM (2005). He is currently working on the IRMA project in the Signal Processing Lab of the FPMs. He is particularly interested in glottal source/vocal tract decomposition of speech and music, phase vocoder, voice activity detection, voice conversion and HMM synthesis.



Ferda Ofli received B.Sc. degree in Electrical and Electronics Engineering, and B.Sc. degree in Computer Engineering from Koç University, Istanbul, Turkey in 2005. He is currently a M.Sc. student in Electrical and Computer Engineering Department and a member of Multimedia, Vision and Graphics Laboratory at Koç University.

He is currently taking part in European projects, SIMILAR NoE and 3DTV NoE. His research interests include image and video processing, specifically, object segmentation and tracking, human body modeling, motion capture and gait/gesture analysis.



Javier Pérez Mayos received his M.S degree in Electrical Engineering from the Royal Institute of Technology (KTH), Stockholm, Sweden, in 2001, and from the Technical University of Catalonia (UPC) in May 2002.

Since May 2002, he is a PhD student at UPC. His research topic is voice source analysis and characterization. The objective is to be able to use this information in voice generation algorithms, so applications like emotional and expressive synthesis, and voice conversion, can benefit from his research.

He has participated in several international speech-to-speech translation projects (LC-STAR, TC-STAR) and has released Gaia, a research-oriented speech-to-speech translation architecture.



Yannis Stylianou is Associate Professor at University of Crete, Department of Computer Science. He received the Diploma of Electrical Engineering from the National Technical University, NTUA, of Athens in 1991 and the M.Sc. and Ph.D. degrees in Signal Processing from the Ecole Nationale Supérieure des Telecommunications, ENST, Paris, France in 1992 and 1996, respectively.

From 1996 until 2001 he was with AT&T Labs Research (Murray Hill and Florham Park, NJ, USA) as a Senior Technical Staff Member. In 2001 he

joined Bell-Labs Lucent Technologies, in Murray Hill, NJ, USA. Since 2002 he is with the Computer Science Department at the University of Crete.

He was Associate Editor for the IEEE Signal Processing Letters from 1999 until 2002. He is Associate Editor of the EURASIP Journal on Speech, Audio and Music Processing. He was served on the Management Committee for the COST Action 277: "Nonlinear Speech Processing" and he is one of the two proponents for a new COST Action on Voice Quality Assessment. He holds 8 patents. Prof. Stylianou participates in the SIMILAR Network of Excellence (6th FP) coordinating the task on the fusion of speech and handwriting modalities.

Introducing Network-Awareness for Networked Multimedia and Multi-modal Applications

Miran Mosmondor^{*}, Ognjen Dobrijevic⁺, Ivan Piskovic⁺, Mirko Suznjevic⁺, Maja Matijasevic⁺, Sasa Desic^{*}

^{*}*Ericsson Nikola Tesla, Research and Development Center, Zagreb, Croatia
{miran.mosmondor, sasa.desic}@ericsson.com*

⁺*University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, Croatia
{ognjen.dobrijevic, ivan.piskovic, mirko.suznjevic, maja.matijasevic}@fer.hr*

Abstract - Due to increased user/service requirements in terms of network quality of service (QoS) parameters, and heterogeneity of end-user access network options and terminal capabilities, introducing “network-awareness” into rich multimedia and multimodal networked applications could provide a critical advantage. An idea behind network-awareness is to let the applications indicate their requirements and to adapt to changing conditions in the network, as well as to let the network “know” of the applications’ resource demands. This approach is based on signaling, as a means to request special treatment for traffic in the network and to receive indications from the network of different conditions. Another important issue for the proposed solution is the simplicity of use. Providing developers with a reusable solution that, to much extent, removes the need for understanding a specific signaling protocol eases and quickens development of the network-aware applications. The project objective was to identify generic signaling functionality, and to create an application programming interface (API) which will enable application developers to create advanced multimodal networked services. The developed API was applied in a case study using a prototype application.

Index Terms - Application Programming Interface, Dynamic service adaptation, End-to-end Quality of Service signaling, Multimedia and multimodal networked applications, IP Multimedia Subsystem

I. INTRODUCTION

With ever more widespread multimedia and multimodal end-user equipment, ranging from devices specifically designed for a particular purpose to generic laptops and mobile phones, a wide range of new services may be envisioned to provide better quality of life, especially for the elderly and the disabled. Examples of such services include universally (“anywhere-anytime”) accessible and context-adaptive information services, medical monitoring and counseling services, and edutainment services based on collaborative virtual environments (CVE) [1]. A CVE may include various means of communication between its participants, including, but not limited to face and body gestures and behavior (performed via users’ representation in

the virtual world, or, avatar), text chat, and, possibly, live voice communication. Further on, adaptation of the content presented to the user may be required in more than one way, taking into account the user's preferences, experience, and (dis)ability, as well as user's terminal capabilities/features, and network conditions. The interdependence of these requirements may be addressed through the “application aspect” and the “communication aspect” [12].

From the network point of view, such applications involving rich multimedia content and real-time interaction impose more strict requirements onto managing, delivering, and monitoring network performance. For example, a too long delay in service response or inability to adapt the content to terminal characteristics may render the service useless. In this work, we are particularly interested in services with multimodal information being exchanged not only at the advanced human-computer interface, but also being transferred through the network. Such services need to go beyond the traditional approach (Fig. 1), where the quality guaranteed by the network is either predefined (e.g. voice quality in fixed telephony), or taken as random (e.g. delay in Web browsing), to a more “network-aware” approach. This means that a certain “control” component is needed at both the client and the service “ends”, which is capable of exchanging control information, or signaling, as illustrated in Fig. 2 below.

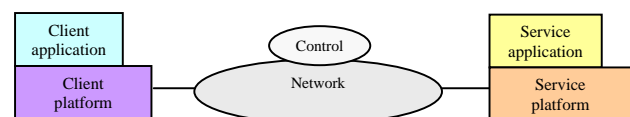


Figure 1. Traditional approach - network assumes predefined behavior

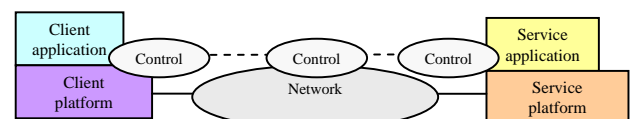


Figure 2. “Network-aware” approach - context and network adaptation

A. Dynamic service adaptation model

A dynamic service adaptation model (DSAM) has been proposed in our earlier work (more details in [10]), which takes into account the heterogeneity of access options and advanced multimedia services in next generation networks, and attempts to further describe and specify the (sets of) parameters referring to:

- end-user access network options and terminal capabilities (client platform)
- user preferences (client application, human-computer interface, personal preferences and/or (dis)ability)
- available resources and costs (network)
- service requirements (server application, server platform)

The proposed model focuses on the provisioning of end-to-end support for signaling QoS requirements at the session layer with the emphasis on virtual reality (VR) services. It includes the entire process of negotiation and renegotiation of QoS parameters, and service adaptation, from when an end-user accesses a VR service until (s)he terminates it. The model (shown in Fig. 3) is centered on a process in a client server architecture in which a client accesses an application server that hosts the service, and consists of a set of functionalities that are logically combined into three entities: *Client*, *Access and Control*, and *Application Server*.

After a user has initiated request for a specific VR service, the *Client* passes it to the *Access and Control* entity, specifying terminal capabilities and user preferences in a

“client profile”. A client profile incorporates user preferences such as acceptable service format(s) and maximum download time, terminal hardware and software, and access network characteristics. The *Access and Control* entity represents a group of service control and management functionalities, and is responsible for identifying the client, authorizing requested network resources, and negotiation of QoS parameters for the service.

The *QoS negotiation and control* (QNC) receives the client request and invokes the *Profile Manager* (PM). The PM retrieves “service profiles” describing various service configurations for the requested service and matches parameters of the service profiles with constraints of the client profile in order to determine achievable service configurations. A service configuration is assumed achievable when: (1) a user’s terminal capabilities are able to support the requested service processing requirements; (2) the user’s access network is able to support the minimum network requirements for all required media elements; and (3) the user’s preferences in terms of desired media elements and acceptable download time can be met.

After the matching process, the PM extracts a set of potential session parameters (i.e. media formats and codec types) from service configurations that are feasible and forwards it to the QNC. The QNC sends offered session parameters to the *Client*, which in return indicates the subset of offered parameters it agrees to. Network entities authorize resources based on the agreed subset of parameters.

The returned parameter subset is sent back to the PM, which then orders the achievable service configurations according to quality based on user perceived quality. Quality of the achievable service configurations is influenced by user preferences (i.e. a user considers video to be of more importance than audio), and different configuration can be used if service degradation or upgrading is required. The service profile with the highest quality configuration is sent to the *QoS Optimization Process* (QOP).

The QOP determines the optimal service operating point and resource allocation taking into account constraints related to service requirements, terminal capabilities and user preferences, and network resource availability and cost. By the service operating point we assume the final configuration of the VR service (included media elements, associated media formats and codec types, etc.) that is to be delivered to the user.

After determination of the service operating point and resource allocation, the QOP sends the final service configuration profile to the QNC of the *Application Server*. The *Application Server* passes the final service profile to the *Client*. In addition, reservation of network resources is invoked.

The *Application Server* is responsible for retrieval and adaptation of hosted VR service based on the calculation carried out by the *Access and Control*. The QNC of the *Application Server* receives the final profile and sends it to the *VR service processor*. If necessary, service content is adapted, after which the *VR service processor* delivers it to the user. A generic sequence diagram of session establishment is shown in Fig. 4.

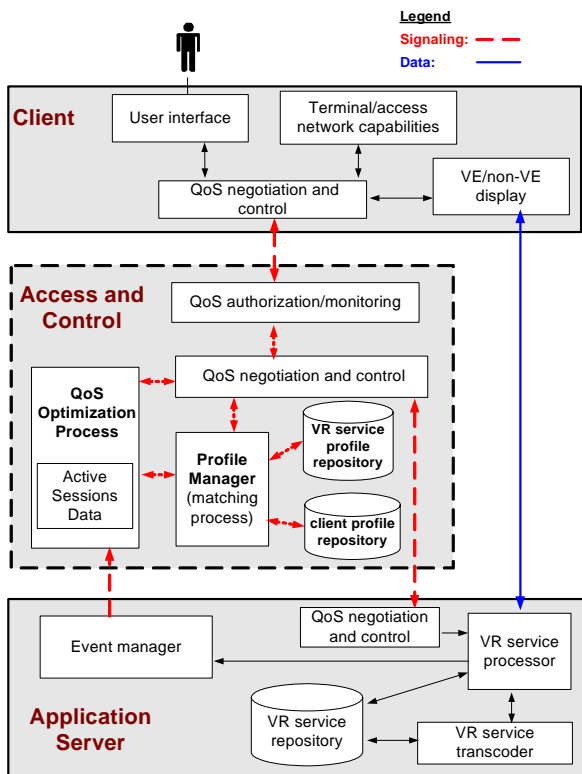


Figure 3. Model for dynamic negotiation and adaptation of QoS

A user's interest in particular virtual environment objects changes dynamically. Depending on provided interactions, a user may, for instance, choose to start video streaming. If an important change in user's interest occurs, a need to determine the new service operating point and reallocate network resources may arise in order to meet new service requirements. A user interaction, or a change perceived by the service itself cause an event to be sent from the service to the *Event Manager* (EM). Using received events, the EM informs the QOP of new service conditions.

Negotiation/adaptation and optimization procedures are invoked throughout the service lifetime in response to significant network conditions and changes occurring in service requirements and constraints like network resource availability, network resource cost, and the client profile. Each of the parties involved - the client side, the server side, and the network - respond to dynamic changes in the system.

Three scenarios are specifically addressed:

- Changes in service requirements refer to addition or detraction of application components (for instance, starting or stopping video and audio streaming) which result in signaling, among rest, reservation or

release of network resources.

- Changes in client profile refer to variations in any client profile parameter (user terminal hardware or software characteristics, access network characteristics, user preferences) and are simulated by sending new client profile versions from the client side.
- Changes in resource availability refer to variations of authorized network resources and result in signaling new conditions to the end-points.

B. Dynamic service adaptation model implementation

While the proposed model is independent of the particular network scenario, its applicability is of particular interest in the 3GPP's (3rd Generation Partnership Project) IP Multimedia Subsystem (IMS) [7], a key path to providing the converged next generation network architecture. An implementation of the model was developed by mapping the DSAM model entities to different nodes of the IMS architecture (more details in [11]). For each of the conditions (session establishment, change in client profile, change in service requirements, and change in resource availability) covered by

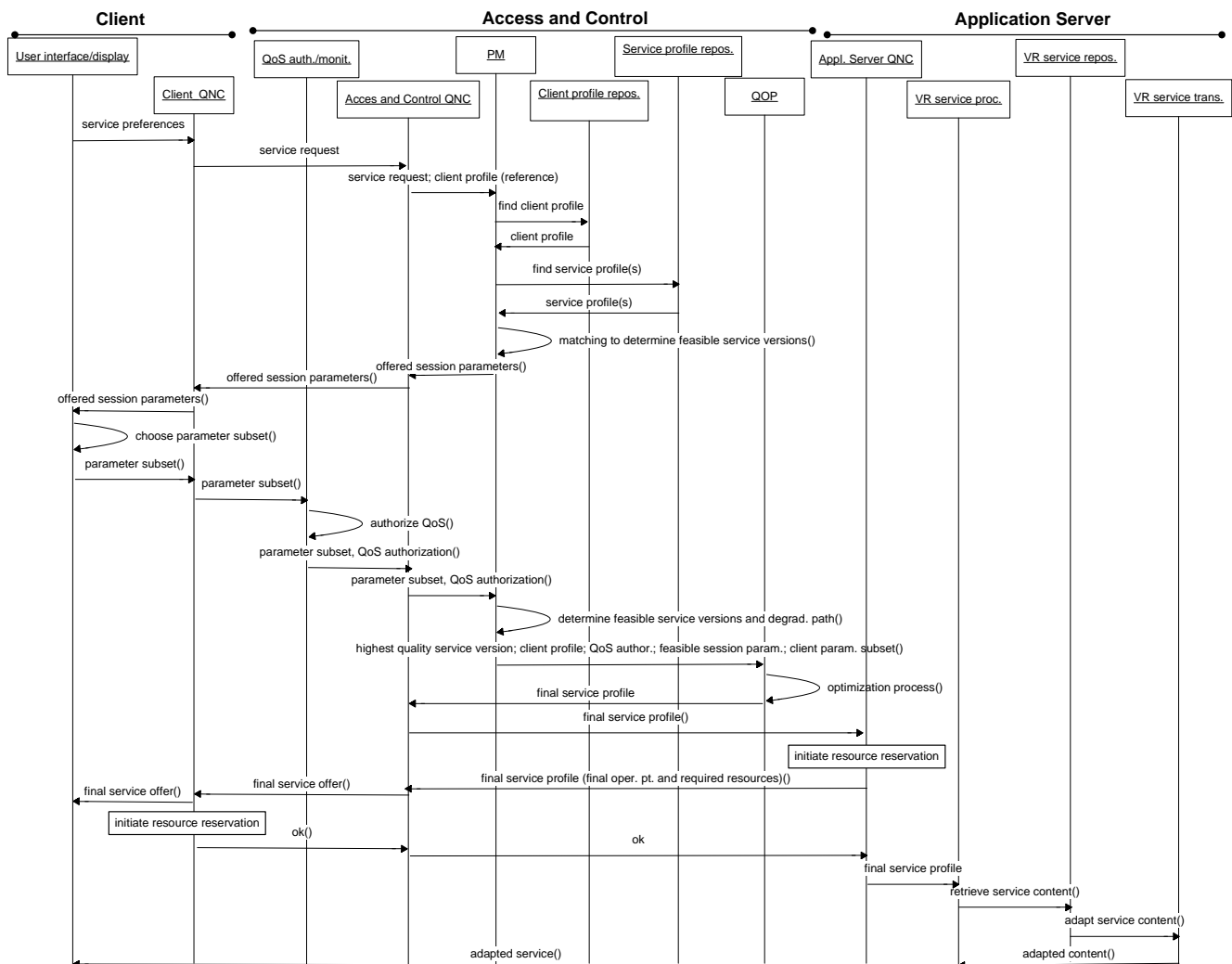


Figure 4. Generic sequence diagram for initial session establishment

the model a specific signaling scenario, that includes exchange of signaling messages between involved parties, has been defined according to the 3GPP specifications [4], [8], [9]. End-to-end signaling is preformed using widely adopted IETF's Session Initiation Protocol (SIP) [2] that, in our case, is used to exchange XML-based client and service profiles. Implementation of this signaling functionality will be used as the basis for API development.

II. DYNAMIC SERVICE ADAPTATION API

The goal of this project was to identify generic signaling functionality for application network-awareness and "fold" it into an API to be used by various multimedia and multimodal applications. The API, named Dynamic Service Adaptation (DSA) API, was designed with client/server architecture in mind meaning that one part of the API is to be used on the client side (DSA Client API related to client application with client platform) and the other part is to be used on the side of an application hosting the service(s) (DSA Server API related to server application with server platform), as shown in Fig. 5.

By using developed API, the application developers should be shielded from the signaling protocol specifics. The functionality of the API covers signaling service requirements (in our case service profile), client characteristics (in our case client profile) and final service configuration during session establishment (service invocation) and session update (service run-time phases) by exchanging messages, and capability of receiving notifications of various events that are related to changing conditions. Session update capability is initiated in response to changes occurring in service requirements, network resource availability and/or costs, and client capabilities - scenarios already referred to as change in service requirements, change in client profile, and change in resource availability. The effects of signaling may include network-aware service adaptation in response to varying conditions, as well as adequate network response to client and service requirements, with the overall goal of providing a better service to the user.

Fig. 6 portrays the building blocks of DSA API described hereafter.

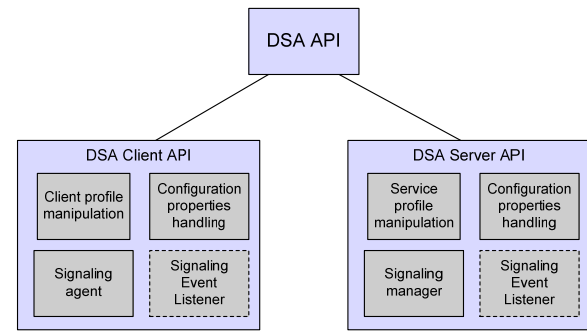


Figure 6. DSA API architecture

1) DSA Client API

For the client part of DSA API several high-level functionalities were abstracted. Most importantly, the client part should handle all the signaling with involved parties in terms of exchanging signaling messages. This includes sending client profiles and session descriptions, as well as receiving notifications of events occurring in the network or at the server side. Furthermore, API implementation should ease client profile manipulation. With these requirements in mind, following modules were identified (Fig. 7):

- Signaling agent
- Signaling event listener
- Client profile handler

Signaling agent entity is the most important part of the Client API, responsible for handling all signaling messages. Several methods were identified as mandatory for this signaling capability. First one is *establishSession()* which initiates the signaling exchange with other network entities involved. As an input argument it should receive client profile description in a XML format that includes definition of client preferences and capabilities. Analogue to this method, the session can be terminated at any time by calling the *terminateSession()* method. The *changeInClientProfile()* method models a scenario when a change in client profile

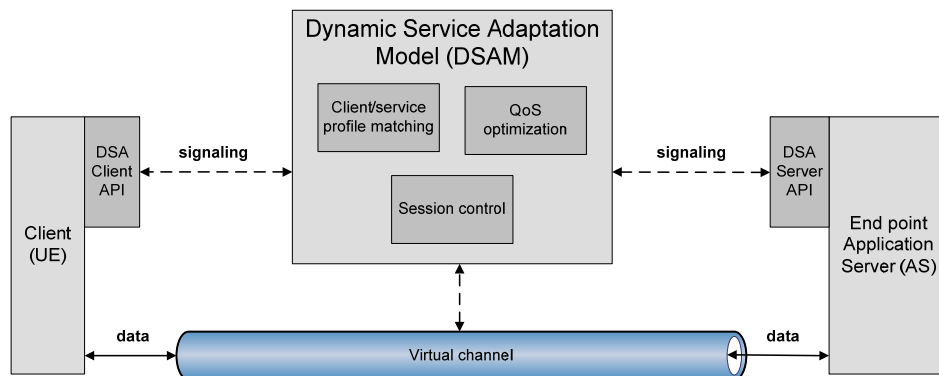


Figure 5. DSA API embedded in DSAM model architecture

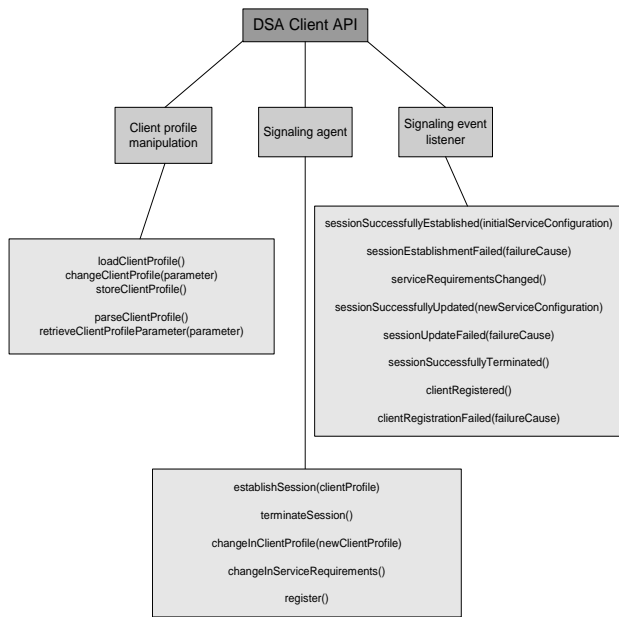


Figure 7. DSA Client API specification

occurs, for instance due to change of access network or user preferences. Another method, the *changeInServiceRequirements()*, covers a scenario when the client side initiates change in service requirements related to signaling release of network resources reserved for (a) particular service component(s). If a user is required to register to use the network services, registration process is invoked by calling the *register()* method.

Signaling event listener entity is responsible for receiving events related to signaling progress. This includes basic notifications on state of session establishment (*session successfully established* and *session establishment failed*), session update (*session successfully updated* and *session update failed*), change in service requirements/signaling release of network resources (*service requirements changed*), and session termination (*session successfully terminated*), regardless of which entity initiated the signaling. Additionally, the client side can be notified of the registration process (*client successfully registered* and *client registration failed*).

Client profile manipulation module is responsible for the client profile creation and modification.

2) DSA Server API

DSA Server API was intended to provide applications with the means to specify service requirements and changes thereof in response to various user demands and network conditions. As specifying service requirements is done in terms of the service profile, this assumes the service parameters to be specified in a standard format after which they are embodied in the signaling messages and delivered to other entities.

Basic requirements of the DSA Server API included signaling capability, based on the proposed signaling functionality, and ability to receive and properly interpret

indications from the network. Signaling capability refers to building blocks and methods that handle signaling specifics based on the proposed message flow diagrams. Indications from the network are based on the signaling progress in a particular scenario, and are meant to signal various network conditions of interest to application (changes in user preferences, capabilities of user terminals, access network conditions and network resource availability). Specification of the DSA Server API is shown in Fig. 8.

Signaling manager entity is directly associated with the signaling capability that is able to manage many clients (users). Its functionality takes care of processing and/or sending proper signaling message depending on developing network conditions or service requirements. Besides defining methods for starting and shutting down this entity, a method for handling changing service requirements in terms of signaling new service configurations has to be modeled. The latter only handles the case where application initiates signaling between involved parties, the rest is managed automatically.

Signaling event listener entity is, analogously to the client side, related to receiving events associated to signalization progress and, through it, to varying network conditions. *Session successfully established* and *session establishment failed* are to receive events specific to setting up a session between an application and a client. This process precedes initial service retrieval. *Session successfully terminated* manages events specific to session termination. *Session successfully updated* and *session update failed* are to handle events specific to session update. These events arise in response to changing network conditions and/or service requirements after session establishment. Any service component needs network resources to be reserved in order to be delivered to a user. As the signaling functionality assumes signaling network (transport) QoS requirements to underlying network entities in order to reserve necessary resources, *reserved network resources released* event has been introduced in order to indicate the release of those resources.

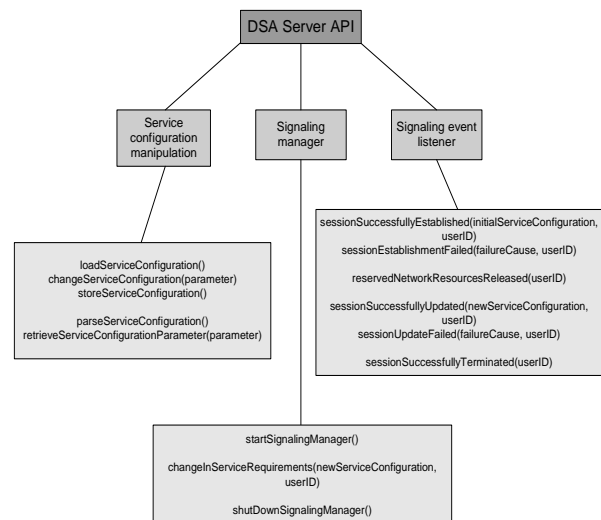


Figure 8. DSA Server API specification

Service configuration manipulation manages service configuration processing in terms of retrieving (*retrieve service configuration parameter*) or changing (*change service configuration*) a particular configuration parameter.

III. DSA API IMPLEMENTATION

DSA API Reference Implementation (RI) is based on the NIST-SIP API [3] and the 3GPP specifications [4], [8], [9] providing SIP signaling mechanisms and specifics.

A. DSA Client API RI

Reference implementation of DSA Client API relies on two Java packages:

- *hr.fer.tel.nims.dsa.client*, and
- *hr.fer.tel.nims.dsa.client.clientprofilehandler*.

First package (Fig. 9) contains *SignalingAgent* and *SignalingAgentException* Java classes, and the *SignalingEventListener* Java interface. *SignalingAgent* corresponds to the *Signaling agent* and is initialized with the reference to an implementation of the *SignalingEventListener* Java interface and a path to the configuration properties file. Methods implemented in the *SignalingAgent* are used for establishing, updating, and terminating the session, as described in the previous section. The *SignalingAgentException* was introduced in order to notify the client with a description of a problem related to the signaling.

The *SignalingEventListener* Java interface was modeled according to the *Signaling event listener* entity. An implementation of the *handleSessionEstablishedEvent()* method is notified of successful session establishment with a server hosting the service, and passed an initial service configuration for the session. An implementation of the *handleSessionEstablishmentFailedEvent()* method is notified of a failure during session establishment and passed a description of a failure cause. Event describing successful session termination is delivered by calling an implementation of the *handleSessionTerminatedEvent()* method. In response to changing network conditions, session has to be updated.

The *handleSessionUpdatedEvent()* method notifies of successful update of the session by delivering new service configuration, while the *handleSessionUpdateFailedEvent()* method delivers a description of a failure cause along. The *handleServiceRequirementsChangedEvent()* method implementation is notified of network resource release for (a) particular service component(s). Methods related to registration process are implemented according to the behavior model described in the previous section.

Package *hr.fer.tel.nims.dsa.client.clientprofilehandler* contains functionalities for handling the client profile specifics. Main Java class for managing both client and service profiles is the *ProfileParser* which implements *ProfileInterface* Java interface. It comprises methods for managing profiles: *getParameter()*, *addParameters()*, *editParameters()*, and *getParameterValue()*. Parsing the profiles is done using the Simple API for XML (SAX) parser [6].

B. DSA Server API RI

Following API specification, reference implementation relies on three Java packages:

- *hr.fer.tel.nims.dsa.server*,
- *hr.fer.tel.nims.dsa.server.eventlistener*, and
- *hr.fer.tel.nims.dsa.profilemanipulation*.

First package contains *SignalingManager* Java class (Fig. 10) that corresponds to the *Signaling manager* entity. Its constructor is initialized with the configuration properties file and the reference to an implementation of the *SignalingEventListener* Java interface. The *startSignalingManager()* method starts, while the *shutdownSignalingManager()* method terminates the components of the manager. As explained previously, the *changeInServiceRequirements()* method initiates signaling new service requirements in terms of new service configurations.

The *hr.fer.tel.nims.dsa.server.eventlistener* defines *SignalingEventListener* Java interface (Fig. 10) that was modeled with the *Signaling event listener* entity. An implementation of the *handleSessionEstablishedEvent()* method is notified of successful session establishment with a particular client and passed an initial service configuration for the session. An implementation of the *handleSessionEstablishmentFailedEvent()* method is notified of a failure during session establishment with a particular client and passed a description of a failure cause. Event describing successful session termination with a particular client is delivered by calling an implementation of the *handleSessionTerminatedEvent()* method. In response to changing any network condition, session has to be updated. The *handleSessionUpdatedEvent()* method notifies of successful update of the session with a particular client by

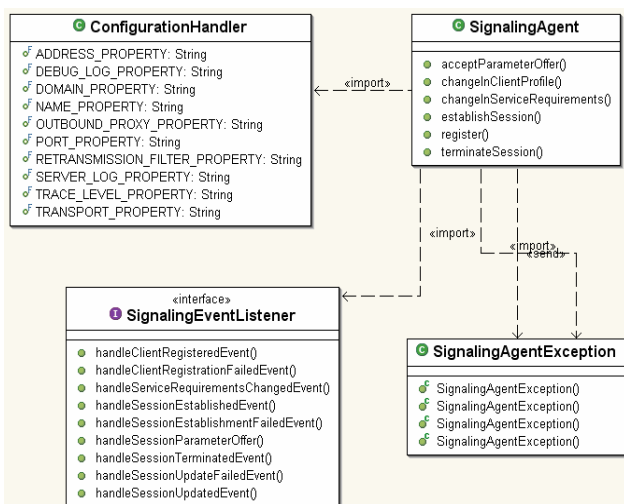
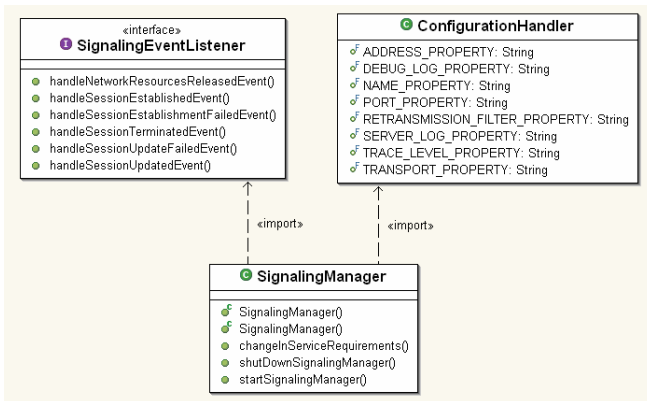


Figure 9. Package *hr.fer.tel.nims.dsa.client**

Figure 10. Package *hr.fer.tel.nims.dsa.server**

delivering new service configuration, while the *handleSessionUpdateFailedEvent()* method delivers a description of a failure cause along. An implementation of the *handleNetworkResourcesReleasedEvent()* method notifies of network resource release for a particular client. Each client must uniquely be identified with its IP address.

The *profilemanipulation* package (Fig. 10) consists of several Java classes that handle the service configuration format (service profile) explained in the first section. The *ProfileInterface* Java interface defines basic format/configuration manipulation methods, while the *ProfileParser* Java class implements methods for parsing it and modifying its values. The *XmlElement* Java class symbolizes a tag in the XML (Extensible Markup Language) structure with accompanying attributes.

IV. CASE STUDY

In order to show the applicability of the DSA API, a prototype Web-based application has been designed and developed. It hosts a 3D virtual world featuring a treasure hunt-like game and was extended with the signaling capability. Example client and service profiles that describe different user preferences, terminal capabilities, access network conditions, and service requirements were specified. Using the prototype application functionality of the API was tested in a laboratory testbed.

A. The prototype application

Our case study application, the *Inheritance Chase*, is a multiplayer game based on the client/server network architecture. The game scenario consists of a real-time adventure similar to a treasure hunt and is taking place in a 3D world developed using the *Virtual Reality Modeling Language* (VRML). Its plot is as follows. Players' rich distant relative has deceased recently and left a vast inheritance. His last will is hidden somewhere in the virtual world and each player has to find it first in order to get the inheritance. To achieve that, they have to follow different audio and/or video clues.

The virtual world (Fig. 11) consists of two scenes: an island with two houses (Fig. 11a), which is a part of the world where

most of the game takes place, and the scene containing a large chessboard (Fig. 11c), associated to one of the clues. After a player enters the game, the main scene is retrieved from the server side. Each player is represented with an avatar (virtual 3D character, Fig. 11b) which is visible to other players. As players explore the world, they come across the clues. Clues that lead players to finding the will were designed in different forms - some of them are streaming audio/video clips, others were implemented using special VRML elements bound to the scenes themselves. There are particular scene objects that are to be selected with the mouse in order to start "streaming" clues playing. All this service content (virtual 3D scenes, avatars, real-time streaming media, texture images) contribute to complexity of the system which, we believe, may serve as an example of an advanced multimedia and multimodal application, and its complex QoS requirements at the transport layer.

Implementation of the application hosting this service is divided into three parts. The first part refers to the SIP signaling functionality in the terms of specifying service requirements using service profiles. This logic was developed in a way to meet dynamic nature of the system and handle exchange of signaling messages as defined by dynamic negotiation and adaptation scenarios (chapter 1.A). The second part is responsible for retrieving the 3D scenes, starting/stopping and displaying audio/video clips, synchronizing virtual world states among different players etc.



a. The island



b. Players' avatars

Figure 11. The Inheritance Chase game



c. The chessboard

Figure 11. The Heritage Chase game

“Real-time media” hints are streamed and displayed using *Java Media Framework (JMF) API* based players. The third one is related to the service content and has been realized using an *Apache Tomcat* Web server. Multiplayer engine of the game is called *DeepMatrix*, and is written using *Java* programming language.

The *Inheritance Chase* game has been developed in three different service versions. Each of them regulates which service components, and in what form, are going to be delivered to a user, depending on the client side capabilities, network conditions, and service requirements. These three support:

- (1) high quality audio and video streaming,
- (2) low quality audio and video streaming, and
- (3) low quality audio streaming only,

each with accompanying set of media codecs offered. Their configurations are stored as service profiles on server that hosts the service and organized according to predefined XML structure. For each service version this implementation provides only static transcoding, which means that service content has to be prepared in advance.

The client side is represented with several different client

profiles based on various user terminal capabilities, access network characteristics, and user preferences. Client profile format is based on the SDPng [5].

B. Laboratory setup and test scenarios

As mentioned before, DSAM model was mapped to the 3GPP IMS architecture. This mapping was used as a reference for DSA API implementation. The API embedded in DSAM prototype implementation entities of a laboratory testbed is shown in Fig. 12. The *Session control element* is responsible for managing the signaling flows. It routes messages from the *Client* to the *End point Application Server* through the *QoS Matching and Optimization Node*. It is also responsible for registration and authentication procedures. The *QoS Matching and Optimization Node* is the central part of DSAM model, responsible for matching process and optimization calculations. The *Policy enforcement* element is used for reservation of the negotiated network resources. It also detects any change in network resource availability. The *Network control element* is used for forwarding signaling messages between the *Client* and the *Session control element*, and for passing negotiated resource reservation parameters to the *Policy enforcement* element. The *Virtual channel* emulates various network conditions.

The prototype application integrated with developed DSA API was tested with the following scenarios [10]:

- Session establishment,
- Change in service requirements,
- Change in client profile, and
- Change in network resource availability.

Session establishment is invoked by an end-user (a player). It includes procedure of registering a user-terminal to the network, negotiating initial service parameters (Fig. 13) in the terms of final service profile, and service retrieval (scene download, Fig. 14) in accordance with negotiated configuration. This scenario also comprises signaling

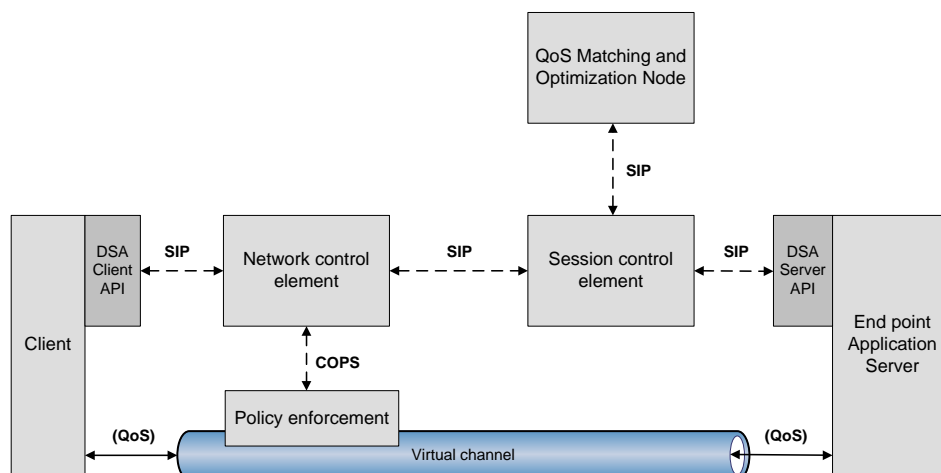


Figure 12. DSA API embedded in laboratory testbed implementation entities

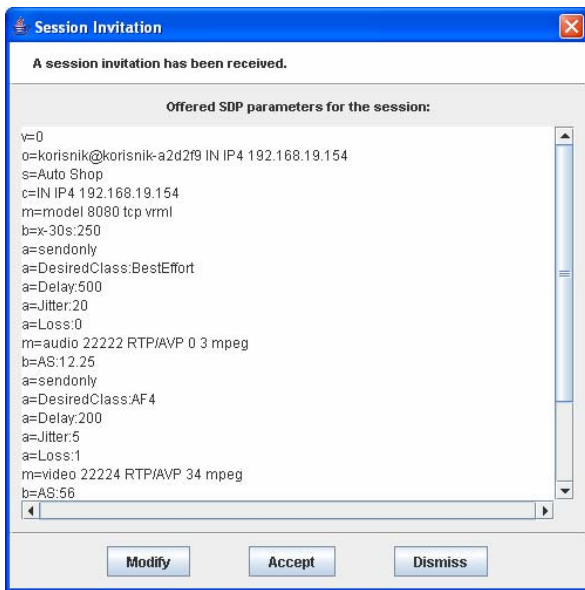


Figure 13. Session parameters offered during initial session establishment

authorization, reservation, and release of network resources used for scene download, as needed.

Change in service requirements is caused by a user initiating an audio and/or video streaming. Signaling new service requirements is invoked by the server side, and new service configuration is negotiated based on information, carried in signaling messages, that are related to streams being requested. The *QoS Matching and Optimization Node* calculates optimal audio and video codec combination based on user preferences, user terminal constraints, network capabilities (bandwidth, delay, loss, etc.), resource cost, and service requirements. Prior to starting media streaming (Fig. 15), reservation of network resources is signaled. Through the *Policy enforcement* entity the *Network control element* reserves the calculated resources at the virtual channel.

The third scenario, change in client profile, is caused by an increase or a decrease in the user's access network bandwidth, which results in new negotiation and optimization process.

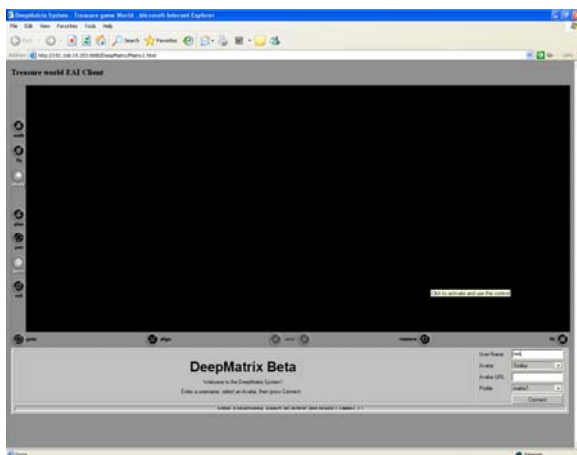


Figure 14. Service retrieval

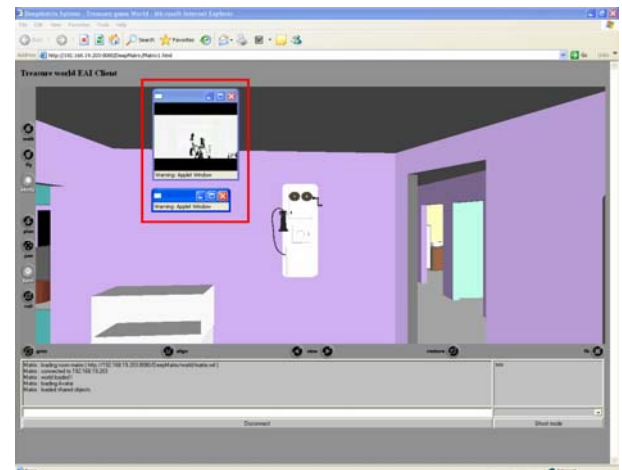


Figure 15. "Streaming" audio and video clues

This change is simulated by sending a new client profile configuration from the client side. If, for instance, media streaming is taking place at that instant, and if a variation of the bandwidth increase is significant, automatic change of the streaming quality/codecs (Fig. 16 and Fig. 17) will occur according to the new service configuration.

Change in network resource availability is detected by the *Network control element* (receives information from the *Virtual channel*). This again invokes negotiation and optimization process, which results in a new service configuration. Automatic changes of service parameters (i.e. audio codec due to a decrease of authorized network resources, Fig. 18 and Fig. 19) at the client and the server side occur in compliance with negotiated service profile.

During service run-time, changes in various parameters/constraints, related to the client side, service requirements, and the network resources can occur, and it was shown that each time an adapted version of all the service components will be delivered to a user.

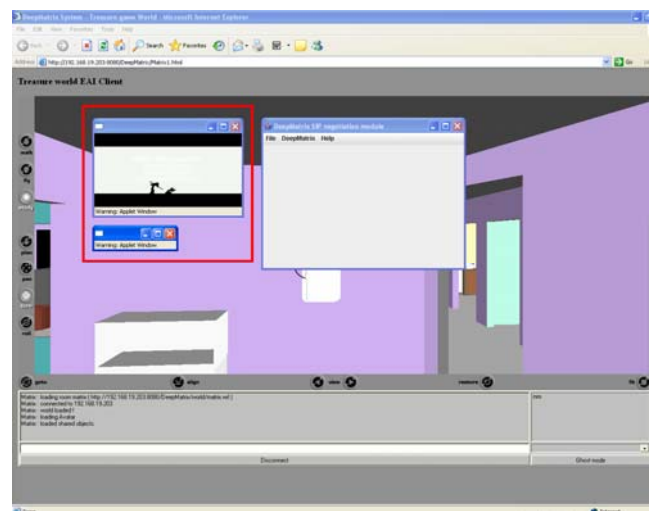


Figure 16. "Streaming" audio and video clues after codec change

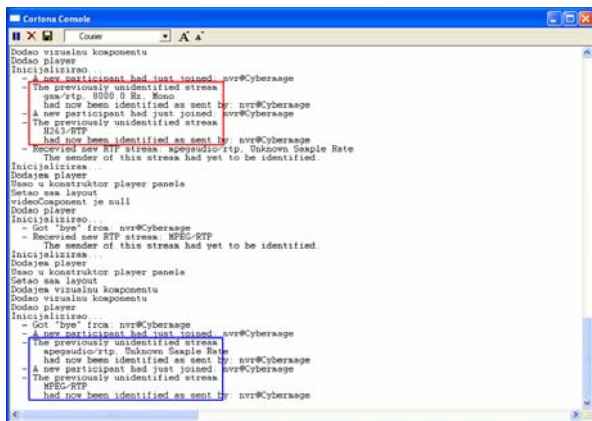


Figure 17. Applied codec change after increase in network bandwidth

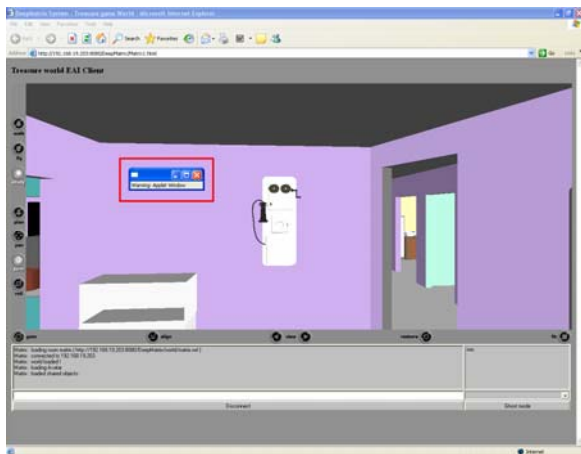


Figure 18. "Streaming" audio clue

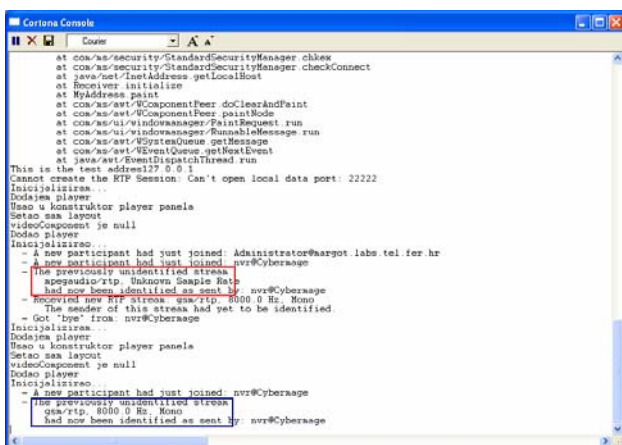


Figure 19. Applied codec change after authorized resources decrease

V. CONCLUSION

The developed API offers various benefits to application developers. It may ease development of advanced multimodal and multimedia applications with network-aware adaptation

and shorten the application development time. The proposed approach differs from current approaches, where applications either (1) do not use signaling at all (e.g. most Internet applications), or, (2) use a standard network and/or service specific signaling protocol (e.g. H.323, SIP) but have the signaling capability built into, and thus inseparable from, the client application or client platform. While the second approach enables the exchange of control information, it is practically impossible to reuse this functionality due to tight coupling with the application. Also, this approach assumes that the application developer knows the signaling protocol specifics very well, and is capable of building the signaling agent into each and every new application from scratch. Finally, once built into the application, signaling support can not be upgraded to, for instance, a more recent release of the signaling protocol without significant effort and rebuilding the whole application. The proposed approach solves these problems.

ACKNOWLEDGMENT

Ivan Piskovic was partially funded by SIMILAR, the European Network of Excellence on Multimodal Interfaces.

This work was supported by research project Networked Virtual Reality in IP Multimedia Subsystem (NIMS), conducted in the cooperation between the Faculty of Electrical Engineering and Computing, University of Zagreb, and the Ericsson Nikola Tesla company from Zagreb.

REFERENCES

- [1] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation*, Addison-Wesley, 1999.
- [2] J. Rosenberg et al., "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [3] NIST-SIP [Online]. Available: <http://snad.ncsl.nist.gov/proj/iptel/>
- [4] 3GPP TS 23.228: "IP Multimedia Subsystem (IMS); Stage 2", Release 7, June 2005.
- [5] Kutscher, Ott, and Bormann, "Session Description and Capability Negotiation", draft-ietf-mmusic-sdpng-08.txt, TZI Universitaet Bremen, February 20, 2005.
- [6] SAX [Online]. Available: <http://www.saxproject.org/sax1-roadmap.html>
- [7] G. Camarillo and G.-M. Miguel-Angel, *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds*, John Wiley & Sons, 2004.
- [8] 3GPP TS 23.218: "IP Multimedia (IM) session handling; IM call model; Stage 2", Release 7, June 2006.
- [9] 3GPP TS 24.228: "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3", Release 7, December 2005.
- [10] L. Skorin-Kapov and M. Matijasevic, "Dynamic QoS Negotiation and Adaptation for Networked Virtual Reality Services" in *Proc. of the Sixth IEEE International Symposium on a World of Wireless and Mobile Multimedia*, Italy, 2005, pp. 344-351.
- [11] L. Skorin-Kapov and M. Matijasevic, "End-to-end QoS Signaling for Future Multimedia Services in the NGN", *Lecture Notes in Computer Science* (0302-9743), St. Petersburg, 2006.
- [12] M. Matijasevic, D. Gracanin, K. P. Valavanis, and I. Lovrek, "A framework for multi-user distributed virtual environments", *IEEE Transactions on Systems, Man and Cybernetics*, Part B: Cybernetics, 32(4):416-429, August 2002.

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. As an undergraduate student with state scholarship he participated in the Summer Camp 2003: "Agent and Visualization Technologies" workshop, jointly organized by Ericsson Nikola Tesla in cooperation with the Faculty of Electrical Engineering and Computing, University of Zagreb. Later, he became one of the project leaders at Summer Camp 2005 "Exploring ICT Frontiers: Agents, IP Multimedia Subsystem, and Distributed Computing". He published several conference papers, some of which were awarded (IEEE 12th MELECON 2004; Best Student Paper Award). His main research interests are in the field of multimedia communications, virtual environments and mobile applications development.

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 in the Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. Her main research interests include networked virtual environments and advanced multimedia for next generation networks. She authored more than 40 publications, and served as a guest editor for several journal special issues. She has been involved in organization of several international conferences. Dr. Matijasevic is a senior member of IEEE, and member of ACM and Upsilon Pi Epsilon Honor Society in the Computing Sciences.

Sasa Desic received his Dipl. Ing., M.Sc. and Ph.D. degrees from the Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, in the years 1997, 1999 and 2004 respectively. From 1997 to 2000 he worked as a research assistant at Faculty of Electrical Engineering and Computing. He has been with Ericsson Nikola Tesla since 2000, currently employed as head of the Research Department. Previously he was the project leader of a joint research project conducted in cooperation with the Faculty of Electrical Engineering and Computing called "Remote operations management" investigating systems for software management on the remote locations. He actively participates in several research projects conducted in cooperation with the academia including project Location Based Services investigating the integration of different sources of location information. Since 2004, he is also an adjunct assistant professor at the University of Zagreb, Faculty of Electrical Engineering and Computing.

Ognjen Dobrijevic received his Dipl. Ing. degree in Electrical Engineering from the University of Zagreb in June 2004. In 2002 and 2003 he participated in the workshop, organized in the collaboration between the Research Department of the Ericsson Nikola Tesla company in Zagreb and the University of Zagreb, Faculty of Electrical Engineering and Computing, where he was engaged in projects Mobility in Advanced Network Architectures and Visualization Technologies. He has been a research assistant at the Faculty of Electrical Engineering and Computing in Zagreb since September 2004, where he is working towards his Ph.D. His main research interests include next generation networks, related QoS signaling issues, and adaptive multimedia applications. He is a member of the IEEE association.

Ivan Piskovic was born 1983 in Zagreb, Croatia. He finished high school in Zagreb in 2001 and is now an undergraduate student of the Faculty of Electrical Engineering and Computing, University of Zagreb. His interests include architecture of next generation networks and related multimedia session-control issues.

Mirko Suznjec was born 1983 in Karlovac, Croatia. He finished high school in Glinina in 2001 and is now an undergraduate student of the Faculty of Electrical Engineering and Computing, University of Zagreb. In 2005 he participated in Ericsson's workshop, Summer Camp 2005 "Exploring ICT Frontiers: Agents, IP Multimedia Subsystem, and Distributed Computing", where he worked on development of a networked virtual environment application with multi-user support. His interests include networked virtual environments and network design.

APPENDIX: SOFTWARE DEMONSTRATION NOTES

In order to run demonstration example, installation and configuration of three software components has to be done. Instructions for running each of the provided components under Windows operating system will be described hereafter.

A. Instructions for installing the Inheritance Chase game and the End point Application Server

First step is to:

- 1) Copy the *matrix_server* folder (further on referred to as '+') to a machine that will run the game server application.

After copying the folder:

- 2) Create *C:\Documents and Settings\User\Desktop* destination folder, if one already does not exist, and copy *sdpcontent.txt* and *sdpParameters.txt* files from the *+sdpmediadescription* folder to the destination folder.
- 3) Edit the *configuration.properties* file, in the *matrix_server* folder, to respond to IP address of the computer and wanted port number.

In order to run the game server, the computer furthermore needs to have the following software installed:

- Apache Tomcat Web server (version 5.5 preferred), and
- Java Media Framework (JMF) API (version 2.1.1e preferred). When the JMF is installed, the *jmf.jar* file from the *<JMF_folder>\lib* folder needs to be copied to the *+lib* folder.

After installation of the Web server:

- 1) Modify the *gallery<1,2,3>.xml* files in the *+serviceprofilerepository\gallery\profiles* folder so that IP address in the *url* parameters responds to IP address of the computer used as the game server.
- 2) Complete folder named *DeepMatrix* needs to be copied into the *<Tomcat_folder>\Apache Software Foundation\Tomcat 5.5\webapps\ROOT* folder (further on marked as '*').
- 3) Class file *EventManager* from the *+bin\nvrcontentserver\vrserviceprocessor* folder needs to be copied into the **WEB-INF\classes\nvrcontentserver\vrserviceprocessor* folder.
- 4) Class file *GalleryServlet* from the *+bin\nvrcontentserver\vrserviceprocessor\...galleryservice* folder needs to be copied into the **WEB-INF\classes\nvrcontentserver\vrserviceprocessor\...galleryservice* folder.
- 5) File *web.xml* in the *<Tomcat_folder>\Apache Software Foundation\Tomcat 5.5\conf* folder needs to be modified so that the comments "around" servlet *org.apache.catalina.servlets.InvokerServlet* and

"around" servlet mapping for the invoker servlet (tag *servlet-name* equals to *invoker*) are deleted.

- 6) Files *Matrix.html* and *Matrix1.html* in the **DeepMatrix* folder have to be modified so the IP addresses correspond to the machine that hosts the game server.
- 7) File *matrix.wrl* in the **DeepMatrix\world* folder has to be modified in scripts *calling* and *calling2* so the IP address/computer name corresponds to the machine that hosts the game server. If an IP address is used, input format must be complied. Furthermore, modify IP address in the first two *url* parameters of the *Anchor* node in the *Object3* Transform node to match the computer being used as the game server.

B. Instructions for installing the QoS Matching and Optimization Node

For the purposes of matching client and service profiles, and optimizing final service configuration that is delivered to a user, the *QoS Matching and Optimization Node (QMON)* needs to be installed. First step is to:

- 1) Copy the *SIP-AS* folder (further on referred to as '#') to a machine that will run the component. It is recommended to use a different machine than one hosting the game server application, but it is not necessary.

After copying the folder:

- 2) Edit the *configuration.properties* file, in the *SIP-AS* folder, so the *javax.sip.IP_ADDRESS* and the *hr.fer.teletk.NETWORK_PRICE_QUOTATION_ADDRESS* parameters respond to IP address of the computer, and *hr.fer.teletk.SIP_STACK_PORT* to a wanted port number. Furthermore, the *javax.sip.OUTBOUND_PROXY* parameter must respond to IP address and port number of the *Server's* configuration.
- 3) Modify all *txt* files in the *#\SimulatedResponses* folder, so that the *ReceivingClientIP* tag responds to IP address of a computer running the client and IP address in the *url* parameter(s) matches IP address of the computer hosting the game server.
- 4) Add the absolute path of the *#lib* folder to the *Path* system variable.

C. Instructions for installing the Client

First step is to:

- 1) Copy the *matrix_client* folder (further on referred to as '\$') to a machine that will run the game client application. It is recommended to use a different machine than one hosting the *QMON*, but it is not necessary.

After copying the folder:

- 2) Edit the *configuration.properties* file, in the *matrix_client* folder, to respond to IP address of the

computer and wanted port number. Furthermore, the *NEXT_HOP* parameter must respond to IP address and the *hr.fer.teletk.SIP_STACK_PORT* port number of the *QMON*.

- 3) Modify the four files, in the *matrix_client* folder, that contain word *gallery* in their names so that IP address in the *Host* parameter corresponds to IP address of the computer hosting the game server.

In order to run the client application, the computer needs to have the following additional software installed:

- Microsoft Java Virtual Machine (MS JVM, version 5.00.3810 preferred),
- a Web browser (Internet Explorer preferred),
- VRML player/viewer for the particular Web browser, i.e. if Internet Explorer is used, Cortona VRML Client is preferred, and
- JMF API (version 2.1.1e preferred). When the JMF is installed, the *jmf.jar* file from the *<JMF_folder>\lib* folder needs to be copied to the *matrix_client\lib* folder.

When installing the JMF, mark all the checkboxes setup offers. In the *Tools->Internet Options...->Advanced* section of the Internet Explorer, usage of the compiler for Microsoft Virtual Machine has to be enabled.

D. Simple demonstration scenario

When software components have been properly configured, each of them can be started using accompanying batch file. Following simple demonstration scenario, a particular functionality of DSAM prototype implementation can be portrayed. Each client must uniquely be identified with its IP address. Before starting demonstration, it should be checked whether the Web server has been started.

Assumed demonstration scenario will reflect functionality of the software components related to first three scenarios of the case study - session establishment, change in service requirements, and change in client profile.

Session establishment is invoked by a user sending a particular client profile - in this case, for instance, the *Matrix audio and video LQ*. This particular client profile depicts conditions in UMTS access network with somewhat lower bandwidth and user's interest in both audio and video component of the service. Before sending the profile, the user has to specify IP address and port number of the game server. Next step in initial session establishment is offering session parameters to the user. For now, it is only allowed to accept offered parameters. After negotiating initial service parameters in the terms of final service profile, session establishment procedure successfully finishes and main 3D scene of the game is retrieved. It is displayed in the window of a Web browser when the user has logged in (use login *mm*).

Now change viewpoints to the scene until one containing a phone box is reached. Selecting it with the mouse starts "streaming" clues playing (adding audio/video streaming to the game refers to change in service requirements). Prior to streaming, signaling these new service requirements is

invoked by the server and new service configuration is negotiated based on the information related to media streams being requested.

While the streaming takes place, the user could, for instance, send the *Matrix audio and video HQ* profile. This client profile depicts the same user's preferences (interest in both audio and video) but somewhat altered conditions in the access network related to a higher bandwidth. Sending new client profile results in new negotiation and optimization process. In this case, when the process finishes the streaming quality improves (audio and video codecs change, according to the new service configuration, which can be seen in the console of the VRML viewer).

Completion of the streaming also results in signaling changed service requirements, but this time associated to detraction of service components.

An instrument of sound and visual creation driven by biological signals

Andrew Brouse, Jean-Julien Filatriau, Kosta Gaitanis, Rémy Lehembre, Benoît Macq, Eduardo Miranda, and Alexandre Zénon

Abstract—Recent advances in new technologies offer a large range of innovative instruments for designing and processing sounds. This paper reports on the results of a project that took place during the eINTERFACE06 summer workshop in Dubrovnik, Croatia. During four weeks, researchers from the fields of brain-computer interfaces and sound synthesis worked together to explore multiple ways of mapping analysed physiological signals to sound and image synthesis parameters in order to build biologically-driven musical instruments. A reusable flexible framework for bio-musical applications has been developed and validated using three experimental prototypes, from whence emerged some worthwhile perspectives on future research.

Index Terms—EEG, EMG, brain-computer interfaces, digital musical instruments, mapping

I. INTRODUCTION

MUSIC and more generally artistic creation has often drawn inspiration from the possibilities offered by technology. For example, the invention of the piano was a key event in the emergence of romantic music. More recently, the electric guitar and synthesizer have allowed elements of Jazz to move towards Pop Music. Digital signal processing and multimedia computers have enabled the creation of an overwhelming gamut of new sounds. More recently, work has begun to discover ways to control these new sounds with the ultimate goal of creating new musical instruments which are playable in real-time.

The present contribution is focused on the development of new musical instruments activated by the electrical signals of the brain (EEG) and of the muscles (EMG). We are exploring features of bio-signals by mapping them to parameters of computer-generated sounds. This work is the continuation of a project initiated last year during the first eINTERFACE workshop in Mons, Belgium [1] [2]. In our previous work we used inverse methods and left/right cortical activity differentiation - as in classical Brain to Computer Interfaces (BCI) [3] - to design the mapping between physiological signals and sound synthesis parameters. We felt, however, that the ‘musification’

This report, as well as the source code for the software developed during the project, is available online from the eINTERFACE’05 web site: www.interface.net.

This research was partly funded by SIMILAR, the European Network of Excellence on Multimodal Interfaces, during the eINTERFACE05 Workshop in Mons, Belgium.

R. Lehembre was supported by a grant from the Belgian NSF(FRIA).

Andrew Brouse and Eduardo Miranda are with the Interdisciplinary Centre for Computer Music Research, University of Plymouth, U.K.

Jean-Julien Filatriau, Rémy Lehembre and Benoît Macq are with the Communications and Remote Sensing Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

Alexandre Zénon is with the Neurophysiology Laboratory, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

of biological signals could benefit by using the richness of the raw brain and muscle signals rather than just relying on the results of analyses. Hence, we took the opportunity of this workshop to explore new fission/fusion strategies by conducting three experiments:

- The first application was the sonification of EEG signals, using a vocal model, which could be either used as a musical instrument or as a diagnostic tool
- The second experiment was more directed to musical applications and interactive performance, and is aimed to generating visual and sonic textures controlled by the results of EEG spectral analysis.
- The last application was a tentative attempt to extend the hyper-instrument paradigm by building a physiologically enhanced didgeridoo that relies on wearable sensor technology [4].

This report is composed of five main sections: a history of music and sonification controlled by biological signals; a theoretical framework which exposit the fission/fusion of biological signals in musical applications; a description of the hardware and software architecture of our platform dedicated to musification of biological signals; a section which details the different EEG analysis methods we have implemented; and finally detailed descriptions of the experiments with some possibilities for improvement for each.

II. HISTORY AND THEORY OF SONIFICATION OF BIOLOGICAL SIGNALS

Whereas the use of biological signals to control music systems has a long and rich history dating back at least 40 years [5], the contemporary notion of sonification of biological data for auditory display is relatively recent, the first articulated writings beginning to appear around 1994 [6]. Sonifications as evidence or as objects of scientific knowledge also present fascinating opportunities to interrogate notions of scientific truth and ontology. In fact, the practice of using sound as a tool for medical diagnosis for example, dates back more than 150 years to the development of the stethoscope by René Laennec [7] and the attendant practice of mediate auscultation. As listening to the body is one of the most basic skills in a standard medical education, trained doctors are thus highly sensitive to sound and its implications for diagnosis. Simultaneously, over the past 150 years or so, scientific measurement equipment has become increasingly sophisticated and precise. The possibility of making highly precise measurements of phenomena has - until recently however - been almost exclusively destined for visual display. That is, the results of

these sophisticated measurements has been, to a very large extent, primarily expressed in visual terms: as graphs, line traces, charts, histograms, waterfall charts etc., either on paper or some similar support, or on a luminous display such as a CRT or TFT computer screen. Recently, the notion of auditory display of scientific or other information has become current. Auditory display has several advantages over visual display especially for critical applications largely due to the ways in which our auditory perceptual apparatus passes information to the brain. By using salient characteristics of sound, such as rhythm, duration, pitch, timbre and harmonic/enharmonic content, it is possible to rapidly and accurately express complex, multimodal information in a manner which can be quickly and accurately grasped by a trained listener. Our auditory apparatus is capable of distinguishing very subtle differences in simultaneous, complex auditory streams and it can do this very quickly and accurately [8]. Work has already been done in the sonification of biological signals such as EEG - notably by Gottfried Meyer-Kress and his early work in EEG sonification - which has been furthered by a workshop at ICAD2004 entitled "Listening to the Mind Listening" and even more recently by a workshop and paper given at ICAD2006. In a related field, Mark Ballora did pioneering work in the sonification of the cardiac rhythms related to the diagnosis of conditions such as sleep apnea [9]. In most of the preceding cases, however, the sonifications were performed "offline", that is, not in real-time. The goal of this part of the project is to develop a real-time system for sonification of biological data. Previous efforts along these lines have led to very specific solutions with particular hardware and software components which have proved hard to re-use and not sufficiently flexible for diverse applications. Our goal, thus, is to begin work upon a flexible, re-usable, open-source framework for the generalized sonification of biological signals. This platform would provide a stable, re-usable, flexible and comprehensive environment for the sonification of human biological data for auditory display. This display would be useful for doctors, scientists, researchers and clinicians in the study and diagnosis of normal and abnormal indicators. Much as this is primarily a tool for scientific research, it is also envisioned as a useful tool for music technologists, composers and performers in the realisation of musical forms which are driven by measured biological phenomena. It is felt that a stable platform for such musical research is as useful in the musical sphere as it is in the scientific one. In fact, an historical survey of biologically driven music, such as brainwave music, shows periods of intense, productive activity followed by quiescent lulls where very little happens. It is felt that these lulls are due in part to a lack of appropriate tools and techniques for consistent and repeatable musical realisation and thus, little opportunity for practices and mastery of bio-instruments such as brainwave music.

III. FISSION AND FUSION OF BIO-SIGNALS

A. Our proposed framework

We proposed to model the design of musical instruments or sonifications as a fission-fusion process. Our theoretical

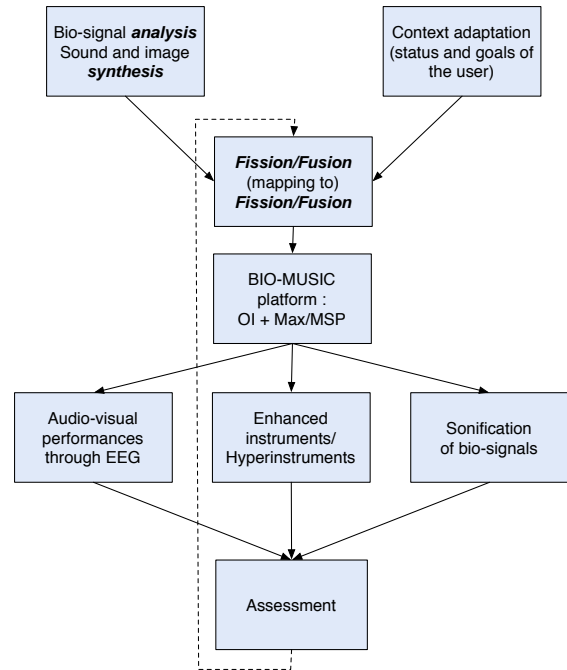


Fig. 1. Framework for the design of biologically-driven musical instruments

framework is shown in Fig. 1. The central issue is the fission from each of the given input modalities (EEG and/or EMG in this case) into salient features channels. These features channels are then fused into commands which activate different aspects of the related sound and image synthesis processes. The process of fission of commands into the output feature channels which are then fused back into the global output signal is also seen as part of the fission-fusion process. This process can be likened to the attendant processes of analysis and resynthesis which are so central to digital signal processing.

B. Mapping

In the literature on digital musical instruments [10], the term mapping refers to the transformations performed upon real-time data received from controllers and sensors into control parameters that drive sound synthesis processes. One of our objectives during this workshop was to design consistent mappings between biological signal features and sound synthesis parameters in order to create biologically-driven musical instruments and sonifications.

C. Usability measurements

The three systems will be improved based upon: assessments of usability and aesthetics by musicians, aesthetic judgments by audiences, and quality of discrimination between relevant EEG patterns in the case of sonification for diagnostic purposes.

IV. THE PLATFORM ARCHITECTURE

A. Towards an open source system

Our aim in the long term is to produce an entirely open source platform dedicated to the real-time analysis of EEG

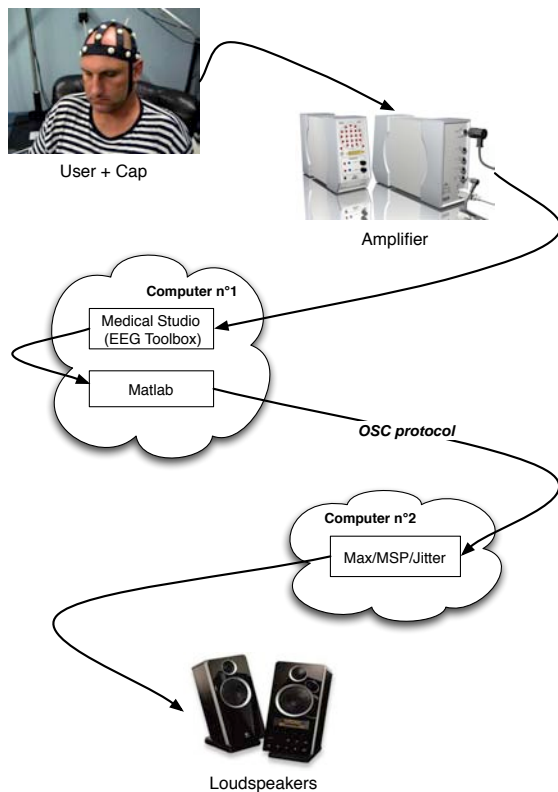


Fig. 2. Architecture of our bio-music platform

signals and other biosignals for musical applications. Since our work is multidisciplinary it involves using resources from different fields of study and thus different software packages are needed. During this workshop we used Matlab for the analysis of EEG signals and Max/Msp/Jitter for the sound and image synthesis. We plan to shift our development toward open source software like Octave [11], Python [12] and PureData [13] in the future. In the following, we describe the architecture of our system (Fig. 2) in a bottom-up way, from hardware data acquisition to software implementation.

B. Hardware

1) *EEG equipment*: EEG signals are recorded with a *dti* [14] cap containing 18 electrodes located according to the 10/20 international positioning system. The signals are amplified with an biosignal amplifier provided by *dti* with a gain of 10^6 and a default sampling rate of 128Hz. Due to limitations in real-time signal processing, we sampled at 64Hz. Once captured, the data is then bandpass filtered between 0.5 and 30 Hz to remove extraneous signals. C_z was used as a reference electrode while P_z was taken for the ground.

2) *EMG equipment*: For EMG signals, we worked with the same equipment but changed the gain to 1000 since EMG signals have much larger amplitudes than EEGs. Disposable electrodes were used, 3 per muscle, with one as a reference and placed near a bone (i.e. elbow or knee), a second was posed along the muscle (belly-bone junction), the third, taken as ground, was via a conductive bracelet worn by the user.

C. Software

Our platform is currently implemented via four software packages running on two computers which manage the specific tasks required by the global application :

1) *MedicalStudio-EEGToolbox*: Acquisition and visualisation is done using EEGToolbox, a plugin written in C++ for MedicalStudio [15], an open source software platform for medical data analysis and display which runs under Linux. This toolbox saves the data and can also send it using UDP to another computer running Simulink under Windows. The connection between the biosignal amplifier and the computer running Linux is made with a usb cable.

2) *Matlab-Simulink*: Matlab was chosen for easy code generation. Although we had developed the previous year a simulink program, we switched to Matlab in order to spare a computer. This way, the acquisition and signal analysis is made on the same Linux running computer. Simulink could not be used because it suffers from different bugs under Linux that makes it hard to use.

3) *Max/Msp/Jitter*[16]: : Max/MSP is a graphical development environment dedicated to real-time interactive applications. In use worldwide for over fifteen years by performers, composers or artists, Max/MSP is a combinaison of Max software for the control of musical applications through MIDI protocol, and MSP, an add-on package for Max enabling the manipulation of digital audio signals in real-time. Jitter is an other additionnal library for Max environment, offering a large range of real-time image and video processing tools.

4) *OpenSoundControl (OSC)* : A link between Matlab and MaxMsp: In order to transfer data between softwares, we used the OSC protocol [17] which sits on top of the User Datagram Protocol (UDP). It allows a fast and reliable data exchange since we work in a local area network. Packets are sent with a header containing the name of the corresponding data as well as the size of the packet. This makes it very useful since the receiving program can easily manage the arriving packets. The maximum size for the packets is 65536 bytes long. We were thus able to send the raw EEG signal and many features computed with matlab to Max/Msp allowing a maximum flexibility (An excerpt from the code is detailed in App. I).

V. FISSION OF BIO-SIGNALS

A. Introduction

We worked with two different bio-signals, EEG and EMG. We describe in this section how to operate a fission of these signals in order to extract relevant features. Let us present briefly these two kind of signals:

- The EEG signal is a rich and complex reflection of neuronal electric activity that takes place in the brain. Since the first electroencephalogram recording made by Berger in 1929, different waves have been described corresponding to several frequency bands. Although these waves are well known, their frequencies and amplitudes are not directly under subject's control, but only reflects very general states of the brain. Therefore, using a simple frequency analysis as input to the sound synthesizer will

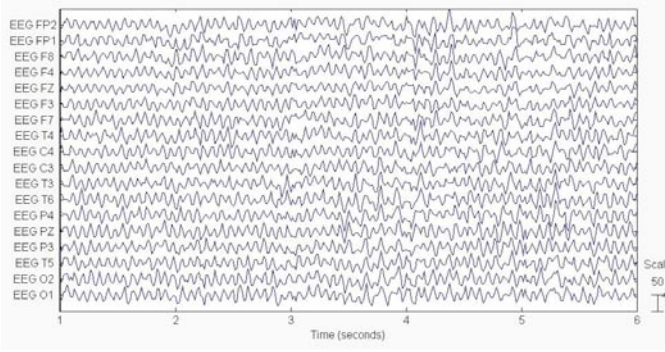


Fig. 3. Recorded EEG, a 13Hz alpha rhythm can be observed. The user was in a drowsy state after a heavy lunch and had taken an espresso

not allow enough controllability. Other, more complex signal properties can reveal more useful. On an other hand, EEGs have a very good time resolution of $\simeq 1ms$ unbeaten by recent methods (fmri..). This property is very valuable for the purpose of musical instrument control and should be taken care of. Finally, since electrodes are placed at different locations, it is important to take into account the spatial information.

- The EMG signal is produced by the electrical potential generated by muscle cells. The increase in contraction strength of the muscle is associated with an increase in the number of cells that produce electrical potentials (depolarisation), and hence an increase in signal amplitude. This signal contains two main waves, a low frequency wave that describes the movement, and a higher frequency wave that includes more precise information on the electrical activity of the muscles. Due to hardware limitations, we focused on the the low frequency band (i.e. the envelope of the signal). The higher frequencies could be used in a further version of the project to take advantage of their higher temporal resolution.

B. EEG fission according to frequency bands

We describe here the partition of the EEG into frequency bands

- Delta (0.5-4 Hz): This wave has first been discovered by W. Gray Walter in 1936 with a patient that had a tumor. Thus in the awake, it is quite alarming to present the slow characteristic waveform of the delta rhythm. However, for a sleeping person, high amplitude delta waves are normally present in the EEG. For our application it appears evident that this rythm will not be of great use unless we create a composition for sleeping performers!
- Theta (4-8 Hz): Scientists still debate whereas theta activity is relevant to an early drowsiness state or if it reflects some kinds of mental activity. Nonetheless it is a faster rythm than delta and could be linked to brain activities such as memory [18], or can be modulated by visual stimulation (ref).
- Alpha (8-12 Hz): Alpha rythm is a leading indicator of subject's relaxation. Alpha synchronization (leading to amplitude increase) occurs in the absence of any visual

stimulation, as for example, when the user closes his eyes. In contrast, any visual stimulation lead to posterior alpha desynchronization. Therefore it is a good tool for our application since it can be used as a switch. Alpha waves could also be associated to conscious visual perception [19].

- Beta (12-24 Hz): Extending over a large bandwidth, the beta activity reflects intense activity such as listening, taking decisions, or more generally, arousal. It is a dominant rythm in the normal adult awake EEG.
- Mu rhythm: This rythm, as the alpha rythm is between 8 and 12 Hz but is specific to imaginary or real movements [20]. It is located in the motor cortex and is contralateral to the movement i.e. for a left hand movement, a desynchronization will appear in the right hemisphere.

Let us recall that the frequencies given above are not strict but subject dependent. The control of these waves by the subject can only been achieved following extensive training. As a consequence, it is difficult to produce a controllable EEG driven musical instrument on the basis of the amplitudes of these signals alone. However we can derive a few indicators from a spectral analysis:

1) *Frequency Values*: A Fast Fourier Transform was used to compute the frequency. We used a 1 sec window to compute a 32 points transform

2) *Spectral Entropy*: The spectral entropy, a measure widely used showing the complexity of a signal, is computed in order to detect salient rhythms. It is given by:

$$H_{sp} = - \sum_f p_f \ln(p_f) \quad (1)$$

where p_f is the probability density function (PDF) that represents the normalization of the power given at frequency f regarding the total power spectrum:

$$p_f = \frac{s_f}{\sum_f s_f} \text{ with } f \in \mathbb{N}^+ \text{ and } f \leq 32 \quad (2)$$

3) *Spectral Edge*: The spectral edge is the frequency under which 95% of the spectral energy can be found This value gives an indication of where the signal is concentrated.

4) *Asymmetry ratio*: In order to detect when the user makes left or right side movement, we use a very simple tool that computes the normalized difference between the power contained in the mu rythm of two electrodes located in the left and right motor cortex (i.e. C_3 and C_4):

$$\Gamma_{[8-12Hz]} = \frac{C_{3,[8-12Hz]} - C_{4,[8-12Hz]}}{C_{3,[8-12Hz]} + C_{4,[8-12Hz]}} \quad (3)$$

This ratio has values between -1 and 1, the sign indicating the side of the body that was moved

C. EEG fission according to signal spatialization

As mentionned above, taking into account the position of the electrodes is extremely important in EEG analysis. Two similar methods, the Common Spatial Subspace Decomposition (CSSD) [21] and the Common Spatial Patterns (CSP) [22], extract information from the most relevant electrodes.

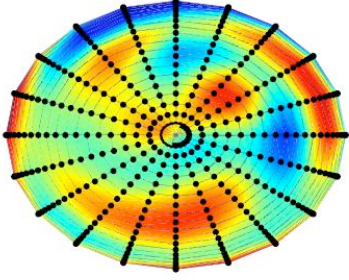


Fig. 4. Inverse Problem visualisation : The black dots are the location of the sources

These methods are known to be the most accurate in the BCI community. However they imply offline preprocessing and low variability between sessions which in our case is seen as a limitation. Indeed, our aim is to produce live music in different environments thus rendering a training session obsolete. An other method that is starting to gain success is based on the principle that the EEG signals are generated by sources (i.e. assemblies of neuronal cells that when combined produce a sufficiently strong current that can be measured at the surface of the scalp) and that the propagation of electrical currents through brain tissues can be modeled with Maxwell's equations. Therefore using a model of the brain it is possible to reconstruct the activity of sources and gain access to the spatial location of brain processes. Besides, this method offers a visualisation of the activity. Having described this method in [1], we will briefly resume the main steps of this method:

1) *Head Model*: We used a four spheres head approximation based on [23], [24] and [25]. Each layer represent, the brain itself, the cephalo-rachidian liquid, the cranial box and the scalp. There are 400 dipoles (Fig. 4) distributed over the cortex (the surface of the first sphere). As an approximation, deep sources are not taken into account. The potential measured on the n electrodes, ϕ , is linked to the value of the m sources, φ , according to the lead field matrix G and additionnal noise η :

$$\phi = G\varphi + \eta \quad (4)$$

The lead field matrix is computed once for a given head model and remains constant further on. Knowing ϕ from the recording, we wish to find φ . Unfortunately this so-called inverse problem is an ill-posed problem since the number of unknowns is much greater than the data at hand. Following is a short description of the inverse problem

2) *Inverse Problem*: Solving Eq. 4 can be done using a bayesian formalism :

$$P(\varphi|\phi) = \frac{P(\phi|\varphi)P(\varphi)}{P(\phi)} \quad (5)$$

where:

- $P(\varphi|\phi)$ stands for the *a posteriori* probability to have the source distribution φ matching ϕ
- $P(\phi|\varphi)$ is the *likelihood* i.e the probability to have the given data according to the sources. It depends on the quality of the recording and on the head model

- $P(\varphi)$ is the *a priori* knowledge we have about the sources.
- $P(\phi)$ is a normalizing probability that can be neglected

Finding the best solution to the inverse problem comes down to maximizing the *a posteriori* probability. This can be achieved in various ways as different methods have been proposed during the past 15 years [26] [27]. We implemented the LORETA algorithm because it gives a maximally smooth solution.

3) *Features*: Four features are derived from the solution of the inverse problem and are sent to the sound processing unit. To compute these features, we divide the source space in four subspace representing the frontal, occipital, left and right sensori-motor parts of the brain. This decomposition is based on the fact that the frontal zone is associated with memory and cognitive processes while the occipital region is linked with visualization. Left and right motor-cortex side are associated with left and right limbs movement. This is a very simplistic view of the brain but is adopted as a first approximation.

D. Further work : EEG fission in 3D

We discussed earlier the importance of taking into account the spectral, spatial and temporal information of the EEG. We studied some techniques of spectral information retrieval and a technique to improve the spatial resolution. We could in a future approach combine the inverse problem and spectral methods. Another approach would be to work with spherical harmonics using an interpolation of the electrodes on a half-sphere. Finally, including temporal constraints in the IP could improve the obtained solutions.

VI. EXPERIMENTS

A. Sonification (Vocalisation) of EEG

The current implementation of sonification uses a source-filter voice synthesis model developed by Nicolas D'Alessandro and others [28] which in this case has been tuned to emulate the multi-phonetic singing chants typically produced by Tibetan Gyuto monks or by Tuvan traditional folk singers. The voice synthesis model as delivered, exposes a limited set of functionalities with given ranges. In the interests of proper encapsulation and OO design, we respect these givens and will work with them. In this case the controller mappings used the F1-4 formant frequencies whilst the F0 was not directly controlled. Additionally, parameters representing "tension", "hoarseness", "chest/head balance" and "fry" were also controllable. Any available mapped data source (alpha, beta, theta, mu etc.) can be used as a controller for any of the synthesis parameter. It was found that the formant frequencies were best controlled by signals which do not change too quickly or vary too greatly. A facility is available to control the positioning of any generated sound source with respect either to a stereo sound field or to a 5.1 quasi-surround sound field.

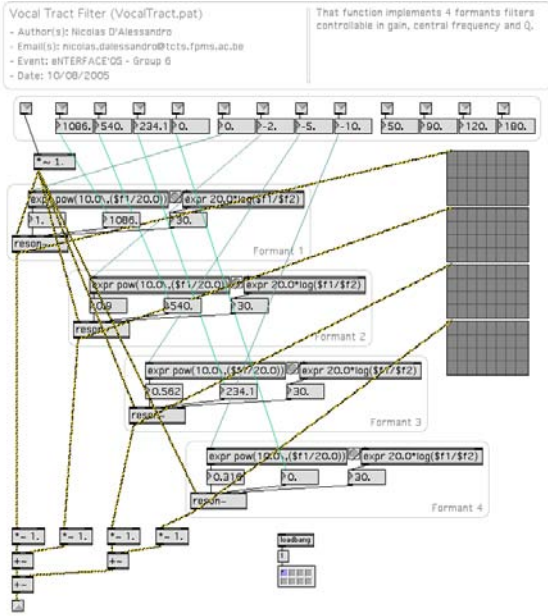


Fig. 5. Vocal Tract Filter realised by Nicolas d'Alessandro et al. It implements four formants controllable in gain, central frequency and Q.

Results: Due to the highly prototypical nature of this platform, no extensive testing was done and it is thus not possible to provide comprehensive analyses of the relative success or failure or suitability of this platform for any current intended usage. Test that were made did indicate that basic functionality of modules and the software as a whole is intact and operational yet many improvements in precision, usability and flexibility are still lacking. Going forward it is envisioned that these characteristics will be ameliorated so that the platform will become a flexible, stable, consistent and useful tool for scientists, medical professionals and musicians in the future.

B. EEG driven audio-visual texture synthesizer

In this instrument we tried to link three modalities by exploiting results of EEG frequency analysis to control both visual and sonic textures synthesis modules (Fig. 6). This approach aimed to provide a visual feedback to the performer/audience enabling a better understanding of the fission/fusion process. Practically, the image synthesis module takes as input parameters data received from EEG analysis module, whereas sound synthesis parameters are extracted from both the output image and the results of EEG analysis. This strategy of linking synthesis processes should enable a strong correlation between resulting image and sound. Both synthesis modules have been implemented in Max/MSP environment, the image processing tasks relying on the specialized additional library Jitter. Following sections give more details on both image and sound synthesis modules.

1) *Creation of the visual texture:* The starting point of the creation of the visual texture is a space/frequency representation of cerebral activity: each second the EEG analysis module transmits to the visualization module a matrix containing

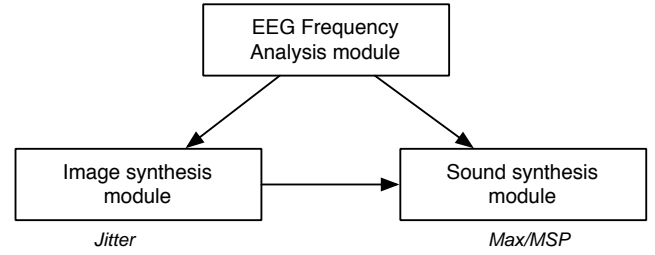


Fig. 6. General scheme of the instrument

the energy in the 32 bands of the spectrum of the signals measured by each of the 18 electrodes. A crossfading effect between consecutive matrixes is then achieved allowing to obtain a continuously and smoothly changing image. This moving image is then distorted: firstly a linear interpolation is done in order to blur the image. At this step of the process, the resulting image is a grayscale texture derived from the space/frequency representation of the EEG analysis (Fig. 7).

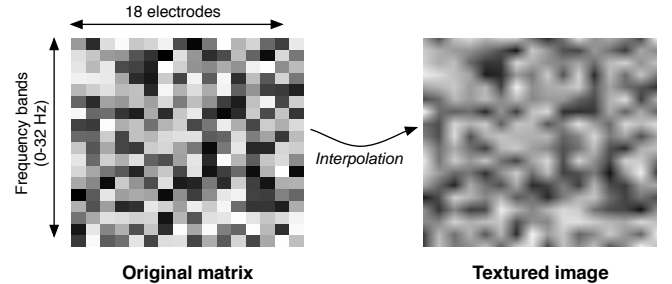


Fig. 7. Creation of a grayscale textured image from the space/frequency representation of brain activity

Then we apply a colorization process, based on color lookup tables, to remap grayscale into colored image. Lookup tables, also called transfer functions, are arrays of numbers where an input number is 'looked up' as an index in the table. The number stored at that index is then retrieved to replace the original number. In our case, we use lookup table to convert a monochrome into RGB value. In grayscale image, low-energy areas are represented in black and gradually whiten when energy increases. Our colorization process modifies the color associated to maximal energy, by defining a new color scale that will map in the resulting image high values, originally represented in white, to a new color defined by the result of EEG analysis image. The choice of the color, called C , associated to the maximum of energy, is driven by the distribution of energy between the alpha, beta and theta bands of the EEG signals. The three RGB components of this color, C_{Red} , C_{Green} and C_{Blue} , are thus weighted by the level of energy L_α , L_β , L_θ in the three frequency bands alpha, beta and theta respectively (Fig. 8). We obtain by this way a direct link between the color of the resulting image and the maximal energy frequency band of the EEG analysis.

The color lookup table is refreshed as soon as new values for alpha, beta, theta bands are received from EEG analysis module, i.e. one time per second. The transfer function used

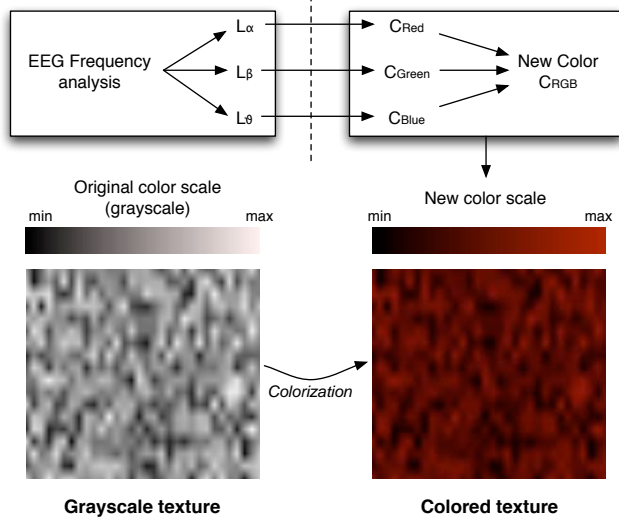


Fig. 8. Colorization of the texture following the distribution of energy in alpha, beta and theta bands

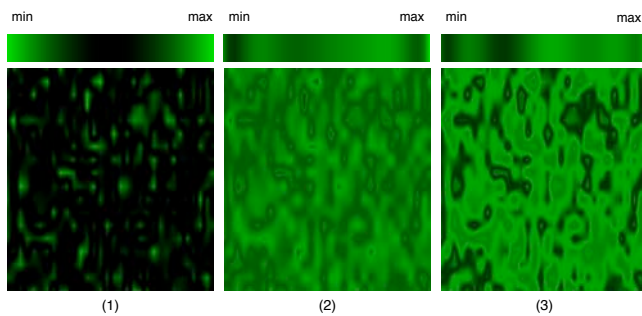


Fig. 9. Textures obtained from the same grayscale texture using different color transfer functions. Rightmost and leftmost images correspond to low and high level of entropy of the signal respectively.

for image in Fig. 8 is linear, but it is also possible to use non linear lookup tables, that give interesting effects on the resulting image and allow to obtain quite different types of visual textures, as shown in Fig. 9. In our instrument six predefined color transfer functions were available, and the choice among them was driven by entropy of the EEG signals, which is an indicator of state of relaxation of the subject, Mapping was done such a way that a dropping of the entropy results a more contrasted image.

2) *Translation in sonic texture*: The translation of the visual texture created from EEG analysis into sound is based on one of the most popular technique of sound synthesis, the subtractive synthesis, widely used in musical applications such as analog synthesizers. The basic principle of subtractive synthesis is the use of complex waveforms, rich in harmonic or inharmonic information, which are then spectrally shaped by filters bank. In subtractive synthesis, the spectral envelope of the resulting sound is the product of the spectral envelope of the source with the frequency response of the filters bank (Fig. 10).

Here we used as audio source a pink noise, whose energy

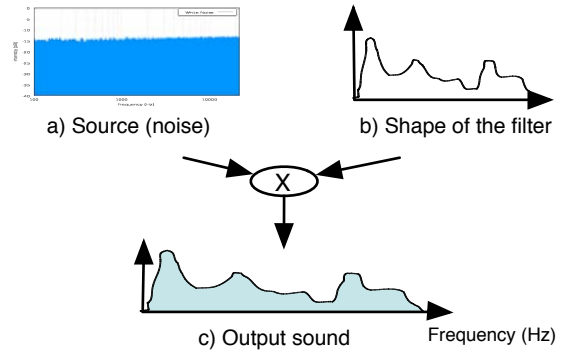


Fig. 10. Principle of subtractive synthesis

is geometrically distributed in the spectrum (constant energy per octave). The implementation of subtractive synthesis in the Max-MSP environment is based on the `fffb` object (fast fixed filter bank), that models a bank of 32 bandpass filters. This object takes as input a list of 32 values controlling the gain of each filter. In our instrument, this list is obtained from the visual texture created from EEG analysis by the following process (Fig. 11) : a sliding window extracts a sharp vertical band of the image (step 1), whose values are stored in a 1-D vector (step 2). This vector is then downsampled to obtain a list of 32 values (step 3) that will be used to drive gains of three filter banks (step 4). In order to musically enrich the resulting sound, we placed three filters bank in parallel, that resonances are differently distributed in the spectrum, implying each of the filters bank to produce its proper and discriminable timbre. Final synthesized sound is a mix of these three sounds whose loudness are respectively controlled by the level of energy in the alpha, beta and theta frequency bands extracted from EEG analysis (step 5), in a similar way of the weighting of RGB components of the final color in the colorization process of the visual texture. This enables a strong correlation between synthesized image and sound, both driven by the results of EEG frequency analysis. Videos demonstrating this instrument are available online [29].

3) *Results and future works*: One aim of this work was to build a brain-computer interface linking image and sound synthesis processes to EEG analysis. We reached this objective by designing a subtractive synthesis instrument that spectral envelop is extracted from a visual texture resulting of EEG analysis. This approach enabled to establish a clear relation between output image and sound. In the future some main tracks of improvement should be investigated. Firstly it would be interesting to modify the space/frequency representation of brain activity that is the basis the creation of the visual texture. Indeed, a spherical representation relying on the localization of the electrodes on the scalp would be closer to the actual spatial brain activity. Concerning the image-to-sound translation, other sound synthesis techniques should be tested, such as additive or granular synthesis, in order to enhance the correlation between the synthesized visual and sonic textures. For this it would be interesting to exploit existing works in the fields of image sonification and auditory display [30]. Finally,

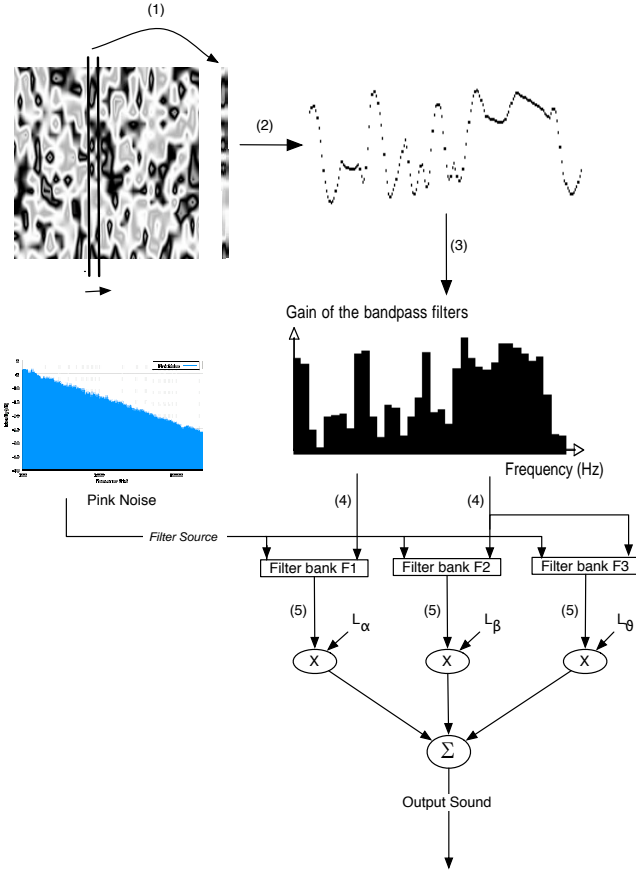


Fig. 11. Image-to-sound texture translation driven by EEG frequency analysis

we should keep working on the improvement of mapping between EEG analysis features and synthesis parameters. In this instrument, the user was actually not able to control the resulting image and sound, mainly because data we interpret as input parameters in the synthesis modules (spectral content of EEG signals) are hardly controllable by the human. In order to increase the playability of the instrument, it could be worth to add in the mapping easily controllable parameters such as EEG features linked to eye blinking. More generally the design of a mapping between EEG analysis results and synthesis parameters in such a brain-computer interface requires an explorative and inventive approach that could only be reached by intensive experimental sessions.

C. EMG enhanced didgeridoo

The third experiment we led during this workshop aimed to design an EMG-enhanced didgeridoo. The didgeridoo is an Australian traditional wind instrument, sometimes described as a wooden trumpet or a drone pipe. Because it is made up without keys, pitch produced by a didgeridoo is limited in a quite sharp range of frequencies, directly related to the dimensions of the instrument. In this experiment we tried to exploit EMG captors measuring contraction of muscles on one leg to enlarge the possibilities of the musician, especially in extending the range of pitch produced by the didgeridoo.

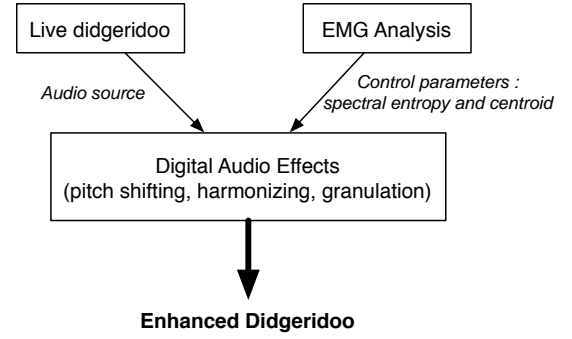


Fig. 12. General scheme of the enhanced didgeridoo

This instrument was running on two computers, one managing Medical Studio for the capture of EMG signal and the other one running Max-MSP for the implementation of digital audio effects. EMG signal, captured with Medical studio, was transferred to Max/MSP, where spectrum centroid, entropy and signal power around a frequency band of 8 Hz were computed. These resulting signals were differentially modulated by leg movements in such a way that the subject was able to control each of them, more or less independently. Two digital audio effects modules were thus designed: in the first one, entropy of the EMG signal, which was the most easily controllable parameter, was used to modify the cutoff frequency of a bandpass filter applied on the didgeridoo's sound. Spectrum centroid controlled a very slight pitch shifting (with a maximum ratio of 1.05) and power in the 8 Hz band controlled the cutoff frequency of a bandpass filter which was used in a feedback loop inside a granular synthesis process. In the second audio effect module, we used entropy of EMG signal to drive two simultaneous pitch shifting processes, one moving downward and another one moving upward. Videos demonstrating these experiments are available online [31]. These quite simple experiments demonstrated the musical potential of EMG-enhanced musical instruments: indeed mapping audio effects parameters with muscles contraction seems to get their control very intuitive and expressive. In the future we will pursue to investigate this field by testing more complex configuration of EMG-enhanced instrument, with multiple captors on several areas of human corpus (arms, neck), providing an actual measure of the physical activity of the musician. Similar experiments will be also carried out with other musical instruments (clarinet, accordion), taking into account the specificity of musical gestures associated to each instrument for the design of captors configuration and mapping strategy.

VII. CONCLUSION AND FURTHER WORKS

Building on the experience gained during the eNTERFACE'05 workshop, we have explored new horizons in bio-music. Last year we focused mainly on left and right hand movements thus working with limited inputs to the sound synthesis algorithms. Our current approach is to take maximum benefit of the richness of the EEG by extracting as many independant features as possible. We have adapted our

modules is to standardise and simplify interfaces with data sources and sinks. To aid in the flexible and rapid ability of users to make fine-grained adjustments in this module, a facility is provided for MIDI continuous controller parameters from any kind of MIDI control surface to be used to adjust the mapping parameters of this module. In this case, the software was prototyped using the Behringer BCR2000, a sophisticated yet easy to use and relatively inexpensive control surface using rotary knobs to adjust continuous controllers. It should be noted that there are similar but independent modules for the mapping of data to visualisation and sonification processes respectively.

ACKNOWLEDGMENT

The authors would like to thank the eNTERFACE'06 organizing team from University of Zagreb, Faculty of Electrical Engineering and Computing, for hosting with great success this second edition of the Similar Summer Workshop.

REFERENCES

- [1] B. Arslan, A. Brouse, C. Simon, R. Lehenbre, J. Castet, J.J. Filatriau, Q. Noirhomme, "A real time music synthesis environment driven with biological signals", ICASSP 2006.
- [2] <http://www.enterface.net/enterface05>
- [3] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller and T. M. Vaughan, "Brain-computer interfaces for communication and control", *Clinical Neurophysiology*, vol.113, 2002, p.767-791.
- [4] A. Kapur, E. Yang, A. Tindale and P. Driessen, "Wearable sensors for real-time musical signal processing", *Proceedings of the Conference on New Interfaces for Musical Expression (NIME05)*, Vancouver, Canada, 2005.
- [5] A. Brouse, *Petit guide de la musique des ondes cérébrales*, Horizon0, vol. 15, 2005.
- [6] G. Kramer (ed.), *Auditory Display : Sonification, Audification and Auditory Interfaces*, Santa Fe Institute, 1994.
- [7] J. Sterne, *The audible past : cultural origins of sound reproduction*, Duke University Press, 2003, pp. 99-136.
- [8] A. Bregman, *Auditory Scene Analysis : The Perceptual Organisation of Sound*, MIT Press, 1990.
- [9] M. Ballora, *Data analysis through auditory display : applications in heart rate variability*, Ph.D. Thesis, McGill University, 2000.
- [10] D. Arfib, J.M. Couturier, L. Kessous and V. Verfaillie, *Mapping strategies between gesture control parameters and synthesis models parameters using perceptual spaces*, *Organised Sound* 7(2), Cambridge University Press, pp. 135-152.
- [11] <http://www.gnu.org/software/octave/>
- [12] <http://www.python.org/>
- [13] <http://pure-data.sourceforge.net/>
- [14] <http://www.dti-be.com/>
- [15] MedicalStudio [Online]. Available: <http://www.medicalstudio.org>
- [16] Max/MSP. [Online]. Available: <http://www.cycling74.com/products/maxmsp.html>
- [17] OpenSoundControl.[Online]. Available: <http://www.cnmat.berkeley.edu/OpenSoundControl/>
- [18] D. Osipova, A. Takashima, R. Oostenveld, G. Fernandez, E. Maris and E. Jensen, "Theta and gamma oscillations predict encoding and retrieval of declarative memory.", *J Neurosci*, 2006, 26, 7523-7531
- [19] C. Babiloni, F. Vecchio, A. Bultrini, G.L. Romani and P.M. Rossini, "Pre- and Poststimulus Alpha Rhythms Are Related to Conscious Visual Perception: a High-Resolution EEG Study.", *Cereb Cortex*, 2005
- [20] G. Pfurtscheller, C. Brunner, A. Schlgl, and F.H.L. da Silva, "Mu rhythm (de)synchronization and EEG single-trial classification of different motor imagery tasks.", *Neuroimage*, 2006, 31, 153-159
- [21] Y. Wang, P. Berg and M. Scherg, "Common spatial subspace decomposition applied to analysis of brain responses under multiple task conditions: a simulation study.", *Clinical Neurophysiology*, vol. 110, pp. 604614, 1999.
- [22] Z. J. Koles and A. C. K. Soong, "EEG source localization: implementing the spatio-temporal decomposition approach.", *Electroencephalogr. Clin. Neurophysiol.*, vol. 107, pp. 343-352, 1998.
- [23] P. Berg and M. Scherg, "A fast method for forward computation of multiple-shell spherical head models.", *Electroencephalography and clinical Neurophysiology*, vol. 90, pp. 5864, 1994.
- [24] J.C.Mosher, R.M. Leahy and P.S. Lewis, "EEG and MEG: Forward solutions for inverse methods", *IEEE Transactions on Biomedical Engineering*, vol.46, 1999, pp.245-259.
- [25] S. Baillet, J.C. Mosher and R.M. Leahy, "Electromagnetic brain mapping", *IEEE Signal processing magazine*, November 2001, pp.14-30.
- [26] C. Michel, M. Murray, G. Lantz, S. Gonzalez, L. Spinelli and R. Grave de Peralta, "EEG source imaging", *Clinical Neurophysiology*, vol.115, 2004, pp. 2195-2222.
- [27] Pascual-Marqui and Roberto Domingo, "Review of methods for solving the EEG inverse problem", *International Journal of Bioelectromagnetism*, 1999, pp.75-86.
- [28] C. d'Alessandro, N. d'Alessandro, S. Le Beux, J. Simko, F. Cetin and H. Pirker, "The speech conductor : gestural control and synthesis" In *proceedings, eNTERFACE'05*, Mons, Belgium
- [29] <http://www.tele.ucl.ac.be/~jjfil/EEGTexture>
- [30] W. S. Yeo and J. Berger., "Application of Image Sonification Methods to Music", in *Proceedings of the 2005 International Computer Music Conference (ICMC 2005)*, Barcelona, Spain, September 2005.
- [31] <http://www.tele.ucl.ac.be/~jjfil/enhancedDidge>

Emotion Detection in the Loop from Brain Signals and Facial Images

Arman Savran¹, Koray Ciftci¹, Guillaume Chanel², Javier Cruz Mota³, Luong Hong Viet⁴, Bülent Sankur¹, Lale Akarun¹, Alice Caplier⁵ and Michele Rombaut⁵

¹Bogazici University, ²University of Geneva, ³Universitat Politècnica de Catalunya, ⁴The Francophone Institute for Computer Science, ⁵Institut National Polytechnique de Grenoble

arman.savran@boun.edu.tr, rciftci@boun.edu.tr, javicm@gmail.com, lhviet@ifi.edu.vn,
guillaume.chanel@cui.unige.ch, bulent.sankur@boun.edu.tr, caplier@lis.inpg.fr, akarun@boun.edu.tr

Abstract—

In this project, we intended to develop techniques for multimodal emotion detection, one modality being brain signals via fNIRS, the second modality being face video and the third modality being the scalp EEG signals. EEG and fNIRS provided us with an “internal” look at the emotion generation processes, while video sequence gave us an “external” look on the “same” phenomenon.

Fusions of fNIRS with video and of EEG with fNIRS were considered. Fusion of all three modalities was not considered due to the extensive noise on the EEG signals caused by facial muscle movements, which are required for emotion detection from video sequences.

Besides the techniques mentioned above, peripheral signals, namely, respiration, cardiac rate, and galvanic skin resistance were also measured from the subjects during “fNIRS + EEG” recordings. These signals provided us with extra information about the emotional state of the subjects.

The critical point in the success of this project was to be able to build a “good” database. Good data acquisition means synchronous data and requires the definition of some specific experimental protocols for emotions elicitation. Thus, we devoted much of our time to data acquisition throughout the workshop, which resulted in a large enough database for making the first analyses. Results presented in this report should be considered as preliminary. However, they are promising enough to extend the scope of the research.

Index Terms—Emotion detection, EEG, video, near-infrared spectroscopy

I. INTRODUCTION

Detection and tracking of human emotions have many potential applications ranging from involvement and attentiveness measures in multimedia products to emotion-sensitive interactive games, from enhanced multimedia interfaces with more human-like interactions to affective computing, from emotion-sensitive automatic tutoring systems to the investigation of cognitive processes, monitoring of attention and of mental fatigue.

The majority of existing emotion understanding techniques is based on a single modality such as PET, fMRI, EEG or static face image or videos. The main goal of this project was to develop a multimodal emotion-understanding scheme using hemodynamic brain signals, electrical brain signals and face images. Studies about the way to fusion the different modalities was also an important goal of the work.

Psychologists agree that human emotions can be categorized into a small number of cases. For example, Ekman et al. [1] found that six different facial expressions (fearful, angry, sad, disgust, happy, and surprise) were categorically recognized by humans from distinct cultures using a standardized stimulus set. In other words, these facial expressions were stable over races, social strata and age brackets, and were consistent even in people blind by birth.

Nevertheless, there are several difficulties in automatic human emotion identification. First, the straightforward correlation of emotions with neural signals or with facial actions may not be correct since emotions are affected by interactions with the environment. As a result, the unfolding of emotions contains substantial inter-subject and intra-subject differences, even though the individuals admit or seem to be in the claimed emotional

This report, as well as the source code for the software developed during the project, is available online from the eNTerFACE'06 web site: www.enterface.net.

situation. Moreover, to design experiments to single out a unique emotion is a very challenging task. These imply that, even small changes in the experimental setup may lead to non-negligible differences in the results.

The majority of existing emotion understanding techniques is based on a single modality such as PET, fMRI, EEG or static face image or videos. The main goal of this project was to develop a multimodal emotion-understanding scheme using functional, physiological and visible data. As an intermediate step, it was necessary to determine the feasibility of fusing different modalities for emotion recognition. These modalities are functional Near Infrared Spectroscopy (fNIRS) electroencephalogram (EEG), video and peripheral signals. Note that these modalities provide us with different aspects of the “same” phenomenon. fNIRS and EEG try to detect functional hemodynamic and electrical changes, peripheral signals give an indication of emotion-related changes in the human body and video signal captures the “visible” changes caused by emotion elicitation.

In the rapidly evolving brain-computer interface area, fNIRS (functional Near Infrared Spectroscopy) represents a low-cost, user-friendly, practical device for monitoring the cognitive and emotional states of the brain, especially from the prefrontal cortex area. fNIRS detects the light (photon count) that travels through the cortex tissues and is used to monitor the hemodynamic changes during cognitive and/or emotional activity.

The second modality to estimate cortical activity is EEG. Using the scalp electrodes, useful information about the emotional state may be obtained as long as stable EEG patterns on the scalp are produced. EEG recordings capture neural electrical activity on a millisecond scale from the entire cortical surface while fNIRS records hemodynamic reactions to neural signals on a seconds scale from the frontal lobe. In fact, electrical activity takes place in order of milliseconds, whereas hemodynamic activity may reach its peak in 6-10 seconds and may last for 30 seconds. In addition to these modalities, peripheral signals, namely, galvanic skin response (GSR), respiration and blood volume pressure (from which we can compute heart rate) were also recorded.

We have combined these four monitoring modes of emotions in two separate pairs, namely: i) fNIRS, ii) EEG, iii) peripheral signals, iv) image or video. Notice that EEG is very sensitive to electrical signals emanating from facial muscles while emotions are being expressed, hence EEG and video modalities cannot coexist. In

contrast, fNIRS is the modality that can be combined with either video signals or with EEG signals.

In summary, the first short-term goal of the project has been to build a reliable database that can be used for all related future research. The second such goal was to prove the viability of a multi-modal approach to emotion recognition, both from instrumentation and signal processing points of view. The final long-term aim is to build an integrated framework for multi-modal emotion recognition for both brain research and affective-computing aspects.

II. MEASUREMENT SETUP AND EMOTION ELICITING

A. Instrumental Setup

To detect and estimate emotions based on brain as well as physiological signals the following sensor setup was prepared: (Figure 1):

- fNIRS sensor to record frontal brain activity,
- EEG sensor to capture activity in the rest of the brain,
- Sensors for acquiring peripheral body processes: a respiration belt, a GSR (Galvanic Skin Response) and a plethysmograph (blood volume pressure)

All these devices were synchronized using a trigger mechanism. Notice that EEG and fNIRS sensor arrangements partially overlap, so that there is no EEG recording on the front. Similarly the fNIRS device covers the eyebrows, occluding one of the image features for emotion recognition

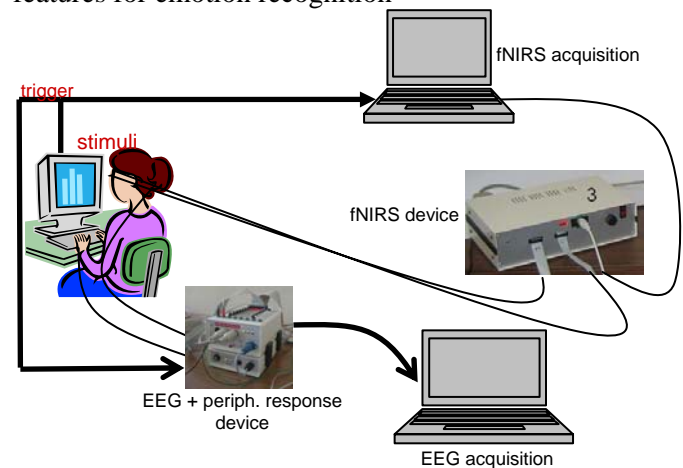


Figure 1 Schematics of EEG and fNIRS acquisition.

The Video-fNIRS acquisition scenario is composed of three computers, *Stimulus Computer*, *fNIRS Computer* and *Video Computer*, each one with the following purpose (Figure 2):

- **Stimulus Computer** shows recorded stimuli to the subjects, sends synchronization signal via the parallel port to the *fNIRS Computer* and stores stimuli start and end instants in a log file.
- **fNIRS Computer** acquires fNIRS data from the fNIRS device.
- **Video Computer** acquires video data from a Sony DFW-VL500 camera.

Synchronization becomes a critical issue when more than one modality is to be recorded, especially when they are recorded on different computers. We have used two synchronization mechanisms: In the first mechanism, the *Stimulus Computer* sends a signal to the *fNIRS Computer* each time a stimulus is shown in the screen via the parallel port. In the second mechanism, the *Stimulus Computer* writes to a log file the instants, with millisecond precision, of each stimulus. This log file is used after recording in the *Video Computer* to mark the frames corresponding to each stimulus. Before the recording process, the *Stimulus Computer* and the *Video Computer* clocks are synchronized using a free internet NTP server localized in Zagreb (ri.ntp.carnet.hr).

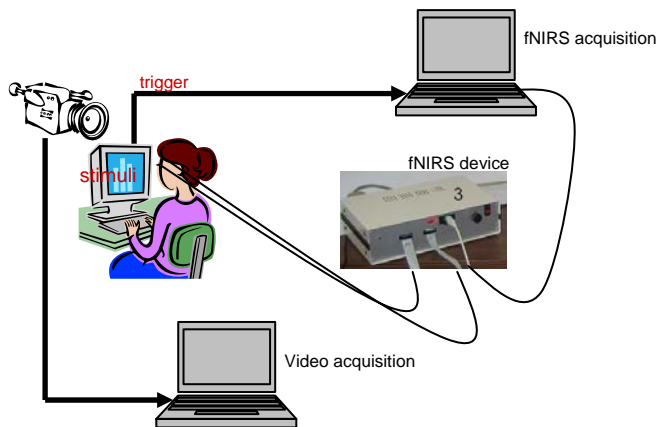


Figure 2 Schematics of Video and fNIRS acquisition.

B. Emotion Eliciting Images

The emotions were elicited in subjects using images from the IAPS (International Affective Picture System) 9. Several studies have shown the usefulness of images to elicit emotional responses that trigger discriminative patterns in both the central and peripheral nervous system (10, 11). The IAPS contains 900 emotionally evocative images evaluated by several American

participants on two dimensions of nine points each (1-9): valence (ranging from positive to negative or unpleasant to pleasant) and arousal (ranging from calm to exciting). The mean and variance of participant judgments for both arousal and valence are computed from these evaluation scores.

We chose images from the IAPS that corresponded to the three emotional classes we wanted to monitor: calm, exciting positive and exciting negative. This was performed by first selecting pictures from IAPS values (1) and then eliminating particular images based on redundancy or particularity of context (for example erotic images were removed). This selection resulted in 106, 71, and 150 pictures respectively for these classes. The selection of the three images subsets, corresponding to the emotional states of interest was instrumented via empirical thresholds on valence and arousal scores:

$$\begin{aligned}
 \text{calm: } & \overline{\text{arousal}} < 4; \quad 4 < \overline{\text{valence}} < 6 \\
 \text{positive exciting: } & \overline{\text{valence}} > 6.8; \\
 & \text{Var}(\text{valence}) < 2; \\
 & \overline{\text{arousal}} > 5 \\
 \text{negative exciting: } & \overline{\text{valence}} < 3; \quad \overline{\text{arousal}} > 5
 \end{aligned} \tag{1}$$

C. Experimental Protocol for fNIRS, EEG and Peripheral Signals

The stimuli to elicit the three target emotions were the above selected images from the IAPS. During the experiment, the subject is seated in front of the computer screen his/her physiological responses (i.e.: fNIRS, EEG and peripheral activity) are being measured. The stimuli are brought to the screen in random order. The subject is asked to watch the images and be aware of his emotional state. In this study, we recorded data from five subjects using the Biosemi Active 2 acquisition system with 64 EEG channel and the peripheral sensors. Due to occlusion from fNIRS sensor arrangement, we had to remove the following ten frontal electrodes: F5, F8, AF7, AF8, AFz, Fp1, Fp2, Fpz, F7, F6, which left us with 54 channels. All EEG signals were recorded at 1024 Hz sampling rate except the first session of participant 1 that was recorded at 256 Hz.

The protocol is detailed in Figure 3 each stimulus consists of a block of five pictures from the same class, this to insure stability of the emotion over time. Each picture is displayed on the screen for 2.5 seconds leading to a total of 12.5 seconds per block. Blocks of different classes are displayed in random order to avoid participant habituation. A dark screen precedes each

block with a cross in the middle to attract user attention and as a trigger for synchronization. The exhibition of the five block images is followed by a dark screen for 10 seconds in order for the fNIRS signals to return to their baseline level.

Emotions are known to be very dependent on past experience so that one can never be very sure whether a block elicits the expected emotion or not. To avoid this problem, we asked the participants to self-assess their emotions after the dark-screen resting period, by giving a score between 1 and 5 for respectively valence and arousal components. This reflection period is not time-limited, which in addition has the benefit of providing an interval for relaxing and/or stretching the muscles.

Self-assessment of the images is a good way to have an idea about the emotional stimulation “level” of the subject. However, since noting down this evaluation necessitates some movements in the subject and enforces an additional prefrontal activity in the brain, some time should elapse for the brain to return to “baseline” before the next image stimulus is offered.

Because of their tight placement, EEG and fNIRS devices can cause some discomfort after a while. For this reason, the whole experiment was divided into three sessions of approximately 15 minutes each. Each session contained 30 blocks, hence 150 images; therefore an experiment consists of a total of 90 blocks or 450 images displayed. The calm and exiting positive classes, containing less than the target number of images were completed with random duplications in different sessions.

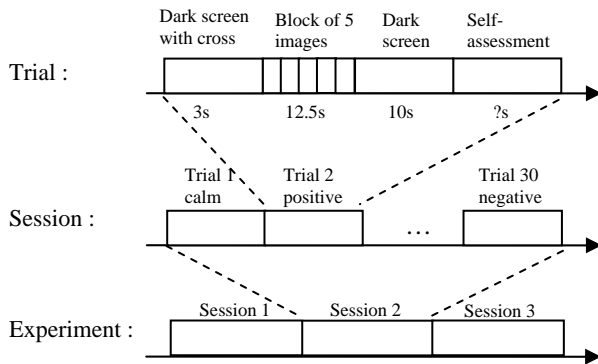


Figure 3 Protocol description

D. Experimental Protocol for Video and fNIRS

Three kinds of emotions, namely neutral, happiness and disgust, are stimulated using series of images and video sequences on the screen of the *Stimulus Computer*.

With this purpose, two protocols have been tested during the recordings.

The first protocol, the one used in *Session 1*, consisted of 5 videos for each emotion from the DaFEx Database, separated with a 20 seconds of a fixation cross (a white cross over a black background). The second protocol, used in *Session 2*, was an improvement of the first protocol. It was noticed that the videos were not enough to make the subjects feel the emotions. In order to make the subject to feel the expected emotion better, a sequence of 5 images collected from the internet were added before the first video of the “happy” and “disgust” sequences.

III. DATABASE COLLECTION

A. Video and fNIRS database

In the Video-fNIRS database there are totally 16 subjects. While one experiment session is performed for 10 subjects, two sessions of experiments in different days are carried out for the other six subjects. There are six women and 10 men subjects with the average age 25 in the database.

The structure of the database is designed in order to make the video post-processing as easy as possible. Video data are recorded frame by frame into separate files. Each filename is formed by subject name, date, time and the stimuli type as follows:

SubjectName-YYYYMMDD-HHMMSS-
FFF_STIMULI.jpg

where these characters denote:

- YYYYMMDD: year in four digit format and month and day in two digit format
- HHMMSS: time expressed in hour, minutes and seconds using 24 hours format
- FFF: milliseconds
- STIMULUS: type of stimulus (happy, disgust, neutral) shown to the subject when the frame was recorded



Figure 4 The structure of the database

Moreover, frames are stored in different folders depending on the type of stimulus. Under these conditions, a subject with name Arman recorded in session 1 on the 1st of August would have a folder in the video database with the structure shown in Figure 4, where the file names of the three sample frames are:

Arman-20060801-182136-599_DISGUST.jpg,
Arman-20060801-182641-586_HAPPY.jpg and
Arman-20060801-181726-131_NEUTRAL.jpg.

The frames corresponding to the cross sign, at the beginning of each recording block, are marked as NOTHING since the data is not related to any emotional state.

B. EEG + fNIRS recordings

We recorded data from five participants all male, and right handed, with age ranging from 22 to 38. For each subject data are divided in three repertories, one per session. For each session we obtained three files categories: one concerns EEG and peripheral information, another concerns fNIRS information and the last contains self-assessments of participants.

EEG and peripheral data

EEG, peripheral and the trigger signals are stored in the same BDF (Biosemi Data Format) file. This format is quite the same as the EDF (European Data Format) so that most software could use it without problems; however you can find a converter from BDF to EDF at <http://www.biosemi.com/download.htm>.

Remember that a trigger is sent in the beginning of each block of images as well as for the start of the protocol.

For more convenience, we extracted the samples where such a trigger appears and save them as markers in a MRK file, except for the first trigger.

Finally we obtained two files: a BDF file with EEG and peripheral signals, and a MRK file containing index of samples for each block of images. These files are named as follow:

PARTA_IAPS_SESB_EEG_fNIRS_DDMMAAAA.bdf
PARTA_IAPS_SESB_EEG_fNIRS_DDMMAAAA.bdf
.mrk

where A is the participant number (1-5), B is the session number (1-3) and DDMMAAAA represents the date of the recording.

Common data

In this section, we describe the files that are common to both modalities and concern the protocol in itself:

- IAPS_Images_EEG_fNIRS.txt, contains three columns, one per session, with the names of the IAPS pictures used in this study;
- IAPS_Eval_Valence_EEG_fNIRS.txt and IAPS_Eval_Arousal_EEG_fNIRS.txt contains in three columns the valence or arousal value for each image;
- IAPS_Classes_EEG_fNIRS.txt list in three columns the associated classes we considered for each block of pictures. Labels can be “Calm”, “Pos” or “Neg”. This can be useful if one does not want to take into account self-assessment of participants.

PartASESB.log lists self-assessment of participants. As for the EEG files, A is the number of the participant while B is the session number

C. fNIRS data

fNIRS data were stored in ASCII format with the file name,

SubjectA_SesB_EEG_fNIRS_DDMMAAAA.txt

for EEG + fNIRS recordings and

SubjectA_SesB_video_fNIRS_DDMMAAAA.txt

for video + fNIRS recordings.

where A is the participant number and B is the session number.

Note that, these files contain raw data, i.e., time-series of concentration changes for three wavelengths. A MATLAB program (loadnirs.m) is needed to convert this signal to oxygenated and deoxygenated hemoglobin values.

D. Practical considerations and problems

The most challenging task was making recordings simultaneously from different devices. Each device was designed to be used alone, and thus were not very suitable for multimodal recordings. For instance, EEG cap and fNIRS probe were clearly obstructing each other's functioning. Thus a special probe should be designed which may hold both EEG electrodes and fNIRS light emitting diode and detectors.

Synchronization was the most time consuming task during the workshop. It took a long time before we arrived at a reasonable and reliable solution for synchronizing the devices.

Deciding on the protocol was perhaps the most critical issue in this study. We used a well-known database for EEG and fNIRS recordings, but we tried to adapt it for our purposes. For video and fNIRS, we actually wanted from the subjects to mimic what they saw. Thus it may be argued whether “mimicking” was the same with “feeling” or not.

For video and fNIRS recordings, it is clear that facial muscle movements caused some noise for fNIRS signals.

We could not have the chance to perform the recordings in an isolated experiment room. Thus, environmental noise definitely corrupted our recordings.

During EEG and fNIRS recordings many participants reported that they had a headache at the end of each session. This is due to the different caps that become more and more uncomfortable along time. More over, they also reported that they never felt some strong positive response while they found negative images a bit too hard. Several participants claimed that the effects of the emotional stimuli decrease after viewing many images in succession, suggesting that they became accustomed to the emotional content.

IV. BRAIN SIGNAL ANALYSIS TECHNIQUES

A. EEG Analysis Techniques

Prior to extracting features from EEG data and performing classification, we need to pre-process signals

to remove noise. Noise can originate from several sources: environment (mainly 50Hz), muscles activity and fNIRS noise. The environmental noise is the easiest to remove by applying a bandpass filter in the 4-45 Hz range. This band is selected because the frequency intervals of interest in EEG are the θ (4-8Hz), α (8-12Hz), β (12-30Hz) and γ (30-45Hz) bands. Muscle activities such as eye-blinks or jaw clenching contaminate EEG signals with strong artifacts. In this study, no special effort was done to remove these artifacts, but subjects were requested to avoid these movements during recordings. One unexpected source of contamination was the fNIRS light activations. As can be observed in Figure 5 fNIRS light activations cause spikes in the EEG recordings, especially in the frontal area. For the moment, no appropriate filtering was designed to remove this type of noise, though independent component analysis (ICA) technique is one potential tool. Finally, to obtain some better focalization on brain activity, we computed a Laplacian reference signal, which consists in subtracting for each electrode the mean signal of its neighbors.

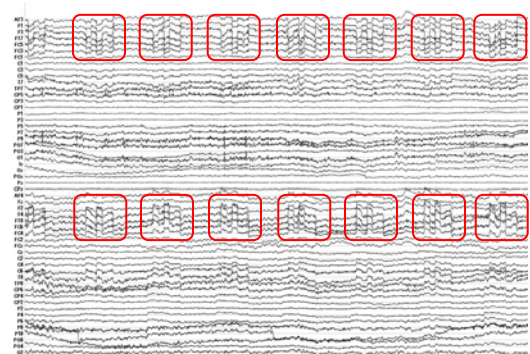


Figure 5 EEG signal sample after pre-processing. fNIRS noise can be observed approximatively every 700ms especially on the frontal electrodes (in red)

Following the preprocessing stage, there are various alternatives for feature extraction. One alternative is to collect EEG energies at various frequency bands, time intervals and locations in the brain. This approach results typically in oversized feature vectors. As a second alternative, Aftanas & al. [10] proved the correlation between arousal variation and power in selected frequency bands and electrodes. These features have also been used in 12 to assess the arousal dimension of emotions. In this project we opted for this first set of features. A third possibility is to compute the STFT (Short Term Fourier Transform) on 12.5 second segments of each trial and electrode, assuming stationarity of the signal within the chosen widow length. This allows taking into account time evolution as

well as spatial distribution of energy. Each atom resulting from the STFT is then considered as a feature or relevant features can be selected by filter or wrapper methods [13].

B. Peripheral Signals Analysis Techniques

Several studies have shown the effectiveness of peripheral sensors in recognizing emotional states (see 12, 13, 15). While there are many variables from the autonomous nervous system that can be used to determine affective status, we will focus to three such variables: GSR, respiration and blood volume pressure. All these signals were first filtered by a mean filtering to remove noise

GSR provides a measure of the resistance of the skin. This resistance can decrease due to an increase of sudation, which usually occurs when one is feeling an emotion such as stress or surprise. Lang [11] also demonstrates correlation between mean GSR level and arousal. In this study, we recorded GSR by positioning two dedicated electrodes on the top of left index and middle fingers. In order to assess the change in resistance, we used the following features:

Value	Comment
Mean skin resistance over the whole trial	Estimate of general arousal level
Mean of derivative over the whole trial	Average GSR variation
Mean of derivative for negative values only	Average decrease rate during decay time
Proportion of negative samples in the derivative	Importance and duration of the resistance fall

The mean value of samples within a session gives us an estimate of the general arousal level of the emotion while the mean derivative reveals the variability of the signal. Computing the mean of derivative for negative values only, or the proportion of negative values for the whole session indicates the importance of the fall in resistance.

Respiration was recorded by using a respiration belt, providing the chest cavity expansion over time. Respiration is known to correlate with several emotions [13]. For example slow respiration corresponds to relaxation while irregularity or cessation of respiration can be linked to a surprising event. To characterize this activity we used features both in the time and frequency domain. In the frequency domain we computed energy by FFT (Fast Fourier Transform) in 10 frequency bands of size $\Delta f = 0.25$ ranging from 0.25Hz to 2.75Hz. Others features are listed below: (

Value	Comment
Power in the 0.25Hz-2.75Hz ($\Delta f = 0.25$ Hz) bands (10 features)	-
Mean of respiration over the whole trial	Average chest expansion
Mean of derivative over the whole trial Standard deviation	Variation of respiration signal
Maximum value minus minimum value	Dynamic range or greatest breath

Finally, a plethysmograph was placed on the thumb of the participant to record his blood volume pressure. This device permits to analyze both relative vessel constriction, which is a defensive reaction [13], and heartbeats that are clearly related to emotions especially in terms of heart rate variability (HRV) (see 11, 13, 15). Heart beats were extracted from the original signal by identification of local maxima, and then the BPM (Beat Per Minute) signal was computed for each inter-beat periods i . This enables us to approximate HRV using standard deviation or mean derivative of the BPM signal. The following features were extracted from blood volume pressure:

Value	Comment
Mean value over the whole trial	Estimate of general pressure
Mean of heart rate over the whole trial	-
Mean of heart rate derivative Standard deviation of heart rate	Estimations of heart rate variability

Finally, all these features were concatenated in a single features vector of size 22, representing the peripheral activity.

C. fNIRS Analysis Techniques

fNIRS provides us with time series of oxygen-rich (HbO₂) and oxygen-poor (Hb) blood concentration changes on the cortical surface. fNIRS signals should be preprocessed first to eliminate high frequency noise and low frequency drifts. Previous studies have shown that involvement of prefrontal cortex in the emotion processing is concentrated in the medial frontal cortex. Thus, it may be a good choice to concentrate on the middle 8 detectors.

Since the hemodynamic response mainly gives an idea about the area of activation, the first line of action has been to detect the presence of active regions in the brain and their variation with stimuli. On the other hand, activated regions are known to vary from subject to subject, and even within subject in the course of experiments. It follows that detection schemes based only on single subject data may not be reliable enough. One solution to this problem is the use of multivariate methods, that is, simultaneous processing and modeling of data from a group of subjects. Some well-known examples are principal component analysis and independent component analysis. This type of methods may give us the emotion-related components.

The noise caused by facial muscle movement aroused as an important source of contamination for fNIRS signals. Since for some detectors this noise is so large with respect to the signal, it is (and will be) hard to extract cognitive and emotional component from the signals.

D. Fusion techniques

The main fusion strategies are data-level fusion, feature-level fusion and decision-level fusion. Due to the disparity of the nature of data in the three modalities, data fusion is not conceivable. On the other hand, fusion at the more abstract levels, feature level and decision level, are both feasible and desirable.

fNIRS-EEG fusion: Recall that the link between electrical activity and hemodynamic activity is supplied by the neurocoupling mechanisms. The EEG modality in one part and the video or fNIRS modality on the other part, have orders of magnitude difference in their relative time scales. However, feature/decision level fusion is possible if one generates fNIRS and EEG feature vectors and/or decision scores for each block of emotional stimuli (12.5 seconds long in our experiment). Alternatively, video features and fNIRS features can be fused at the feature or decision level on a block-by-block basis.

V. VIDEO BASED EMOTION DETECTION

Video signals are quite rich in facially expressed emotions, especially for the happiness and disgust cases. Facial expressions are formed by motions or deformations of mouth, eyebrows and even of eyelids. Also, facial skin may get deformed, such as wrinkles in the forehead or inflations on the cheeks. In this particular experiment, however, we do not have access to the eyebrow information due to the occlusion by the fNIRS probe on the forehead (Figure 6).

We have therefore extracted facial features from mouth and eyes, and then analyzed and classified the data as in the block diagram of Figure 6. We used comparatively two methods for facial feature segmentation: active contour-based technique [3, 4] and active appearance models (AAM) [8]. For the classification we are using Transferable Belief Model (TBM) method [2, 7].

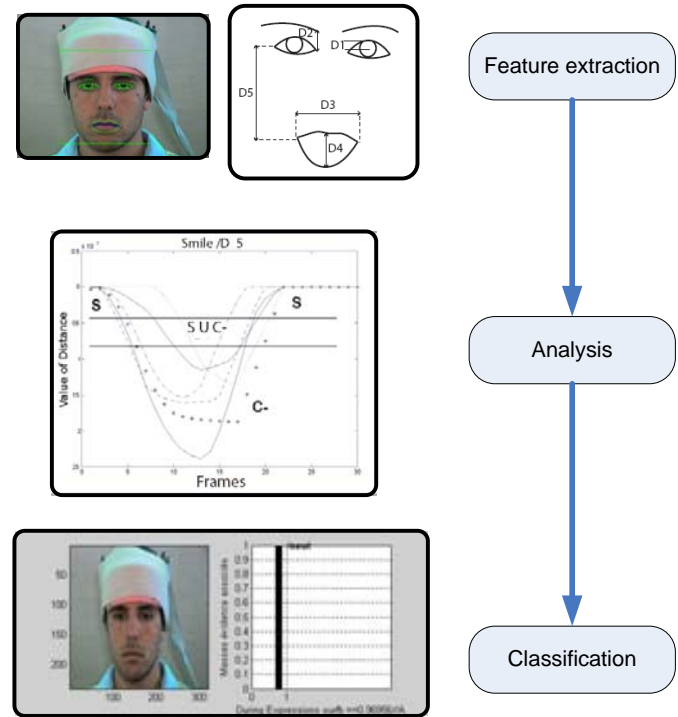


Figure 6 Illustration of the feature extraction, analysis and classification system for emotion detection.

A. Active Contours for Facial Feature Extraction

Active contours are widely used for segmentation purposes. However first, the face itself and the eyes must be located. We have used the detector in the Machine Perception Toolbox (MPT) [5]. We had to execute the face detection in each image and bypass its tracking ability due to stability problems. This in turn, slows down the process. Wherever MPT cannot detect a face, we recurred to the OpenCV library face detection tool. The OpenCV algorithm detects faces in general with higher precision albeit at lower speeds than the MPT.

Following fiducial point localization, lips, eyes and eyebrows are segmented by fitting curves automatically and frame-by-frame, using the algorithm described in [3, 4]. This algorithm uses a specific predefined parametric model such that all the possible deformations can be taken into account. The contours are initialized by extracting certain characteristic points, such as eye corners, mouth corners and eyebrows corners automatically. In order to fit the model to the contours, a

gradient flow (of luminance or of chrominance) through the estimated contour is maximized. As remarked above, the model fitting to the eyebrows and mouth was not satisfactory, the first due to the occluding fNIRS probe and the latter whenever there were beard and moustache (Figure 7). Even in the absence of such impediments, we have observed that this algorithm works well only when the facial images are neutral (open eyes and closed mouth). Finally, the tracking mode of this algorithm was not available during the workshop. Therefore, we applied the algorithm described in the next section in order to have working results.

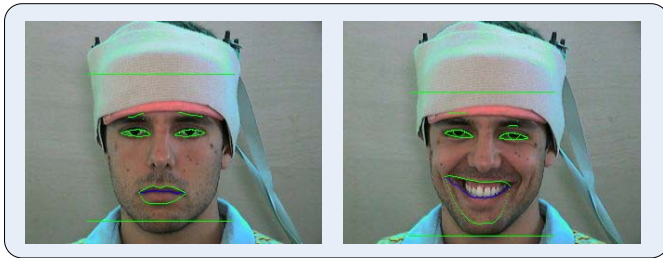


Figure 7 Examples of correct (left) and incorrect (right) localizations

B. Active Appearance Models (AAM)

Using Active Appearance Models (AAM) is a well-known technique for image registration [8], in which statistical models of appearances are matched to images iteratively by modifying the model parameters that control modes of shape and gray-level variation. These parameters are learned from a training dataset. The training of the AAM algorithm is initiated with manually annotated facial images, as illustrated in Figure 8. 37 landmark points are chosen from the easily identifiable locations on the face, and their 2D coordinates constitute the shape vector for the face images. In this study, they are chosen appropriately to cover the face, mouth and eye regions for eventual segmentation of face contour, lips and eyes. After creating the shape vectors from all of the training images, they are aligned in a Euclidean frame by Procrustes algorithm. Finally, principal component analysis (PCA) technique is applied to reduce the dimension of the shape vectors, resulting in 11 modes of shape variation that account for 95 percent of variance in the training set.

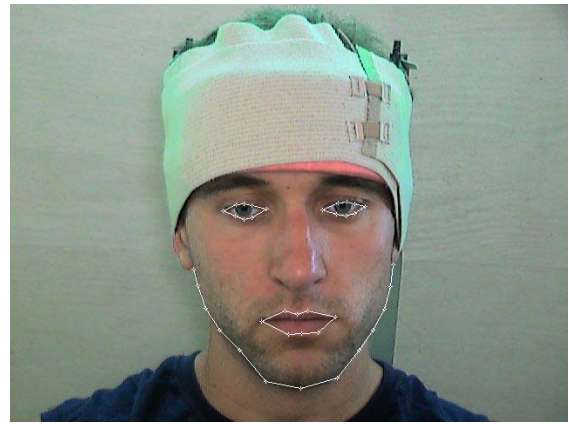


Figure 8 Annotated image with 37 landmark points

The next step is to create the texture vectors for the training images. For this purpose, Delaunay triangulation (Figure 9) is performed so that the triangular patches are transformed to the mean shape (Figure 10). Thus, shape-free patches of pixel intensities are obtained. Also, to diminish the effect of lighting differences, a further alignment in the gray level is performed. Finally, PCA is applied to the texture vectors as in the shape vectors.



Figure 9 Delaunay triangulation



Figure 10 Texture image after transforming to mean shape

The third and final step of the algorithm is to combine shape and texture vectors. These two types of vectors are concatenated into one big feature vector, after appropriate weighting. These weighting coefficients are obtained using an estimation procedure. A final PCA is applied to this combined vector, the resulting vectors being called the appearance vector.

The goal of AAM is to fit models to the images and to synthesize various facial appearances. This is accomplished by modifying and optimizing the combination weights. In face modeling, best pose parameters, which are planar translations, rotations and scaling, are estimated. Briefly, this optimization is realized by iteratively minimizing the difference between the input image pixel intensities and the model instances.

In this study, an AAM is trained and tested for a subject. Some sample results are given in Figure 11, where the detected facial contours corresponding to happiness and disgust moods are tracked in video. For this subject total 26 training images were chosen from the training video database. These images were chosen in order to include different neutral, disgust and happiness expressions with varying head pose and eye motion to cover sufficient amount of face motion.

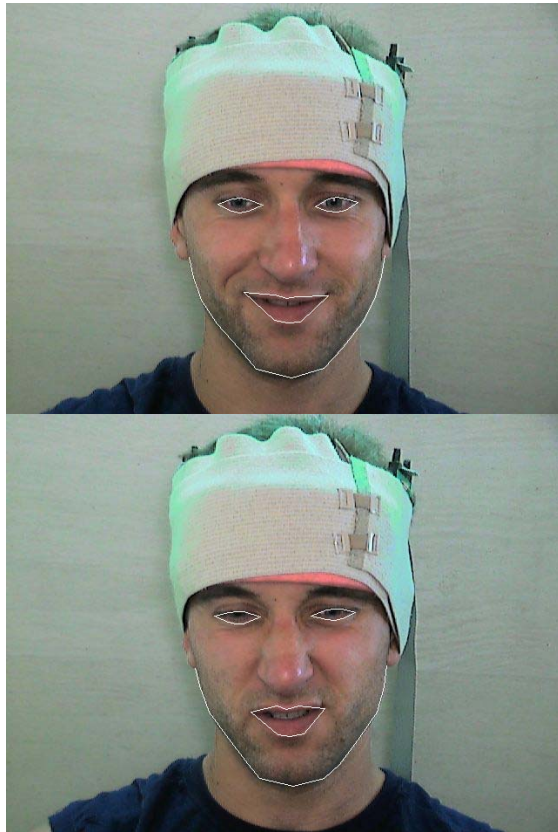


Figure 11 Detected facial contours in video corresponding to happiness (above) and disgust (bottom)

C. Classification

In this study, Transferable Belief Model (TBM) algorithm [7], which is based on belief theory, is applied for the classification of anger, disgust and neutral expressions. First, some facial distances, as illustrated in Figure 6, are calculated from the extracted contours. These distances are: eye opening (D_1), distance between the inner corner of the eye and the corresponding corner of the eyebrow (D_2), mouth opening width (D_3), mouth opening height (D_4), distance between a mouth corner and the outer corner of the corresponding eye.

Briefly, in the TBM algorithm each facial expression is characterized by a combination of symbolic states, which are evaluated from the calculated distances. For distance D_i , the symbolic state is found by thresholding. In Figure 12, the representation of the symbolic states $\{C+, C-, S, SC+, SC-\}$ and the thresholds (a, b, c, d, e, f, g, h) are shown. While states $C+$, $C-$, S are representing positive activation, negative activation and no activation, respectively, other two states are denoting the doubt between activation and no activation. The threshold values are found in a training phase automatically as described in [7]. In Figure 12, the y-axis shows the piece of evidence (PE) according to the belief theory. After having found the states and the PEs for each symbolic state of each distance, conjunctive combination rule, which is explained in [7], is applied to combine the information coming from each distance. With this rule, combined PE for each expression is calculated, and the decision is made by choosing the expression that gives highest value for the combined PE. Details about the fusion process can be found in [7].

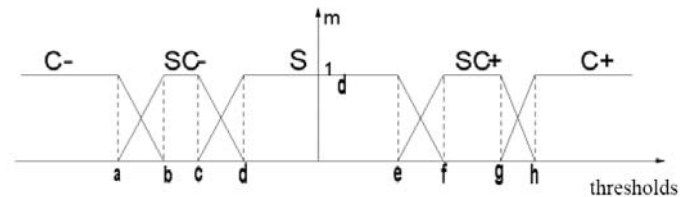


Figure 12 Characterization of a distance with thresholds

VI. CONCLUSIONS AND FUTURE WORK

Project 7 entitled “Emotion Detection in the Loop from Brain Signals and Facial Images “ at Dubrovnik eNTERFACE had three goals in mind:

- i) Common database building
- ii) Interest and feasibility of these modalities
- iii) Assessment of emotion detection performance of individual modalities and their fusion.

Briefly, we have made significant progress in goals 1 and 2, while the 3rd goal must be revisited in a future project.

Common database building: A considerable database containing video, fNIRS and EEG signals has been built. We have already mentioned in Section 6 about the incompatibility of video and EEG. As a consequence, two separate databases were built, one encompassing EEG (including physiological signals) and fNIRS modalities, and the other encompassing video and fNIRS modalities. Second, there was an unpredicted interference effect between EEG and fNIRS setups. The elimination of the EEG & fNIRS interference is not insurmountable, though we did not have time to address the problem during the workshop. Third, the critical synchronization problem between the modality pairs has been ingeniously solved in two alternative ways. The fourth issue was the determination of the proper protocols as well as stimulation material. Although we used the standard methods and materials as in the literature, some subjects reported unsure or inadequate stimulation especially during the prolonged experiments. Subject discomfort and fatigue was another aggravating factor.

Interest and feasibility of these modalities: There is increasing interest in literature for emotion detection and estimation in humans. However, there exist separate literatures, one set of papers published in neuroimaging and neural signal processing journals, the other set of papers appearing in computer vision and man-machine interface journals. We believe the joint use of modalities was for the first time addressed in this workshop, as far as the open literature is concerned. Individual modalities do not fair very well in emotion assessment, hence we believe the multimodal approach will certainly improve the classification performance.

Assessment of emotion detection performance of individual modalities and their fusion: This part of the project has not been completed and is left as a future work.

ACKNOWLEDGMENT

The authors gratefully acknowledge all participants that volunteered in studies.

REFERENCES

1. Ekman, P., Levenson, R.W., Friesen, W.V., 1983. Autonomic nervous system activity distinguishing among emotions. *Science* 221, 1208–1210.
2. Smet, PH. Data fusion in the Transferable Belief Model. *Proc ISIF, Frane(2000)* 21-33
3. Eveno, N., Caplier, A., Coulon, P.Y.: Automatic and Accurate Lip Tracking. *IEEE Trans. On CSVT*, Vol. 14. (2004) 706–715.
4. Hammal, Z., Caplier, A. : Eye and Eyebrow Parametric Models for Automatic Segmentation. *IEEE SSIAI, Lake Tahoe, Nevada (2004)*.
5. Machine Perception Toolbox (MPT)
<http://mplab.ucsd.edu/grants/project1/free-software/MPTWebSite/API/>.
6. Open Computer Vision
<http://opencvlibrary.sourceforge.net/FaceDetection>
7. Hammal: Facial Features Segmentation, Analysis and Recognition of Facial Expressions using the Transferable Belief Model 29-06-2006
8. G.J.Edwards, C.J.Taylor, T.F.Cootes, "Interpreting Face Images using Act Active Appearance Models", *Int. Conf. on Face and Gesture Recognition* 1998. pp. 300-305
9. P.J. Lang, M.M. Bradley, and B.N. Cuthbert, "International affective picture system (IAPS): Digitized photographs, instruction manual and affective ratings", *Technical Report A-6*, University of Florida, Gainesville, FL, 2005.
10. L.I. Aftanas, N.V. Reva, A.A. Varlamov, S.V. Pavlov, and V.P. Makhnev, "Analysis of Evoked EEG Synchronization and Desynchronization in Conditions of Emotional Activation in Humans: Temporal and Topographic Characteristics", *Neuroscience and Behavioral Physiology*, 2004, pp. 859-867.
11. J.P. Lang, M.K. Greenwald, M.M. Bradley, A.O. Hamm, "Looking at pictures: affective, facial, visceral, and behavioral reactions", *Psychophysiology*. 1993 May; 30(3), pp. 261-273.
12. G. Chanel, J. Kronegg, D. Granjean, T. Pun, "Emotion assessment: arousal evaluation using EEG's and peripheral physiological signals",

Proc. Int. Workshop on Multimedia Content Representation, Classification and Security, Istanbul, 2006, pp 530-537.

13. G.H. John, R. Kohavi, K. Pfleger, "Irrelevant Features and the Subset Selection Problem", Machine Learning: Proceedings of the 11th International Conference, San Francisco, 1994, pp. 121-129.
14. J. .A Healey, "Wearable and Automotive Systems for Affect Recognition from Physiology", PhD Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2000.
15. Lisetti, F. Nasoz, "Using Non-Invasive Wearable Computers to Recognize Human Emotions from Physiological Signals", Journal on Applied Signals Processing, 2004, pp. 1672-1687.

RAMCESS: Realtime and Accurate Musical Control of Expression in Singing Synthesis

N. D'Alessandro¹, B. Doval², S. Le Beux², P. Woodruff¹ and Y. Fabre¹

¹*TCTS Lab, Faculté Polytechnique de Mons (Belgium)*, ²*LIMSI-CNRS, Université Paris XI (France)*

Abstract—The main purpose of this project is to develop a full computer-based musical instrument allowing realtime synthesis of expressive singing voice. The expression will result from the continuous action of an interpreter through a gestural control interface. That gestural parameters will influence the voice characteristics thanks to particular mapping strategies.

Index Terms—Singing voice, voice synthesis, voice quality, glottal flow models, gestural control, interfaces.

I. INTRODUCTION

EXPRESSIVITY is nowadays one of the most challenging topics in view by the researchers in speech synthesis. Indeed, recent synthesizers provide acceptable speech in term of intelligibility and naturalness but the need to improve human/computer interactions brought researchers to develop more “human”, more expressive systems. Some recent realizations have shown that an interesting option was to record multiple databases corresponding to a certain number of “labelled” expressions (e.g. happy, sad, angry, etc) [1]. At synthesis time, the expression of the virtual speaker is set by choosing the units in the corresponding database.

Last year, during eNTERFACE'05 [2], we decided to investigate the opposite option. Indeed, we postulated that “emotion” in speech was not the result of switches between labelled expressions but a continuous evolution of voice characteristics extremely correlated with context. Thus, we developed a set of flexible voice synthesizers “conducted” in realtime by an operator [3]. After some tests, it was clear that such a framework was particularly efficient for singing synthesis.

Remarkable achievements have been recently reached in singing voice synthesis. A review of state of the art can be found in [4]. Technology seems mature enough for replacing vocals by synthetic singing, at least for backing vocals [5] [6]. However, existing singing synthesis systems suffer from two restrictions: they are aiming at mimicking singers rather than creating new instruments, and are generally limited to MIDI controllers.

We think it worthwhile to extend vocal possibilities of voice synthesizers and design new interfaces that will open new musical possibilities. In a first attempt we decided to restrain our survey on voice quality control to the boundaries of natural voice production. As a matter of fact, it is always better trying to mimic one particular voice, as we are disposed to hear someone behind the synthesizer. This process enables to achieve analysis by synthesis : once we are able to perceive more naturalness in the synthesized voice, this means that we understood something in voice production process. It

is then easier to go astray from these limits when dealing with a musical application in a more creative way.

II. AIMS OF THE WORK

Our aims for this eNTERFACE'06 workshop can be summarized in three main axes. First, we target the implementation of intra- and inter-dimensional mappings driving low-level parameters of source models (e.g. complex interactions between vocal effort and tenseness, represented by the phonetogram). Then, we investigate the effects of the vocal tract in voice quality variations (e.g. the singer formant, lowering of the larynx). Finally, source/filter coupling effects (e.g. relations between harmonics and formants frequencies) are analysed, and several mechanisms are implemented (e.g. overtone, croatian, bulgarian, occidental singing).

III. BACKGROUND IN SINGING SYNTHESIS

Speech and singing both result from the same production system: the voice organ. However, the signal processing techniques developed for their synthesis evolved quite differently. One of the main reasons for this deviation is: the aim for producing voice is different for the two cases. The aim of speech production is to exchange messages. For singing, the main aim is to use the voice organ as a musical instrument. Therefore a singing synthesis system needs to include various tools to control (analyze/synthesize or modify) different dynamics of the acoustic sound produced: duration of the phonemes, vibrato, wide range modifications of the voice quality, the pitch and the intensity, etc. some of which are not needed in most of the speech synthesis systems. A pragmatic reason for that separation is that singing voice synthesizers target almost exclusively musical performances. In this case, “playability” (flexibility and real-time abilities) is much more important than intelligibility and naturalness. Discussions about various issues of singing synthesis can be found in [7] [8].

As described in [9], frequency-domain analysis/modifications methods are frequently preferred in singing synthesis research due to the need to modify some spectral characteristics of actual recorded signals. The most popular application of such a technique is the phase vocoder [10], which is a powerful tool used for many years for time compression/expansion, pitch shifting and cross-synthesis.

To increase flexibility, short-time signal frames can be modeled as sums of sinusoids (controlled in frequency, amplitude and phase) plus noise (controlled by the parameters of a filter which is excited by a white noise). HNM (Harmonic plus

Noise Model) [11] provides a flexible representation of the signal, which is particularly interesting in the context of unit concatenation. That representation of signals is thus used as a basis in many singing synthesis systems [12] [13] [14] [15].

Another approach is to use the source/filter model. Several models of glottal pulse has been proposed with different quality and flexibility. A complete study and normalisation of the main models can be found in [16]. For example, the R++ model has been used in the famous Voicer [17]. LF [18] and CALM [19] models have been used during eNTERFACE'05 [3]. Other differences appear in the method used to compute the vocal tract transfer function. Some systems [20] compute the formants from the magnitude spectrum: a series of resonant filters (controlled by formants frequencies, amplitudes and bandwidths). Some other systems compute an acoustic representation of the vocal tract, as a cascade of acoustic (variant-shape) tubes. For example, the SPASM synthesizer [21] uses digital waveguides [22] to model acoustic features of oral, nasal cavities and throat radiation (driven by a frequency-domain excitation model). The model was extended to variable length conical sections by Välimäki and Karjalainen [23].

There exist also some particular approaches like FOF (*Formes d'Ondes Formantiques*) synthesis [24], used in CHANT [25], which performs synthesis by convolving a pulse train with parallel formant wave functions (time-domain functions corresponding to individual formants resonance).

IV. VOICE PRODUCTION

Voice organ is usually described as a "source/filter" system. Glottal source is a non-linear volume velocity generator where sound is produced by complex movements of vocal folds (larynx) under lungs pressure. A complete study of glottal source can be found in [26]. Sounds produced by the larynx are then propagated in oral and nasal cavities which can be seen as time-varying filtering. Finally, the flow is converted into radiated pressure waves through lips and nose openings (cf. Figure 1).

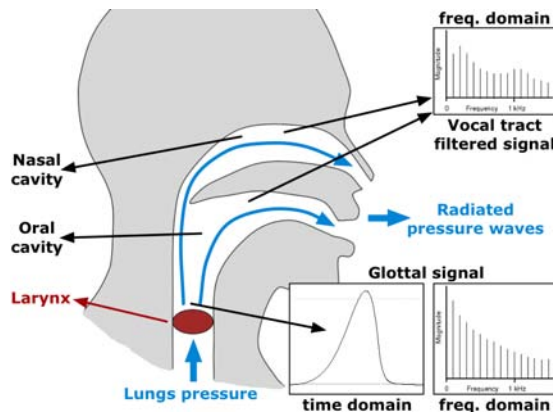


Fig. 1. Voice production mechanisms: vocal folds vibrations, vocal tract filtering and lips/nose openings radiation.

In the context of signal processing applications, and particularly in singing synthesis, some simplifications are usually accepted. First, lips and nose openings effect can be seen

as derivative of the volume velocity signal. It is generally processed by a time-invariant high-pass first order linear filter [27]. Vocal tract effect can be modeled by filtering of glottal signal with multiple (usually 4 or 5) second order resonant linear filters.

Contrary to this "standard" vocal tract implementation, plenty of models have been developed for representation of glottal flow, with differences in accuracy and flexibility. Usual models are KLGLOTT88 [28], R++ [29], Rosenberg-C [30], LF [18] [31] and more recently, CALM [19].

V. THE GLOTTAL SOURCE

In this section, we describe the work related to the realtime generation of the glottal source signal. We first explain our theoretical basics: the modelization of the glottal flow as the response of a causal/anticausal linear system (CALM). Then, we will describe two different implementations achieved during this workshop: a bufferized computation of a causal stable filter (v1.x) and a sample-by-sample computation of a causal unstable filter (v2.x).

A. The Causal/Anticausal Linear Model (CALM) [19]

Modelling vocal tract in spectral domain (with resonant filters central frequency, amplitude and bandwidth) is very powerful in term of manipulation because spectral description of sounds is close to auditory perception. Traditionally, glottal flow has been modeled in time domain. A spectral approach can be seen as equivalent only if both amplitude and phase spectra are considered in the model.

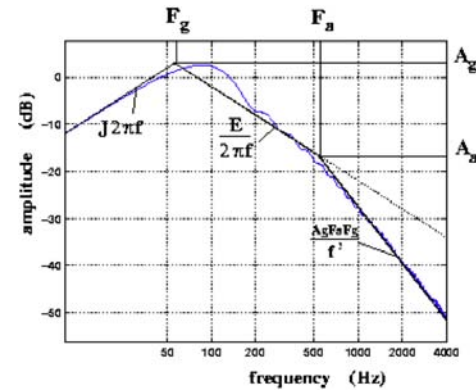


Fig. 2. Amplitude spectrum of the glottal flow derivative: illustration of glottal formant (F_g , A_g) and spectral tilt (F_a , A_a).

For amplitude spectrum, two different effects can be isolated (cf. Figure 2). On the one hand, an amount of energy is concentrated in low frequencies (i.e. below 3 kHz). This peak is usually called "glottal formant". We can see that bandwidth, amplitude and position of the glottal formant change with voice quality variations. On the other hand, a variation of spectrum slope in higher frequencies (called "spectral tilt") is also related to voice quality modifications.

Considering both "glottal formant" and "spectral tilt" effects, two cascading filters can be used. A second order resonant low-pass filter ($H_1(z)$) for glottal formant, and a first order

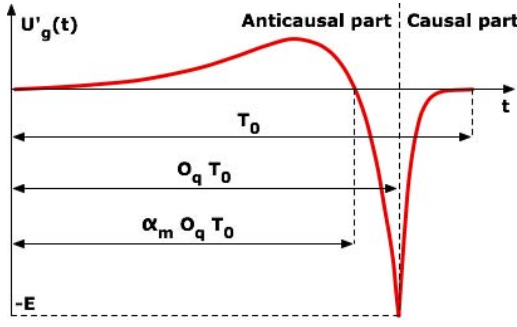


Fig. 3. Time-domain representation of derived glottal pulse: anticausal part and causal part.

low-pass filter ($H_2(z)$) for spectral tilt. But phase information indicates us that this system is not completely causal. Indeed, as it is illustrated on Figure 3, glottal pulse is a combination of a "increasing" (or active) part and a "decreasing" (or passive) part. The decreasing part, called the return phase, mainly influences the spectral tilt and hence is causal. And we can also show that the second order low-pass filter has to be anticausal in order to provide a good phase representation. This information is sometimes referred as the mixed-phase representation of voice production [32].

A complete study of spectral features of glottal flow, detailed in [19], gives us equations linking relevant parameters of glottal pulse (F_0 : fundamental frequency, O_q : open quotient, α_m : asymmetry coefficient and T_L : spectral tilt, in dB at 3000Hz) to $H_1(z)$ and $H_2(z)$ coefficients. Expression of b_1 as been corrected, compared to [19] and [33].

Anticausal second order resonant filter:

$$H_1(z) = \frac{b_1 z}{1 + a_1 z + a_2 z^2}$$

$$a_1 = -2e^{-a_p T_e} \cos(b_p T_e)$$

$$a_2 = e^{-2a_p T_e}, b_1 = E T_e$$

$$a_p = -\frac{\pi}{O_q T_0 \tan(\pi \alpha_m)}, b_p = \frac{\pi}{O_q T_0}$$

Causal first order filter:

$$H_2(z) = \frac{b_{T_L}}{1 - a_{T_L} z^{-1}}$$

$$a_{T_L} = \nu - \sqrt{\nu^2 - 1}, b_{T_L} = 1 - a_{T_L}$$

$$\nu = 1 - \frac{1}{\eta}, \eta = \frac{1}{\cos(2\pi \frac{3000}{F_e}) - 1} \frac{e^{-T_L/10 \log_{10}(10)} - 1}{1}$$

Full anticausal processing is only possible offline, by running algorithms backwards on data buffers. In a realtime context, anticausal response can be processed with two different methods. On the one hand, the response of a causal version of

$H_1(z)$ is stored backwards ($v1.x$). On the other hand, $H_1(z)$ is replaced by a unstable causal filter and the "divergent" impulse response is truncated ($v2.x$). We can also note that in order to be useful our implementations have to be able to produce correct glottal flow (GF) and glottal flow derivative (GFD). Indeed, the GFD is the acoustical signal used to synthesize the voiced sounds, but the GF is important in the synthesis of turbulences, involved in unvoiced and breathy sounds.

B. RealtimeCALM v1.x Implementation

This implementation is the continuation of the development tasks of eNTERFACE'05 [3] and work presented to NIME'06 [33]. In this algorithm, we generate the impulse response by *period-synchronous anticausal processing*. It means that in order to achieve the requested waveform, the impulse response of a causal version of H_1 (glottal formant) is computed, but stored backwards in a buffer. This waveform is truncated at a length corresponding to instantaneous fundamental frequency ($F_0 + Jitter$). This algorithm is now integrated in both Max/MSP [34] [35] and Pure Data [36] external objects (for Mac OS X, Windows and Linux): *almPulse~ v1.x*. Then the resulting period is filtered by H_2 (spectral tilt). This algorithm is also integrated in both Max/MSP and Pure Data external objects: *stFilter~ v1.x*. Coefficients of H_1 and H_2 are calculated from equations described in subsection *The Causal/Anticausal Linear Model (CALM)* and [19]. Thus, both time-domain and spectral-domain parameters can be sent.

Actually, we take advantage of physical properties of glottis to propose this real-time algorithm. Indeed, glottal pulse corresponds to opening/closing movements of vocal folds. It means that impulse responses generated by H_1 and H_2 filters can't overlap. Thus, impulse responses can be stored backwards and truncated period-synchronously without changing too much their spectral properties.

Truncation of the CALM waveform at each period gives quite good synthesis results. Nevertheless, several configurations of parameters (e.g. high value of α_m plus low value of O_q) make the impulse response oscillating inside the period, which gives signals that are no more related to glottal source phenomena and changes voice quality perception. Thus, earlier truncation points and windowing options have been tested (e.g. first zero crossing of the GF, first zero crossing of the GFD). This study has shown us that it is not possible to set a truncation point inside the period which gives simultaneously correct synthesis results on the GF and the GFD (even with a synchronized half-Hanning windowing¹). This modelization problem and limitations due to the use of period buffer drove us to change the architecture of this synthesis module ($v2.x$). Discontinuity in GFD due to GF truncation is illustrated at the Figure 4.

C. RealtimeCALM v2.x Implementation

This part explains another version of the anticausal filter response computation. It avoids the use of period buffer. Main

¹This windowing method multiplies the increasing part of the glottal pulse (flow or derivative) – meaning the part between the zero crossing and the positive maximum – by the left part of a Hanning window.

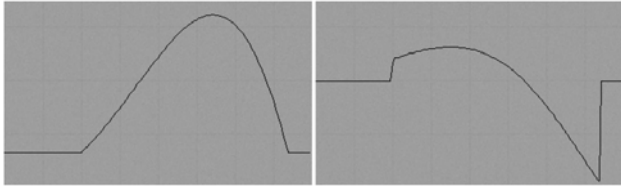


Fig. 4. Discontinuity in GFD (right) due to GF truncation at the first zero crossing of the CALM period (left).

idea behind this solution was to decrease memory allocations, in order to be able to generate simultaneously the glottal flow and the glottal flow derivative, with their own truncation points and windowings².

Instead of computing a causal version of the impulse response offline and then copying it backwards into a fixed buffer, the computation is here straightforward. The iterative equation corresponds indeed to the unstable anticausal filter. Anyway, the explosion of the filter is avoided by stopping the computation exactly at the Glottal Closure Instant (GCI). We can also note that glottal flow and glottal flow derivative can both be achieved with the same iterative equation, only changing the values of two first samples used as initial conditions in the iteration process.

One other main implementation problem is that the straightforward waveform generation has to be synchronized with the standard Pure Data's performing buffer size. This standard size is 64 samples which, at an audio rate of 44100Hz, corresponds to a frequency of approximately 690 Hz. Most of the time, the fundamental frequency of the glottal flow is less than 690 Hz, which means that several buffers are necessary to achieve the complete computation of one period. But whenever a buffer reaches the end, the main performing routine is called and thus the values of a_1 and a_2 have to be frozen as long as the period time has not been reached. A flag related to the opening of the glottis is then introduced, fixed to the value of the period (in samples), and the values of a_1 and a_2 are not changed until this flag is decreased to 0. Once values of T_0 , T_e , γ , a_p , and b_p have been calculated at the opening instant, only a_1 and a_2 have to be frozen, as these are the only variables that are taken into account in the equations of the derivative glottal waveform.

We just tested the glottal flow/glottal flow derivative generation alone, without the addition of any vocal tract. However, strong tests have been carried out concerning this implementation and revealed that this version is more robust than the previous one. In particular, this implementation is not stuck when exotic values are sent to the algorithm. Finally, we can note that this upgrade only concerns the *almPulse~* module. The spectral tilt filtering module (*stFilter~*) was not modified.

²We can observe that our method will change the link between those two waveforms. Indeed, if two separated truncation points and windowings are applied, what we call "glottal flow derivative" is no more the derivative of the glottal flow.

D. Dimensionnal Issues

The next step in the realization of our singing tool was to define perceptual dimensions underlying the control of voice quality, and implement analytic mapping functions with low-level synthesis parameters. Dimensionnal features of voice were first collected from various research fields (signal processing, acoustics, phonetics, singing), completed, and described in a formalized set [33] [37].

- *Melody* (F_0): short-term and long-term elements involved in the organization of temporal structure of fundamental frequency;
- *Vocal Effort* (V): representation of the amount of "energy" involved in the creation of the vocal sound. It makes the difference between a spoken and a screamed voice [38] [39] [40] [41];
- *Tenseness* (T): representation of the constriction of the voice source. It makes the difference between a lax and a tensed voice [26];
- *Breathiness* (B): representation of the amount of air turbulences passing through the vocal tract, compared to the amount of voiced signal [26];
- *Hoarseness* (H): representation of the stability of sound production parameters (especially for fundamental frequency and amplitude of the voice);
- *Mecanisms* (M_i): voice quality modifications due to phonation type involved in the sound production [42].

E. Description of Mapping Functions

Once dimensions are defined, two main tasks can be investigated. First, the implementation of mapping functions between these dimensions and low-level parameters. Then, identification and implementation of inter-dimensionnal phenomena. In this area, many different theories have been proposed relating several intra- or inter-dimensionnal aspects of voice production [28] [41] [43] [44] [45] [46]. We decided to focus on some of them – like direct implementation of tenseness and vocal effort, realization of a phonetogram, etc. – and design our synthesis platform in order to be able to extend it easily (e.g. correct existing relations, add new mapping functions, etc.). All current parameters are defined for a male voice.

Relations between Dimensions and Synthesis Parameters

During this workshop, we focused on several aspects of the dimensionnal process. First, we consider relations between a limited number of dimensions (F_0 , V , T and M_i) and synthesis parameters (O_q , α_m and T_l). Then, we decided to achieve our data fusion scheme by considering two different "orthogonal" processes in the dimensionnal control. On the one hand, vocal effort (V) (also related to F_0 variations, cf. next paragraph: *Inter-Dimensionnal Relations*) and mechanisms (M_i) are controlling "offset" values of parameters (O_{q_0} , α_{m_0} , T_{l_0}). On the other hand, tenseness (T) controls "delta" values of O_q and α_m around their offsets (ΔO_q , $\Delta \alpha_m$). Considering this approach, effective values of synthesis parameters can be described as:

$$O_q = O_{q_0} + \Delta O_q$$

$$\alpha_m = \alpha_{m_0} + \Delta\alpha_m$$

$$T_l = T_{l_0}$$

Following equations consider V and T parameters normalized between 0 and 1 and M_i representing the i^{th} phonation mechanism.

- $O_{q_0} = f(V|M_i)$

$$O_{q_0} = 1 - 0,5 \times V|M_1$$

$$O_{q_0} = 0,8 - 0,4 \times V|M_2$$

- $\alpha_{m_0} = f(M_i)$

$$\alpha_{m_0} = 0,6|M_1$$

$$\alpha_{m_0} = 0,8|M_2$$

- $T_{l_0} = f(V)$

$$T_{l_0}(dB) = 55 - 49 \times V$$

- $\Delta O_q = f(T)$

$$\Delta O_q = (1 - 2T) \times O_q + 0,8T - 0,4|T \leq 0,5$$

$$\Delta O_q = (2T - 1) \times O_q + 2T + 1|T > 0,5$$

- $\Delta\alpha_m = f(T)$

$$\Delta\alpha_m = (0,5T - 1) \times \alpha_m - 1,2T + 0,6|T \geq 0,5$$

$$\Delta\alpha_m = (0,25 - 0,5T) \times \alpha_m + 0,4T - 0,2|T > 0,5$$

Last adaptation on parameters concerns a perceptual distortion of O_q (square distortion) and α_m (square root distortion) between their ranges of variation (O_q : 0,4 to 1; α_m : 0,6 to 0,8).

Inter-Dimensionnal Relations: the Phonetogram

One important characteristic of human voice production is that we are not able to produce any fundamental frequency (F_0) at any vocal effort (V). A strong relationship exists between these two perceptual features. For example, one could not produce a very low pitch (around 80Hz) at a sound pressure level higher than 80dB (for a male speaker) or conversely to produce a high pitch at low intensity. Trying to do so results in a sudden stop of vocal production. This relationship is called phonetogram, and the evolution of this dependency is varying very much from one speaker to another, considering for example that the subject is a trained singer or not, male or female, has pathological voice or not, etc. As a first approach, we decided to implement an average phonetogram, relying on the work of N. Henrich [46]. Figure 5 and Figure 6 represent two average phonetograms for male and female.

Moreover, this phenomenon involves different types of laryngeal configurations. We here dealt with mainly two configurations, first and second mechanisms of the vocal folds (M_1 and M_2). This two laryngeal mechanisms are, in the

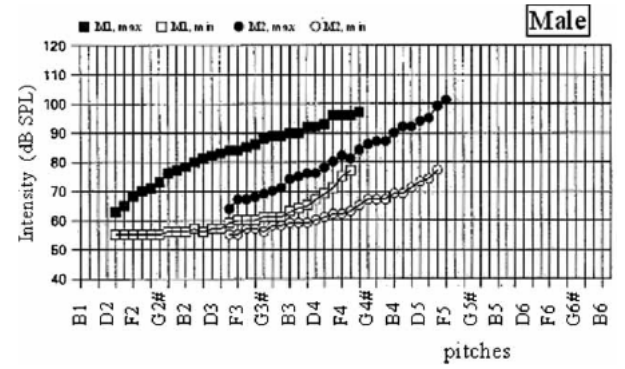


Fig. 5. Average voice range profile of male singers in mechanisms M_1 and M_2 [46].

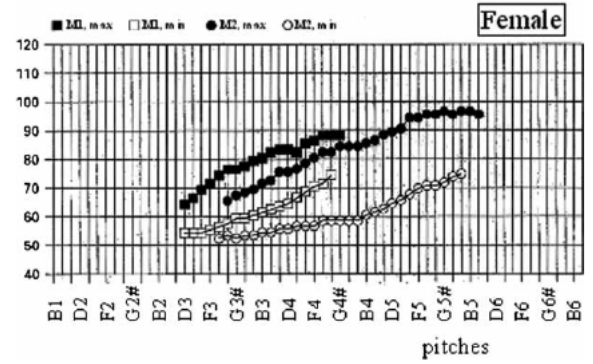


Fig. 6. Average voice range profile of female singers in mechanisms M_1 and M_2 [46].

common singing typology, referred as chest and falsetto registers. Hence, as shown on Figure 5 and Figure 6, it is also not possible to produce any frequency in both mechanisms, but the two configurations have an overlapping region in the middle of the phonetogram. This region enables the passing between the two mechanisms. Following the work presented in [47], the frequency range where this passing can occur is about one octave (or 12 semi-tones). The main characteristic of this passing is to provoke a break in the fundamental frequency (F_0). Thus, when producing an increasing glissando from M_1 to M_2 , there is an average 8 semi-tones break, whereas it is approximately 12 semi-tones when performing a decreasing glissando. Breaking intervals probabilities are depicted on Figure 7 and Figure 8. On the first one we can actually note that the frequency breaks also depends on the fundamental frequency where it occurs.

So as to say that this phenomenon introduces an hysteresis. For most of untrained speakers or singers this break is uncontrollable whereas trained singers are able to hide more or less smoothly this break, although they cannot avoid mechanism switch.

VI. THE VOCAL TRACT

In this section, we describe the implementation of a vocal tract model. This module is based on a physical "tubes-based" representation of vocal tract filter, which is simultaneously

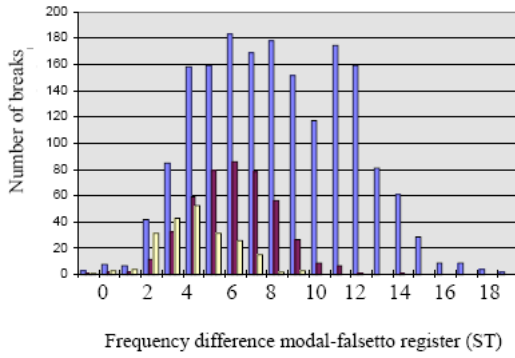


Fig. 7. Frequency drops densities in semi-tones from Chest(or Modal) to Falsetto register. In blue, when the break happens at 200Hz, in red at 300Hz, in yellow at 400Hz [47].

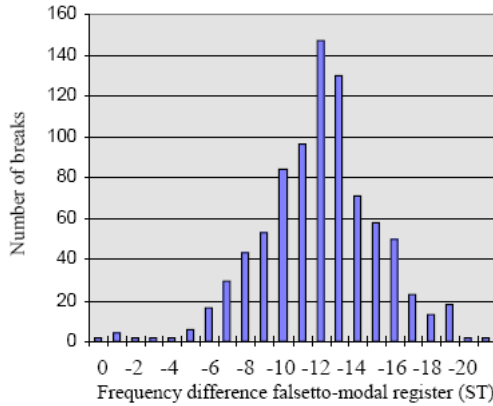


Fig. 8. Frequency drops densities in semi-tones from Falsetto to Chest(or Modal) register [47].

controllable with geometrical (areas) and spectral (formants) parameters.

A. The lattice filter

A geometrical approach of vocal tract representation

Linear Predictive Coding [48] is a method for representing the spectral envelope of a digital signal of speech in compressed form, using the information of a linear predictive model. The order of the filter is related to the complexity of the envelope, and also the number of control parameters. Thus, for representing a five-formant singing vowel, a filter containing five pairs of conjugated poles (for the resonances), and two simple poles (for the glottic wave) is needed, adding up to a total of fourteen parameters.

The LPC parameters (commonly named a_i) are non linearly interpolable. This implies that, for two configurations $[a_1 a_2 \dots a_n]$ and $[b_1 b_2 \dots b_n]$ corresponding to two vowels, a linear interpolation between both of these vectors will not correspond to a linear interpolation between the two spectra, and could even lead to unstable combinations. For these reasons, we will use another implementation of the LPC filter: the *lattice filter*. The control parameters of such a filter are called *reflection* coefficients (commonly named k_i). Such a filter is represented in Figure 9. It is composed of different sections, each characterized by a k_i parameter.

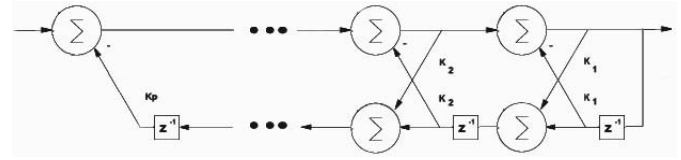


Fig. 9. Representation of k_p cells of a lattice filter.

The reflection coefficients correspond to physical characteristics of the vocal tract, which may be represented by a concatenation of cylindrical acoustic resonators, forming a lossless tube. This physical model of the lattice filter is represented in Figure 10. Each filter section represents one section of the tube; the forward wave entering the tube is partially reflected backwards, and the backward wave is partially reflected forwards. The reflection parameter k_i can then be interpreted as the ratio of acoustic reflections in the i^{th} cylindrical cavity, caused by the junction impedance with the adjacent cavity. This value varies from 1 (total reflection) to -1 (total reflection with phase inversion), and is equal to 0 when there is no reflection.

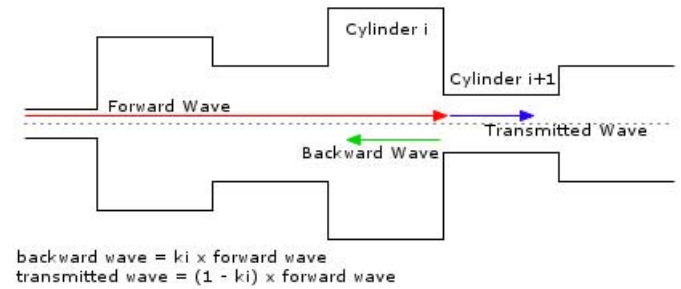


Fig. 10. Geometrical interpretation of the lattice filter: transmitted and backwards waves at each cell junction.

The filter will be stable if the k_i parameters are between -1 and 1. However, there is no direct relationship between these parameters and sound: a small modification of k_i does not imply a small modification of the spectrum. Thus, instead of using the reflection coefficients, we will be using the different cylinder areas, named A_i , which can be easily deduced from the reflection coefficients with the following expression:

$$\frac{A_i}{A_{i+1}} = \frac{1 + k_i}{1 - k_i}$$

By acting on these A_i parameters, the interpreter is directly connected to the physical synthesis instrument. The sound spectrum will then evolve with acoustical coherence, which makes it more natural to use. Moreover, the stability of the filter is guaranteed for all A_i values.

B. Coefficients Conversion Framework

In order to use the area parameters of the lattice filter (A_i), a Max/MSP object was created to convert them to k_i values which are used in the lattice filter. Several sets of A_i parameters corresponding to different vowels were calculated. After selecting one of these presets, certain sections of the

vocal tract can be modified by a percentage ΔA_i , which has the effect of opening or closing that section of the oral cavity.

A second approach to controlling the lattice filter was considered: a formant-based scheme was used to represent the spectral envelope, and the formant features, F_i , were converted to k_i parameters (after conversion to the LPC a_i coefficients), and then to A_i areas to control the lattice filter. This allowed us to easily model certain phenomena that are well known in speech processing, like overtone singing or the singer formant [49] [50], by acting on analytical parameters (the formants) rather than geometrical parameters (the areas). Similarly to the control of the areas, the formants have presets for different vowels and can be modified by a percentage ΔF_i .

The parameters conversion framework described above is represented in Figure 11.

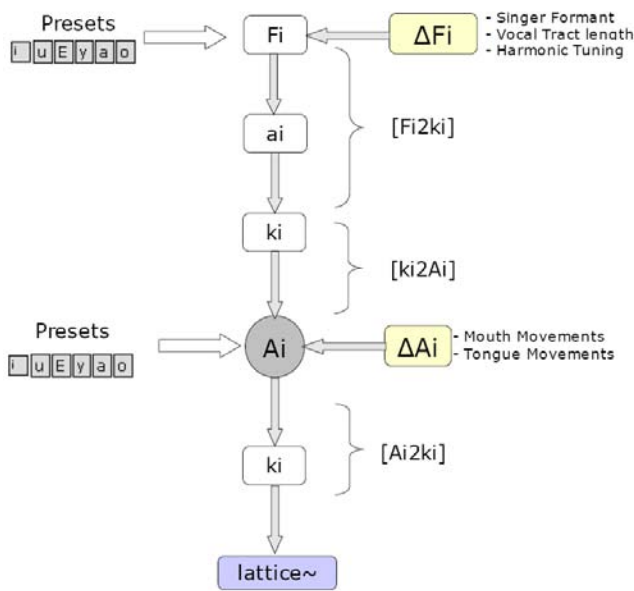


Fig. 11. Coefficients conversion and presets/modifications framework.

VII. ABOUT THE REAL-TIME CONTROL OF VOICE SYNTHESIS

In this section, we comment some experimentations we realized in order to evaluate expressive and performing abilities of systems we developed. Modules were intergrated together inside Max/MSP and various control devices (and various combinaisons) were dynamically connected with mapping matrix. This set of tests allowed us to reach efficient configurations, considering several performing styles (classical singing, overtone singing, etc) which were demonstrated at the end of the workshop.

A. Concerning Voice Source

In order to be able to compare expressive skills of this system with the one developped before [3] [33] [37], we decided to keep the same control scheme: a graphic tablet. In that way, we were able to evaluate really clearly ameliorations

achieved with this new mapping functions. Early experimentations demonstrated us that independant control of tenseness and vocal effort is really increasing performing possibilities. Anyway, current mapping equations still provide some unlikely parameters combinaisons, resulting e.g. in "ultra-tensed" perception or unwilling dynamics variations.

The implementation of the phonetogram is also a major improvement in term of naturalness. It also gives better results in terms of expressivity than without monitored control of loudness (more linear). Although we deeply investigated this phenomenon, we did not yet integrated this frequency break in the system, as we did not find a satisfying solution for controlling it. It is not straightforward to translate this frequency break in the control domain, as our hand gestures are mainly continuous and as basic switch from one configuration to another is not really satisfying from a musical point of view, as it results in a break in frequency range and thus "wrong" notes.

B. Concerning Vocal Tract

The vocal tract was controlled using a data glove (P5glove [51]) as shown in Figure 12. The glove was mapped to the area parameters of the lattice filter in four different ways:

- The folding of the fingers control the opening angle of the mouth (represented in Figure 13) (see Figure 14)
- The hand movement along the z-axis controls the position of the "tongue" in the vocal tract (towards the back or the front of the mouth)
- The hand movement along the y-axis controls the vertical position of the tongue (near or far from the palate) (see Figure 14)
- The hand movement along the x-axis changes the vowels (configurable from one preset to another, for example from an /a/ to an /o/)



Fig. 12. Vocal tract control with a data glove: 5 finger flexion sensors and 3 dimensions (x,y,z) tracking.

This configuration allowed us to achieved typical vocal tract modification techniques – like overtone singing – quite easily. Indeed, as the spectral representation (F_i) is really efficient to configure some presets (e.g. offset vowel) or let running automatic tasks (e.g. harmonic/formant tuning), the constant access to geometrical "delta" features (ΔS_i) allows user to

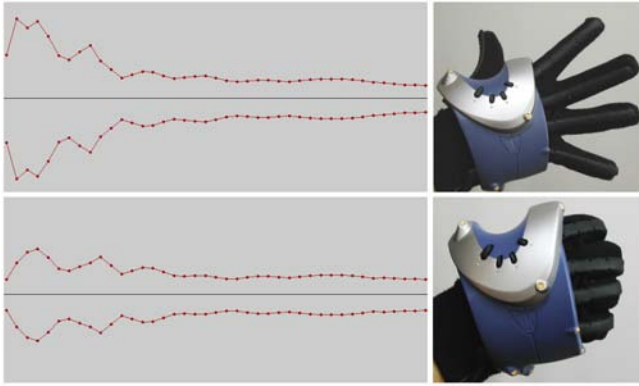


Fig. 13. Mouth opening control: finger flexion sensors mapped to variation of 9 first A_i .

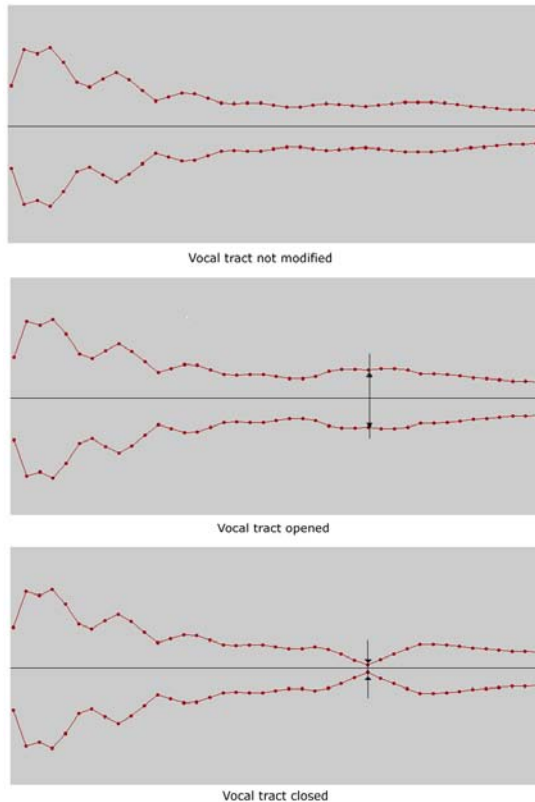


Fig. 14. Vertical tongue position control.

achieve refined tweaking techniques (e.g. lowering the vocal tract, changing tongue position, etc.) and that way increasing expressivity.

C. Transversal Remarks

In overall, at this stage of development, the synthesizer allows to control 17 parameters which are namely : pitch, vocal effort, tenseness, mechanisms, the first two formants, the singer's formant, vocal tract length, gain, transition between vowels, width of the vocal tract, position of the tongue and mouth opening (5 parameters). Considering all these parameters, only the actions on mechanisms is not a continuous

parameter, so as to say that 16 parameters have to be monitored thanks to continuous parameters. From the controllers side, we have all in all 17 continuous parameters (out of 33), meaning that we are actually theoretically able to control all needed parameters. However, the problem is that from user's side, it is impossible to manipulate three interfaces at the same time. There are actually two solutions : one is to have multiple users (2 or 3) being in control of the interfaces, the other one is to use one-to-many mappings, allowing the performer to control several parameters with the same controller.

VIII. CONCLUSIONS

In this workshop, our main aim was to build a performant singing musical instrument allowing a wide range of expressive possibilities. Our actual work results in the implementation of new models for voice source and vocal tract, working in real-time, which are strategic tools in order to be able to work further. Improvement of expressivity in this new system really encourage us to go forward with this approach. Moreover, our modular architecture drives us to go to a widely extensible synthesis platform which will be really usefull in order to continue to integrate other results (existing and coming) from voice production sciences.

ACKNOWLEDGMENT

The authors would like to thank SIMILAR Network of Excellence (and through it the European Union) which provides ressources allowing researchers from all Europe to meet, share and work together, and then achieving really exciting results. We also would like to thank croatian organization team (responsible: Prof. Igor Pandzic) which maintain local structures in order to manage the work and the life of more than 50 people. Finally, we would like to thank our respective laboratories (in our case: TCTS Lab, Mons, Belgium and LIMSI-CNRS, Paris, France) which adapt their research agendas in order to allow us to participate to those annual summer events.

REFERENCES

- [1] "<http://www.loquendo.com/>"
- [2] "<http://www.interface.net/interface05/>"
- [3] C. d'Alessandro, N. D'Alessandro, S. L. Beux, J. Simko, F. Cetin, and H. Pirker, "The speech conductor: Gestural control of speech synthesis," in *Proceedings of eINTERFACE'05 Summer Workshop on Multimodal Interfaces*, 2005.
- [4] M. Kob, "Singing voice modelling as we know it today," *Acta Acustica United with Acustica*, vol. 90, pp. 649–661, 2004.
- [5] "<http://www.virsyn.de/>"
- [6] "<http://www.vocaloid.com/>"
- [7] X. Rodet and G. Bennet, "Synthesis of the singing voice," *Current Directories in Computer Music Research*, 1989.
- [8] X. Rodet, "Synthesis and processing of the singing voice," in *Proceeding of the First IEEE Benelux Workshop on Model-Based Processing and Coding of Audio (MPCA-2002)*, Leuven, Belgium, 2002.
- [9] P. Cook, "Identification of control parameters in an articulatory vocal tract model, with applications to the synthesis of singing," Ph.D. thesis, Stanford University, 1990.
- [10] J. Moorer, "The use of the phase vocoder in computer music application," *Journal of the Audio Engineering Society*, vol. 26, no. 1-2, pp. 42–45, 1978.
- [11] J. Laroche, Y. Stylianou, and E. Moulines, "Hns: Speech modifications based on a harmonic plus noise model," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 2, 1993, pp. 550–553.
- [12] M. Macon, L. Jensen-Link, J. Oliviero, M. Clements, and E. George, "A singing voice synthesis system based on sinusoidal modeling," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, 1997, pp. 435–438.
- [13] K. Lomax, "The analysis and the synthesis of the singing voice," Ph.D. thesis, Oxford University, 1997.
- [14] Y. Meron, "High quality singing synthesis using the selection-based synthesis scheme," Ph.D. thesis, University of Michigan, 2001.
- [15] P. Cano, A. Loscos, J. Bonada, M. de Boer, and X. Serra, "Voice morphing system for impersonating in karaoke applications," in *Proceedings of the International Computer Music Conference*, 2000.
- [16] B. Doval, C. d'Alessandro, and N. Henrich, "The spectrum of glottal flow models," *Acta Acustica*, vol. In press, 2006.
- [17] L. Kessous, "A two-handed controller with angular fundamental frequency control and sound color navigation," in *Proceedings of the 2002 Conference on New Interfaces for Musical Expression (NIME-02)*, 2002.
- [18] G. Fant, J. Liljencrants, and Q. Lin, "A four-parameter model of glottal flow," *STL-QPSR*, vol. 4, pp. 1–13, 1985.
- [19] B. Doval and C. d'Alessandro, "The voice source as a causal/anticausal linear filter," in *proc. Voqual'03, Voice Quality: Functions, analysis and synthesis, ISCA workshop*, Geneva, Switzerland, Aug. 2003.
- [20] B. Larson, "Music and singing synthesis equipment (musse)," *Speech Transmission Laboratory Quarterly Progress and Statut Report (STL-QPSR)*, pp. (1/1977):38–40, 1977.
- [21] P. Cook, "Spasm: a real-time vocal tract physical model editor/controller and singer: the companion software system," in *Colloque sur les Modèles Physiques dans l'Analyse, la Production et la Création Sonore*, 1990.
- [22] J. O. Smith, "Waveguide filter tutorial," in *Proceedings of the International Computer Music Conference*, 1987, pp. 9–16.
- [23] V. Välimäki and M. Karjalainen, "Improving the kelly-lochbaum vocal tract model using conical tubes sections and fractionnal delay filtering techniques," in *Proceedings of the International Conference on Spoken Language Processing*, 1994.
- [24] X. Rodet, "Time-domain formant wave function synthesis," vol. 8, no. 3, pp. 9–14, 1984.
- [25] X. Rodet and J. Barriere, "The chant project: From the synthesis of the singing voice to synthesis in general," *Computer Music Journal*, vol. 8, no. 3, pp. 15–31, 1984.
- [26] N. Henrich, "Etude de la source glottique en voix parlée et chantée," Ph.D. thesis, Université Paris 6, France, 2001.
- [27] G. Fant, *Acoustic theory of speech production*. Mouton, La Hague, 1960.
- [28] D. Klatt and L. Klatt, "Analysis, synthesis, and perception of voice quality variations among female and male talkers," *J. Acous. Soc. Am.*, vol. 87, no. 2, pp. 820–857, 1990.
- [29] R. Veldhuis, "A computationally efficient alternative for the liljencrants-fant model and its perceptual evaluation," *J. Acous. Soc. Am.*, vol. 103, pp. 566–571, 1998.
- [30] A. Rosenberg, "Effect of glottal pulse shape on the quality of natural vowels," *J. Acous. Soc. Am.*, vol. 49, pp. 583–590, 1971.
- [31] G. Fant, "The lf-model revisited. transformations and frequency domain analysis," *STL-QPSR*, 1995.
- [32] B. Bozkurt, "Zeros of the z-transform (zzt) representation and chirp group delay processing for the analysis of source and filter characteristics of speech signals," Ph.D. dissertation, Faculté Polytechnique de Mons, 2004.
- [33] N. D'Alessandro, C. d'Alessandro, S. Le Beux, and B. Doval, "Realtime calm synthesizer, new approaches in hands-controlled voice synthesis," in *NIME'06, 6th international conference on New Interfaces for Musical Expression*, IRCAM, Paris, France, 2006, pp. 266–271.
- [34] D. Zicarelli, G. Taylor, J. Clayton, jhno, and R. Dudas, *Max 4.3 Reference Manual*. Cycling'74 / Ircam, 1993-2004.
- [35] —, *MSP 4.3 Reference Manual*. Cycling'74 / Ircam, 1997-2004.
- [36] M. Puckette, *Pd Documentation*. <http://puredata.info>, 2006.
- [37] C. d'Alessandro, N. D'Alessandro, S. L. Beux, and B. Doval, "Comparing time-domain and spectral-domain voice source models for gesture controlled vocal instruments," in *Proc. of the 5th International Conference on Voice Physiology and Biomechanics*, 2006.
- [38] R. Schulman, "Articulatory dynamics of loud and normal speech," *J. Acous. Soc. Am.*, vol. 85, no. 1, pp. 295–312, 1989.
- [39] H. M. Hanson, "Glottal characteristics of female speakers," Ph.D. thesis, Harvard University, 1995.
- [40] —, "Glottal characteristics of female speakers : Acoustic correlates," *J. Acous. Soc. Am.*, vol. 101, pp. 466–481, 1997.
- [41] H. M. Hanson and E. S. Chuang, "Glottal characteristics of male speakers : Acoustic correlates and comparison with female data," *J. Acous. Soc. Am.*, vol. 106, no. 2, pp. 1064–1077, 1999.
- [42] M. Castellengo, B. Roubeau, and C. Valette, "Study of the acoustical phenomena characteristic of the transition between chest voice and falsetto," in *Proc. SMAC 83, vol. 1*, Stockholm, Sweden, July 1983, pp. 113–23.
- [43] P. Alku and E. Vilkman, "A comparison of glottal voice source quantification parameters in breathy, normal and pressed phonation of female and male speakers," *Folia Phoniatr.*, vol. 48, pp. 240–54, 1996.
- [44] H. Traunmüller and A. Eriksson, "Acoustic effects of variation in vocal effort by men, women, and children," *J. Acous. Soc. Am.*, vol. 107, no. 6, pp. 3438–51, 2000.
- [45] N. Henrich, C. d'Alessandro, B. Doval, and M. Castellengo, "On the use of the derivative of electroglottographic signals for characterization of non-pathological phonation," *J. Acous. Soc. Am.*, vol. 115, no. 3, pp. 1321–1332, Mar. 2004.
- [46] N. Henrich, C. d'Alessandro, M. Castellengo, and B. Doval, "Glottal open quotient in singing: Measurements and correlation with laryngeal mechanisms, vocal intensity, and fundamental frequency," *J. Acous. Soc. Am.*, vol. 117, no. 3, pp. 1417–1430, Mar. 2005.
- [47] G. Bloothoof, M. van Wijck, and P. Pabon, "Relations between vocal registers in voice breaks," in *Proceedings of Eurospeech*, 2001.
- [48] J. D. Markel and A. H. Gray, *Linear prediction of speech*. Springer-Verlag, Berlin, 1976.
- [49] B. STORY, "Physical modeling of voice and voice quality," in *proc. Voqual'03, Voice Quality: Functions, analysis and synthesis, ISCA workshop*, Geneva, Switzerland, Aug. 2003.
- [50] G. Carlsson and J. Sundberg, "Formant frequency tuning in singing," *J. Voice*, vol. 6, no. 3, pp. 256–60, 1992.
- [51] "<http://www.vrealities.com/p5.html>"

Nicolas D'Alessandro holds an Electrical Engineering degree from the Faculté Polytechnique de Mons (FPMs) since 2004. I did the master's thesis in the Faculty of Music of the University de Montréal (UdeM) (supervisor: Prof. Caroline Traube). That work gathered the development of applications based on perceptual analogies between guitar sounds and voice sounds, and a study of mapping strategies between (hand-based) gestures and speech production models (source/filters and concatenative approaches). He started a PhD thesis in September 2004 in the TCTS Lab of the FPMs (supervisor: Prof. Thierry Dutoit) related to the real-time control of unit-based synthesizers.

Boris Doval holds an Engineering degree from the "Ecole Centrale de Paris" since 1987. He did his master thesis at IRCAM and his PhD thesis at "Université Paris VI" on fundamental frequency estimation of sound signals. He then joined LIMSI-CNRS as an associate professor where he concentrated until now on voice source analysis, synthesis and modelisation. In particular, he coorganized the ISCA workshop VOQUAL'03 on voice quality in 2003.

Sylvain Le Beux graduated from Master in Electronics, Telecommunications and Informatics from CPE Lyon engineer school in 2004. During his training he did an internship at Infineon Technology A.G. in Munich for one year, and achieved his master's thesis at IRCAM which topic was about speech recognition. So he actually helped IRCAM wreck on a nice beach using calm insense. He then graduated from a Master thesis on embedded systems and data processing from Orsay University in 2005, and then integrated LIMSI Laboratory where he is currently achieving his PhD focused on the gestural control of speech synthesis and relationship between intention and expressivity.

Pascale Woodruff holds an Electrical Engineering degree from FPMs since June 2004. She is currently working on a project which aims to improve the workflow in industrial maintenance by equipping technicians with a multimodal wearable system allowing them to access maintenance information using speech and/or other modalities.

Yohann Fabre is ending a master thesis in audiovisual technologies at the ISIS (Ingénierie des Systèmes, Image et Son) of Valenciennes. He is currently working as a trainee on voice synthesis at the TCTS Lab of FPMs.

Multimodal Signal Processing and Interaction for a Driving Simulator: Component-based Architecture

A. Benoit, L. Bonnaud, A. Caplier

Institut National Polytechnique de Grenoble, France, LIS Lab.

Y. Damousis, D. Tzovaras

Centre for Research and Technology Hellas – Thessaloniki, Greece, IT Institute

F. Jourde, L. Nigay, M. Serrano

Université Joseph Fourier, Grenoble 1, France, CLIPS Lab.

L. Lawson

Université Catholique de Louvain, Belgium, TELE Lab.

Abstract— After a first workshop at eINTERFACE 2005 focusing on developing video-based modalities for an augmented driving simulator, this project aims at designing and developing a multimodal driving simulator that is based on both multimodal driver's focus of attention detection as well as driver's fatigue state detection and prediction. Capturing and interpreting the driver's focus of attention and fatigue state will be based on video data (e.g., facial expression, head movement, eye tracking). While the input multimodal interface relies on passive modalities only (also called attentive user interface), the output multimodal user interface includes several active output modalities for presenting alert messages including graphics and text on a mini-screen and in the windshield, sounds, speech and vibration (vibration wheel). Active input modalities are added in the meta-User Interface to let the user dynamically select the output modalities. The driving simulator is used as a case study for studying software architecture for multimodal signal processing and multimodal interaction using two software component-based platforms, OpenInterface and ICARE.

Index Terms— Attention level, Component, Driving simulator, Facial movement analysis, ICARE, Interaction modality, OpenInterface, Software architecture, Multimodal interaction.

I. INTRODUCTION

THE project aims to study component-based architecture using two platforms, namely OpenInterface [1] and ICARE [2] [3], for combining multimodal signal processing

analysis and multimodal interaction. OpenInterface is a component-based platform developed in C++ that handles distributed heterogeneous components. OpenInterface supports the efficient and quick definition of a new OpenInterface component from an XML description of a program. By so doing, any program can be included as an OpenInterface component and can then communicate with any other existing OpenInterface component. As opposed to OpenInterface, ICARE is a conceptual component model for multimodal input/output interaction [2]. One implementation of the ICARE model is defined using JavaBeans components [3].

In this project, we study the development of a multimodal interactive system using the OpenInterface platform while the component architecture is along the ICARE conceptual model. The selected case study for this project is a driving simulator [4].

The structure of the paper is as follows: first we present the selected case study by explaining the rationale for selecting this interactive system from a multimodal interaction point of view and by giving an overview of the interactive system. We then recall the key points of the two platforms, OpenInterface and ICARE before presenting the software architecture along the ICARE conceptual model. We then detail the software architecture that has been implemented followed by a discussion on the tradeoffs and differences with the initial ICARE architecture.

II. CASE STUDY: DRIVING SIMULATOR

A. Rational for selecting a driving simulator

The case study is a driving simulator. Indeed, facing the

This report, as well as the source code for the software developed during the project, is available online from the eINTERFACE'05 web site: www.enterface.net.

This research was partly funded by SIMILAR, the European Network of Excellence on Multimodal Interfaces, during the eINTERFACE'06 Workshop in Dubrovnik, Croatia.

sophisticated sensing technology available in modern cars, multimodal interaction in cars constitutes a very challenging domain. The key issue in terms of interaction design is that the main task of the user is the driving one, a critical task which requires a driver to keep her/his eyes on the road. A driving task relies on local guidance that includes sub-tasks involving control of the vehicle and knowledge of the environmental situation. In this context of a driving task, our application domain, our goals are:

- to capture a driver's focus of attention,
- to capture a driver's state of fatigue,
- to predict a driver's state of fatigue,
- to design and develop an output multimodal user interface for presenting alert messages to the driver.

Several projects focus on User Interfaces (UI) in cars and involve various interaction technologies such as trackpad fixed on the steering wheel [5], dedicated buttons, mini-screens as well as head-up display (HUD) technology. For example HUDs are used for displaying icons and texts, usually found on the dashboard of a car, in the windshield as shown in Figure 1.



Fig. 1. In-car HUD (from [5]).

We distinguish two main classes of UI studies in cars: design of interactive dashboards that nowadays include a screen (e.g., graphical user interface for controlling the radio and so on) and Augmented Reality (AR) visualizations. Several on-going projects focus on Augmented Reality (AR) visualizations for the driver using head-up display (HUD) technology. For example for displaying navigation information or for guiding the driver's attention to dangerous situations, transparent graphics (e.g., transparent path of the route) are directly projected onto the windshield [6] as shown in Figure 2, making it possible for the driver to never take her/his eyes off the road.



Fig. 2. In-car Augmented Reality: Guiding driver's attention to dangerous situation. The arrow indicates the position of imminent danger (from [6]).

Complementary to these projects, our task focuses on supporting the driving activity by monitoring and predicting the state of the driver (attention and fatigue). Instead of focusing on external dangers (e.g. a potential collision with a

car coming from behind as in Figure 2), the project aims at detecting dangerous situations due to the driver's fatigue state and focus of attention. From the Human-Computer Interaction point of view, the project focuses on multimodal input and output interaction that combines passive input modalities (implicit actions of the driver) for detecting dangerous situations as well as active modalities (explicit actions of the driver) for perceiving alarms (output active modalities) and for changing the output modalities (input active modalities).

B. Overview of the driving simulator

Starting from the programs developed during a first workshop at eNTERFACE 2005 [4], the overall hardware setting of the driving simulator includes:

- 3 PCs: one under Windows for the driving simulator, one under Linux for capturing and predicting the driver's states (focus of attention and state of fatigue), and one on Windows for the output user interface developed using the ICARE platform (JavaBeans component).
- 1 LOGITECH webcam sphere
- 1 LOGITECH force feedback wheel
- 1 video-projector
- 2 loudspeakers

Figure 3 shows the system in action. For software, the driving simulator we used is the GPL program TORCS [7] and the multimodal interaction is developed using the two platforms OpenInterface and ICARE.



Fig. 3. Multimodal driving simulator: demonstrator in use.

III. COMPONENT PLATFORMS

A. OpenInterface platform

OpenInterface is a component-based platform developed in C++ that handles distributed heterogeneous components. OpenInterface supports the efficient and quick definition of a new OpenInterface component from an XML description of a program. Although the platform is generic, in the context of the SIMILAR project, the OpenInterface platform is dedicated to multimodal applications. We define a multimodal

application as an application that includes multimodal data processing and/or offers multimodal input/output interaction to its users.

Figure 4 gives an overview of the platform. Each component is registered in OpenInterface Platform using the Component Interface Description Language (CIDL) and described in XML. The registered components properties are retrieved by the Graphic Editor (Java). Using the editor the user can edit the component properties and compose the execution pipeline (by connecting the components) of the multimodal application. This execution pipeline is sent to the OpenInterface Kernel (C/C++) to run the application.

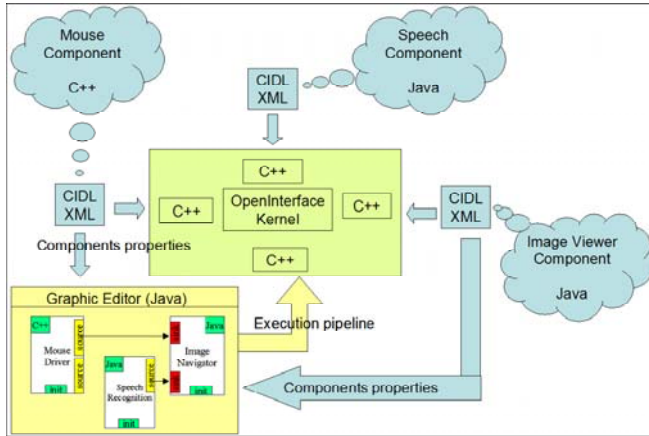


Fig. 4. Overview of the OpenInterface platform.

OpenInterface is designed to serve three levels of users: programmers, application designers (AD) and end-users. Programmers are responsible for the development and integration of new components into the platform. The application designers focus on end-user's needs and are aware of the resources provided by the platform. The AD will use the graphical editor to assemble components in order to develop a multimodal application. End-users interact with the final application whose components are executed within the platform.

B. ICARE platform

ICARE (Interaction CARE -Complementarity Assignment, Redundancy and Equivalence-) is a component-based approach which allows the easy and rapid development of multimodal interfaces [2] [3]. The ICARE platform enables the designer to graphically manipulate and assemble ICARE software components in order to specify the multimodal interaction dedicated to a given task of the interactive system under development. From this specification, the code is automatically generated. The currently developed ICARE platform that implements a conceptual component model that describes the manipulated software components, is based on the JavaBeans technology [8]. The ICARE conceptual model includes:

1. Elementary components: Such components are building blocks useful for defining a modality. Two types of ICARE elementary components are defined: Device components and

Interaction Language components. We reuse our definition of a modality [9] as the coupling of a physical device d with an interaction language L : $\langle d, L \rangle$. In [10], we demonstrate the adequacy of the notions of physical device and interaction language for classifying and deriving usability properties for multimodal interaction and the relevance of these notions for software design.

2. Composition components: Such components describe combined usages of modalities and therefore enable us to define new composed modalities. The ICARE composition components are defined based on the four CARE properties [10]: the Complementarity, Assignment, Redundancy, and Equivalence that may occur between the modalities available in a multimodal user interface. We therefore define three Composition components in our ICARE conceptual model: the Complementarity one, the Redundancy one, and the Redundancy/Equivalence one. Assignment and Equivalence are not modeled as components in our ICARE model. Assignment and Equivalence are not modeled as components in our ICARE model. Indeed, an assignment is represented by a single link between two components. An ICARE component A linked to a single component B implies that A is assigned to B. As for Assignment, Equivalence is not modeled as a component. When several components (2 to n components) are linked to the same component, they are equivalent. As opposed to ICARE elementary components, Composition components are generic in the sense that they are not dependent on a particular modality.

The two ICARE composition components, Complementarity and Redundancy/Equivalence have been developed in C++ as connectors within the OpenInterface platform.

In the following section, examples of ICARE component assemblies are provided in the context of the multimodal driving simulator.

IV. SOFTWARE ARCHITECTURE OF THE MULTIMODAL DRIVING SIMULATOR

In this section, we first present the overall architecture along the ICARE conceptual model that we defined at the beginning of the project followed by the implemented architecture developed during the workshop. We finally conclude by a discussion of the tradeoffs and differences between the initial conceptual architecture and the implemented one.

A. ICARE conceptual architecture

In Figure 5, we present the overall software architecture of the entire multimodal driving simulator in order to highlight the scope of the code organized along the ICARE conceptual model. As pointed out in Figure 5, within the architecture, we identify two types of link between the ICARE components and the rest of the interactive system:

- For inputs, the connection between the ICARE Input components and the rest of the interactive system is at the level of the elementary tasks. From explicit or implicit actions performed by the driver (i.e., the user) along various modalities, the ICARE components are responsible for defining elementary tasks that are independent of the used modalities. Such elementary tasks are then transmitted to the Dialogue Controller. One example of a driving task is the “accelerate” task.
- For outputs, the Dialogue Controller is sending elementary presentation tasks to the ICARE output components that are responsible for making the information perceivable to the driver along various output modalities. One example of an elementary task is the “present alarm” task.

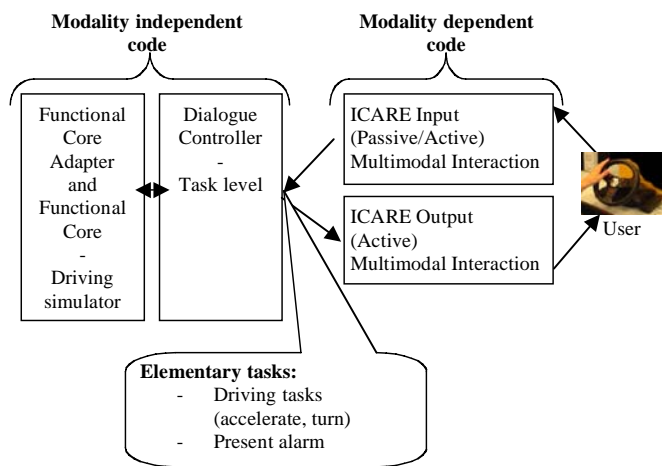


Fig. 5. Overall architecture of the multimodal driving simulator.

Because we reuse the GPL driving simulator TORCS [7] that we extend to be multimodal, some parts of the architecture of Figure 5 are already developed. Figure 6 shows the code that we need to develop along with the existing TORCS code. All the modalities for driving (input modalities based on the steering wheel and the pedal) and for displaying

the graphical scene are reused and not developed with ICARE components.

To better understand the extensions to be developed, Figure 7 presents the task tree managed by the Dialogue Controller. Within the task tree, the task “Choose output modalities” does not belong to the main Dialogue Controller of the driving simulator but rather belongs to a distinct Dialogue Controller dedicated to the meta User Interface (meta UI) as shown in Figure 8. Indeed the meta UI enables the user to select the modalities amongst a set of equivalent modalities. Such a task, also called an articulatory task, does not correspond to a task of the driving simulator itself. The meta UI includes a second Dialogue Controller (Dialogue Controller (2) in Figure 8) as well as ICARE input components for specifying the selection. The selection is then sent by the second Dialogue Controller to the ICARE output components [11].

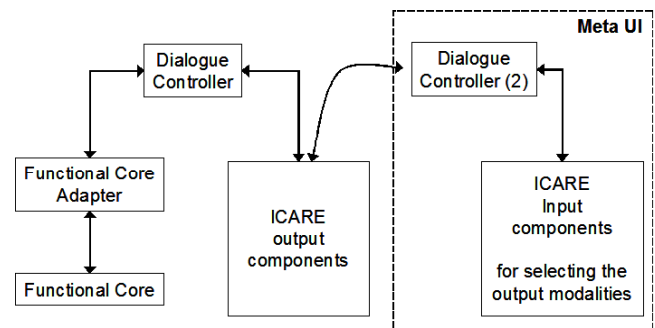


Fig. 8. Meta User Interface: ICARE components within an overall software architecture of an interactive system and the meta UI that enables the selection of equivalent modalities by the user (from [11]).

To obtain the final ICARE architecture, for each elementary task of Figure 7, an ICARE diagram is defined. Figure 9 presents the four ICARE diagrams designed for the four elementary tasks to be developed.

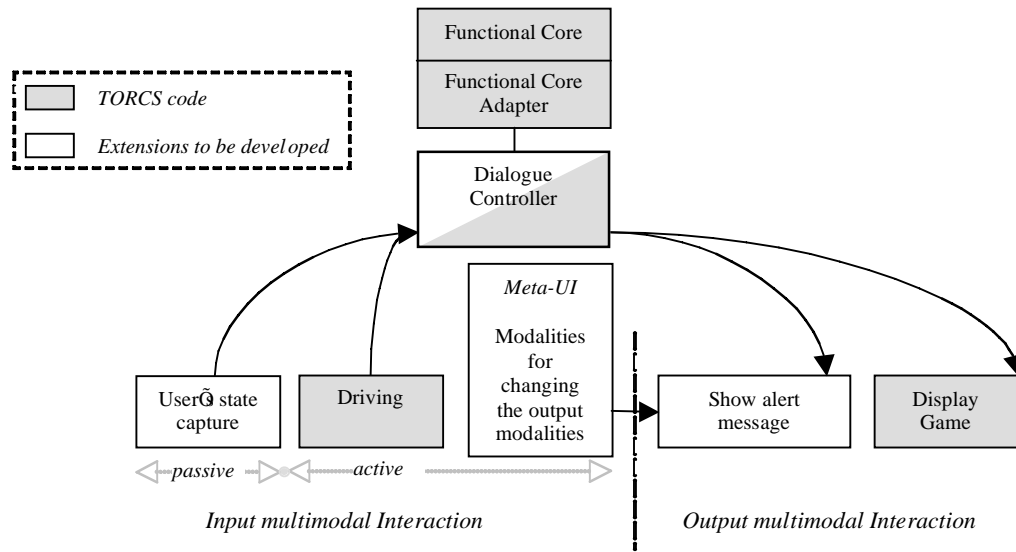


Fig. 6. TORCS code and extensions to be developed within our architecture.

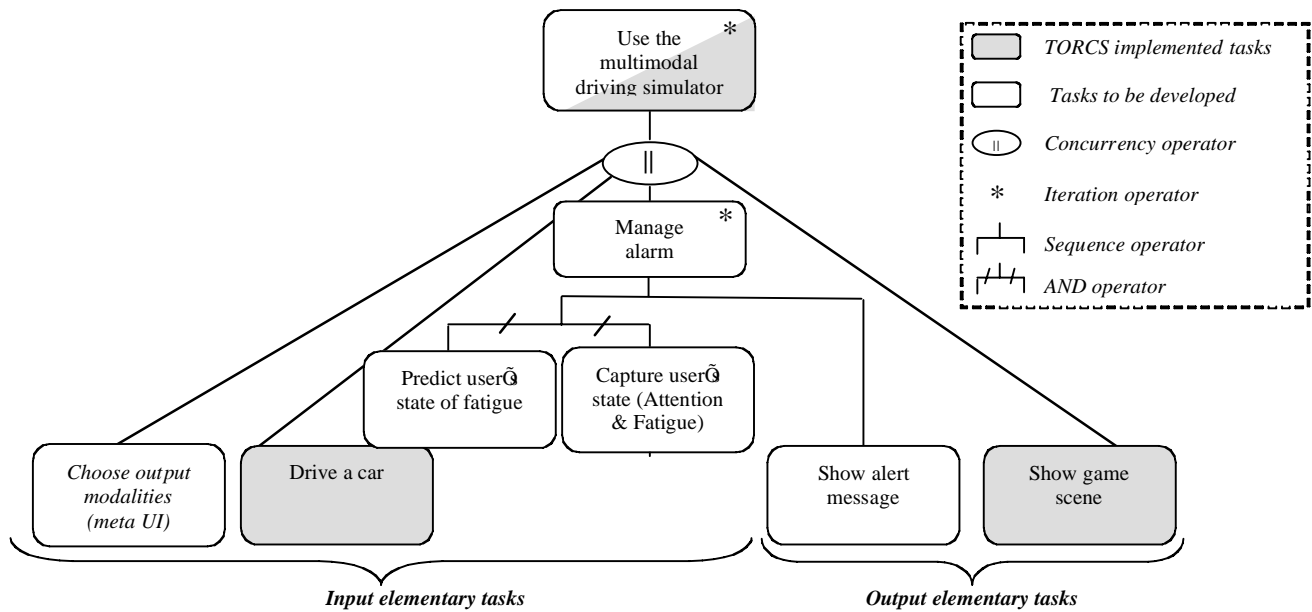


Fig. 7. Hierarchical Task Analysis (HTA): Task tree corresponding to the Dialogue Controller.

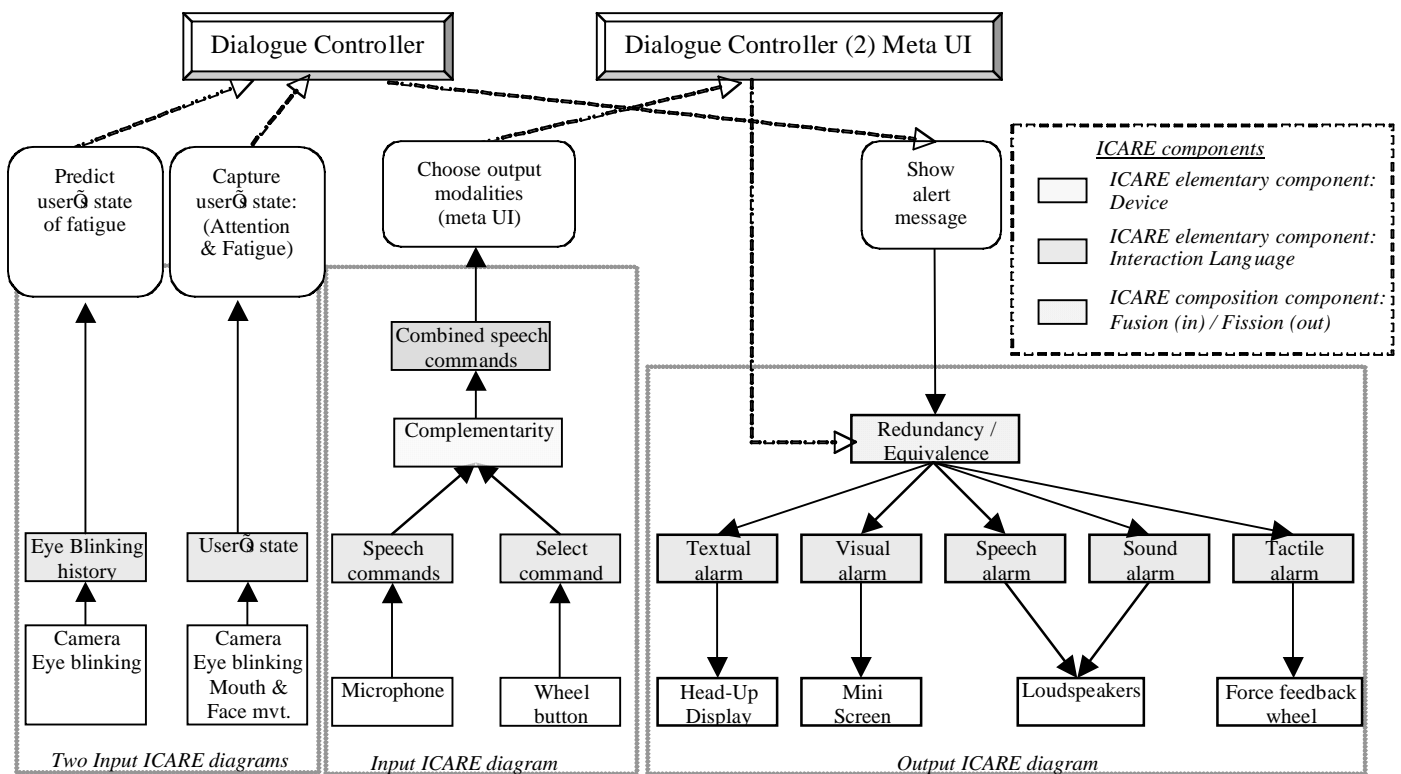


Fig. 9. ICARE diagrams for the elementary task of Figure 7.

The ICARE diagrams for the multimodal driving simulator include pure modalities and two composition components.

- For input, pure modalities made of a device and an interaction language components are used for the two tasks; (i) capture the user's state of fatigue and attention and (ii) predict the user's state of fatigue. These two modalities are passive input modalities. The

modality for capturing the user's state is based on eye blinking and mouth movement (yawning) for detecting the state of fatigue and on face movement for capturing the focus of attention. Instead of one pure modality, we can also define three modalities, one for the state of fatigue based on mouth movement, one for the state of fatigue based on eye blinking and one for the focus of

attention. The three modalities will then be combined by two composition components as shown in Figure 10.

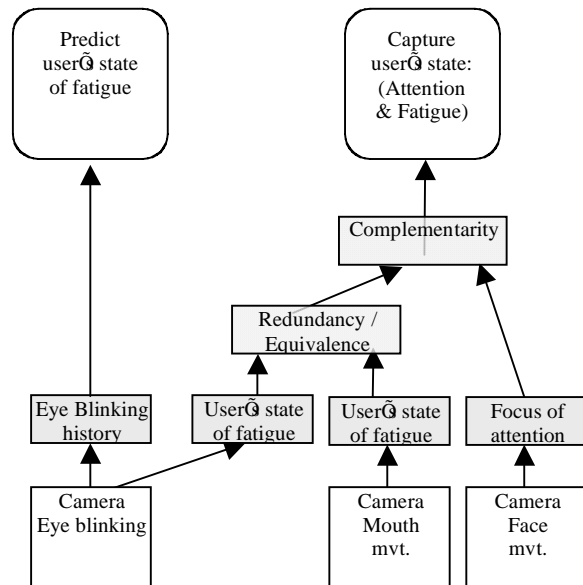


Fig. 10. Combined modalities for capturing and detecting user's state.

For selecting the output modalities the user issues speech commands such as “windshield screen voice beep tactile” for selecting all the output modalities. For using this combined modality, the user first selects a wheel button then issues the voice command, and finally selects again the button to indicate the end of the speech commands. As shown in Figure 9 two pure modalities, speech and button, are combined by a Complementarity composition component. Finally an Interaction Language component is responsible for combining all the recognized words between the two button press events. The output of this component is a list of selected modalities that is sent to the second Dialogue Controller of the meta User Interface.

- For outputs, five pure modalities made of a device and an interaction language component are defined for presenting an alarm. Such modalities are combined by a Redundancy/Equivalence composition component. This composition component implies that the five modalities can be used all together in a redundant way or that only a sub-set of the modalities (1 to 5 modalities) can be used in a redundant way.

Having presented the ICARE overall software architecture of the multimodal driving simulator, we now present the implemented architecture and in particular which components of the architecture have been implemented in OpenInterface.

A. Implemented architecture

We first describe the implemented OpenInterface components and then explain in the following section the differences between the ICARE conceptual architecture and the implemented architecture. We have developed six

OpenInterface components:

- One OpenInterface component is dedicated to the video stream. Such a component is not explicit in the ICARE architecture since it represents a supplementary layer of the physical device driver.
- One OpenInterface component is implementing the software interface to be able to send messages to the TORCS code.
- One OpenInterface component implements all the ICARE diagrams for the task “Show alert message” of Figure 9 as well as the meta User Interface. This component has been implemented with ICARE JavaBeans components. The final implemented ICARE diagram is presented in Figure 11. First, due to time constraints, the Complementarity component of Figure 9 has not been used for developing the combined active modalities based on speech and a steering wheel button. Second, we decided to add a new modality for choosing modalities using dedicated buttons on the steering wheel. The two modalities are then equivalent for the task “Choose output modalities”.

One OpenInterface component

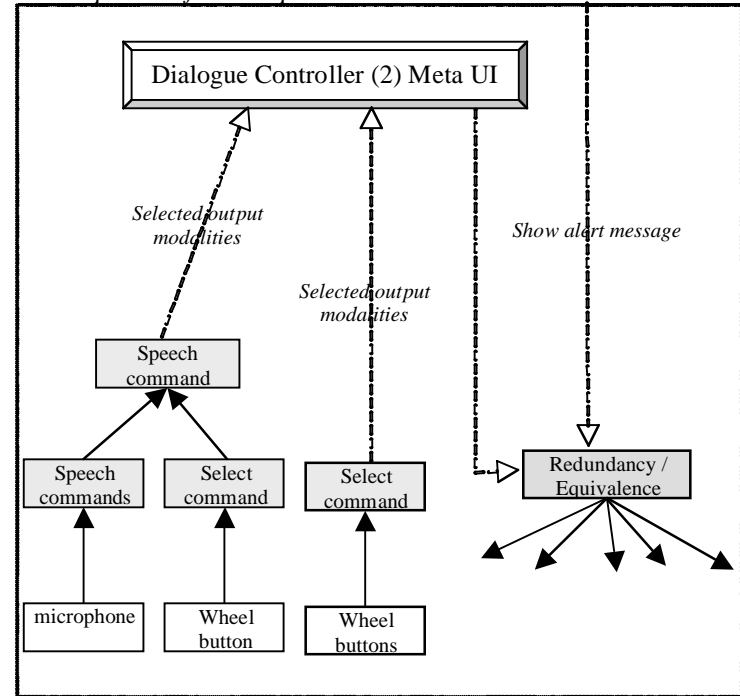


Fig. 11 Implemented ICARE components for the output modalities and meta User Interface. All the ICARE components are encapsulated in one OpenInterface component.

- One OpenInterface component corresponds to the “Eye Blinking history” for predicting the user's state of Fatigue.
- Two OpenInterface components correspond to the ICARE diagram of Figure 10 for capturing the user's state (Attention & Fatigue). Figure 12 presents the implemented processes of these two implemented

OpenInterface components.

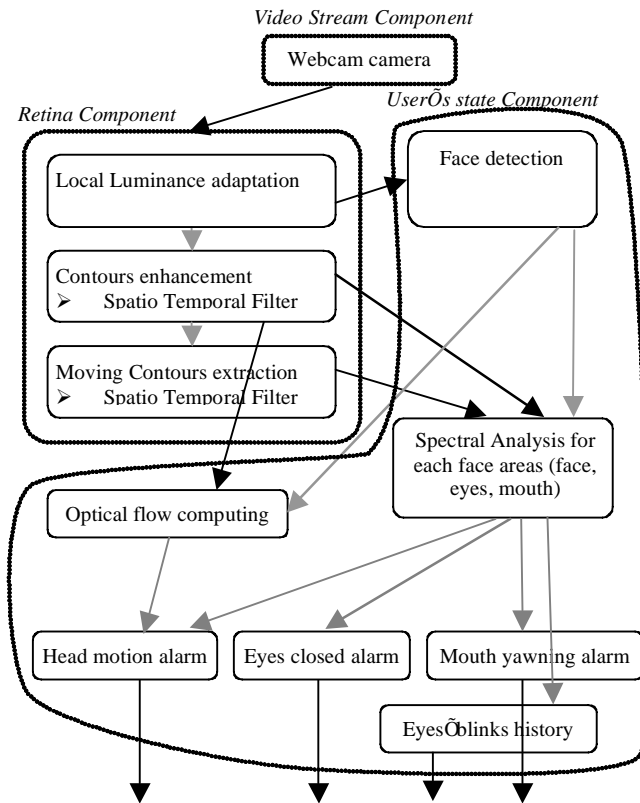


Fig. 12 Implemented OpenInterface components for capturing the user's state. Starting from the input provided by the Video Stream component, two OpenInterface components, namely Retina component and User's State component, have been implemented for providing four outputs: Focus of attention (head motion), duration of eyes closed, yawning and eye blinking history.

The video analysis system for capturing user's state is composed of two OpenInterface components: a prefiltering component that enhances the input data and extracts different information. The second component computes the user face analysis and outputs different indicators related to the user's state.

Retina Component description

Once a frame is acquired from the video stream component, it is processed by the Retina component. This component is a filter coming from the modeling of the human retina [12, 13]. It provides three outputs for each frame:

- a gray picture close to the input frame but with a corrected luminance. This output allows a better extraction of the details of the picture in the dark areas by enhancing locally the sensitivity to the luminance.
- a picture of all the contours in the input frame. This output contains only the contours of the input. It is robust against spatio-temporal noise and luminance variations. It allows a description of the details of the input such as eyes and mouth contours which are used by the fatigue detection.

- a picture of all the moving contours. This output only reports energy on the areas in which contours are moving. It allows event detection and motion description [14].

As an illustration, Figure 13 shows the three outputs of this component according to a frame input. The data provided by this Retina component are directed to the different modules of the User's State component.

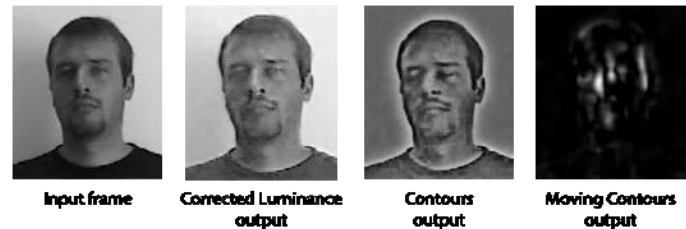


Fig. 13 Illustration of the outputs of the Retina component.

Description of the User State component

The first module of the User State component provides the position of the head in the visual scene, and is made of the Machine Perception Toolbox [14]. This module accepts as input a gray level picture. Nevertheless, luminance variations on the face can make this module fail. Then, in order to make it more robust, its input is the corrected luminance output of the Retina component instead of the Video Stream Component.

Once the face is detected, two modules work in parallel. The first is the Optical Flow Computing module, which computes the velocity on the face area with the help of neuromorphic velocity filters [15]. This module provides the horizontal and vertical estimated velocities. The second module is the Spectrum Analysis module. It consists of the log polar spectrum analysis of both contour output and moving contour output of the Retina component. This module is based on the modeling of the primary visual cortex area V1. As explained in [13] [16], by analyzing the temporal response of the log polar spectrum of the moving contours response of the face, it is possible to retrieve motion event alarms and motion orientation when motion is occurring. Finally, this module provides alarms for different face motions: the global head motion, eyes and mouth motions (opening/closing). Also, by analyzing the temporal evolution of the Retina component contour output, it is possible to evaluate the state "Open" or "Close" of the eyes and the mouth yawning.

Outputs generation

The User State module provides different outputs that are used by the components presenting the alarms.

Three outputs are alarms related to the estimation of the driver fatigue level. An alarm is sent when the user closes his eyes for more than a specified duration (we experimentally fix it to 200ms). Another is sent when the driver yawns.

Also, an alarm is generated when the user moves his head longer than a specified period (we experimentally fix it to 300ms). The generation of this alarm is based on the data provided by the Optical Flow Computing module and the

global head spectrum analysis. Once a head motion event is detected by the Spectrum Analysis module, the velocity data coming from the Optical Flow module and motion orientation coming from the Spectrum Analysis module are fused to generate the appropriate alarm in the event that the information is redundant.

These alarms are developed to signal user fatigue dynamically. In order to provide a long term prediction of hypo-vigilance, we generate a last output which is a list of the duration of the eye blinks encountered in the last 20 seconds. This output is sent to the hypo-vigilance prediction component.

Sleep prediction component

The aim of this component is to provide the driver with a warning several minutes before he/she loses control of the vehicle due to extreme hypo-vigilance or sleep. The prediction is made based on the 20 second eyelid activity history of the subject. Specifically the input of the component is the start and end timestamps of the blinks as these are registered by the video analysis system.

The output of the component is a binary value 1 or 0 corresponding to warning or no warning.

The prediction of the component is calculated via the fuzzy fusion of several features that characterize the blinking behavior of the driver (Fuzzy Expert System). These features that were selected based on literature review [17], [18] and the expertise gained in previous related projects such as AWAKE [19] are the following:

- **Long blinks duration:** the blinks in the 20 second window are filtered and only the ones lasting more than 0,3s are kept. If the number of long blinks is larger than 2 the sum of their durations is the long blink duration feature. Else the LBD = 0.
- **Maximum interval between blinks** is defined as the interval between the end of the current blink and the beginning of the next ($t1[\text{blink}+1] - t3[\text{blink}]$).
- **Blinking rate.**

Although these features are not the most efficient ones they were the only ones that could be extracted given the input data and the camera used for video acquisition (30fps). Features that take into account velocity characteristics of the blinks are reported to have greater accuracy [20], however for the extraction of these features a high speed camera capable of 200fps and special software is needed.

In the following figure a schematic representation of the fuzzy system's premise space is shown. The features form a three dimensional space and their partitioning using three fuzzy sets per input leads to the formation of 27 fuzzy rules. Each fuzzy rule has a different output thus giving us the ability to model 27 different blinking behaviors prior to the sleep onset. The final output/prediction of the system is calculated by combining the outputs of the fuzzy rules that are triggered by the eyelid activity pattern (LBD | Max interval | blinking rate) on real time.

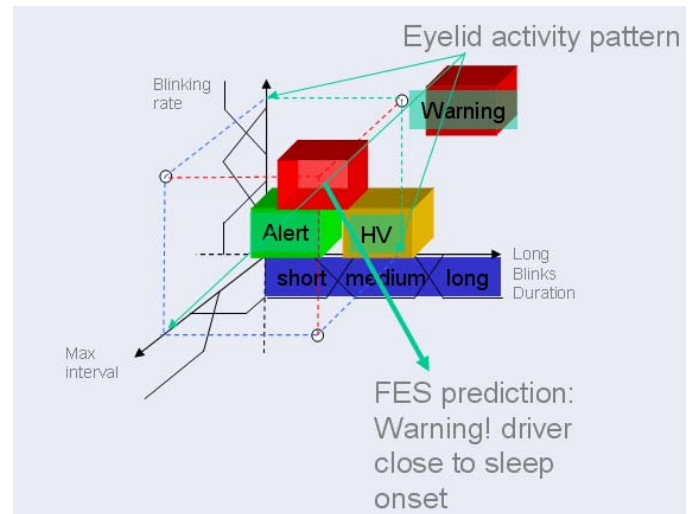


Fig. 14 A schematic of the FES premise space. Depending on which fuzzy rules are triggered by the eyelid activity pattern the output of the system is calculated in real time.

For the training of the fuzzy system's parameters data from 30 drowsy drivers were used, namely the blinking history of the subjects and the timestamps of the accidents during the driving sessions.

The method that was used for training was a real-coded genetic algorithm and the fitness function was chosen so as to maximize the correct predictions ratio and minimize the number of alarms so as to be as unobtrusive to the driver as possible [21]. Even though the training of the FES parameters with a GA takes a substantial amount of time that can reach one hour, once the parameters are trained the system generates its output instantly for online operation. Tests that were carried out during the workshop using this data led to a prediction accuracy of 80% for the training set of 30 drivers.

The component was developed in C++ and was delivered in the form of a dll for integration.

B. Discussion: tradeoffs and compatibility between ICARE and OpenInterface

There is no direct 1-1 mapping between the ICARE component architecture and the implemented OpenInterface component architecture. Nevertheless we demonstrated the compatibility and feasibility of the approach.

For the user's state capture, the two implemented OpenInterface components define large components. The componentization as described in Figure 10 would have been a difficult task since the code is developed in Matlab. Matlab had been initially used for exploring solutions. For providing final components after a feasibility phase made in Matlab, it would be useful to fully redevelop the final version in C++. Moreover we did not define one component for each feature used in the image, as advocated by Figure 10, for efficiency reasons because this would involve duplication of the video stream input.

For the output multimodal interface, we show the benefit of the ICARE approach, that is, that it allows us to quickly add a

new equivalent modality for selecting the output modalities within the meta User Interface and that it allows us to reuse components such as the Device component Loudspeakers for lexical feedback from the speech recognizer.

More OpenInterface components could have been defined corresponding to the ICARE software architecture: this was not pursued simply due to time constraints.

V. CONCLUSION

By considering the case study of a driving simulator, we focused on designing a software component architecture for multimodal interfaces from the Human-Computer Interaction domain, and how to implement it using the OpenInterface as well as the ICARE platforms. The compatibility of the two platforms is evident since several ICARE components are encapsulated within one OpenInterface component.

In future work, we first need to integrate the user's state prediction component within the demonstrator. We also plan to define new OpenInterface components particularly for the developed output multimodal interfaces. Moreover new native OpenInterface connectors could be defined corresponding to the ICARE output composition components. This work has already been done for the input ICARE composition components although we did not use them in this case study.

Moreover we would like to use new passive modalities for capturing the stress level of the user based on biological signals analysis. We are currently defining the corresponding OpenInterface components. We plan to integrate the stress level within our demonstrator as part of the meta User Interface for automatically selecting the output modalities in addition to allowing the user to select them.

Finally we would be interested to perform some usability experiments and to study the benefit of our component architecture in quickly modifying multimodal interaction and retesting the interaction as part of an iterative user centered design method.

REFERENCES

- [1] *SIMILAR, European Network of Excellence*, WP2, OpenInterface platform. www.similar.cc
- [2] J. Bouchet and L. Nigay, "ICARE: A Component-Based Approach for the Design and Development of Multimodal Interfaces", in *Proc. CHI'04 conference extended abstract*, ACM Press, 2004, pp. 1325-1328.
- [3] J. Bouchet, L. Nigay and T. Ganille, "ICARE Software Components for Rapidly Developing Multimodal Interfaces", in *Proc. ICMF'04 conference*, ACM Press, 2004, pp. 251-258.
- [4] A. Benoit et al., "Multimodal Focus Attention Detection in an Augmented Driver Simulator", in *Proc. eNTERFACE'05 workshop*, 2005, pp. 34-43. www.INTERFACE.net/INTERFACE05/
- [5] J-F. Kamp, "Man-machine interface for in-car systems. Study of the modalities and interaction devices", Ph.D. dissertation, ENST, Paris, 1998.
- [6] M. Tonniss, C. Sandor, G. Klinker, C. Lange, H. Bubb, "Experimental Evaluation of an Augmented Reality Visualization Car Driver's Attention", in *Proc. ISMAR'05*, IEEE Computer Society, 2005, pp. 56-59.
- [7] TORCS Driver Simulator: torcs.sourceforge.net
- [8] JavaBeans 1.01 specification, Sun Microsystems 1997. java.sun.com/products/javabeans/docs/
- [9] L. Nigay, J. Coutaz, "A Generic Platform for Addressing the Multimodal Challenge", in *Proc. CHI'95 conference*, ACM Press, 1995, pp. 98-105.

- [10] L. Nigay, J. Coutaz, "The CARE Properties and Their Impact on Software Design", in *Intelligence and Multimodality in Multimedia Interfaces*, 1997.
- [11] B. Mansoux, L. Nigay and J. Troccaz, "Output Multimodal Interaction: The Case of Augmented Surgery", in *Proc. HCI'06 conference*, Springer-Verlag and ACM Press, 2006, to appear.
- [12] W. Beaudot, "The neural information processing in the vertebrate retina: A melting pot of ideas for artificial vision", PhD Thesis in Computer Science, INPG (France) december 1994.
- [13] A. Benoit, A. Caplier "Head nods analysis : interpretation of non verbal communication gestures " IEEE, ICIP, Genova, Italy, 2005
- [14] Machine Perception Toolbox (MPT) <http://mplab.ucsd.edu/grants/project1/free-software/MPTWebSite/API/>.
- [15] A. Torralba, J. Hérault "An efficient neuromorphic analog network for motion estimation." IEEE Transactions on Circuits and Systems-I: Special Issue on Bio-Inspired Processors and CNNs for Vision. Vol 46, No. 2, February 1999.
- [16] A. Benoit, A. Caplier "Hypovigilance Analysis: Open or Closed Eye or Mouth ? Blinking or Yawning Frequency ?" IEEE, AVSS, Como, Italy, 2005.
- [17] Yannis Damousis, Dimitrios Tzovaras: Correlation between SP1 data and parameters and WP 4.4.2 algorithms, SENSATION Internal Report, November 2004.
- [18] Alex H. Bullinger et al "Criteria and algorithms for physiological states and their transitions, SENSATION_Del_1_1_1.doc", SENSATION Deliverable 1.1.1, August 2004
- [19] A. Giralt et al. "Driver hypovigilance criteria, filter and HDM module", AWAKE Deliverable 3.1, September 2003.
- [20] Johns, MW The amplitude-Velocity Ratio of Blinks: A New Method for Monitoring Drowsiness.
- [21] I. G. Damousis et al "A Fuzzy Expert System for the Early Warning of Accidents Due to Driver Hypo-Vigilance", presented at the Artificial Intelligence Applications and Innovations (AIAI) 2006 Conference, 7-9 June, 2006, Athens, Greece.

A. Benoit was born in 1980 in France. He graduated from Institut National Polytechnique de Grenoble (INPG). Currently he is a Ph.D. candidate in the Laboratoire des Images et des Signaux (LIS) of Grenoble. His research interests are in the areas of human motion and head motion analysis. His work is based on the human visual perception system. He is also teaching signal processing at the Master level.

L. Bonnaud was born in 1970 in France. He graduated from the École Centrale de Paris (ECP) in 1993. He obtained his PhD from IRISA and the Université de Rennes-1 in 1998. Since 1999 he is teaching at the Université Pierre-Mendès-France (UPMF) in Grenoble and is a permanent researcher at the Laboratoire des Images et des Signaux (LIS) in Grenoble. His research interests include segmentation and tracking, human motion and gestures analysis and interpretation.

A. Caplier was born in 1968 in France. She graduated from the École Nationale Supérieure des Ingénieurs Électriciens de Grenoble (ENSIEG) of the Institut National Polytechnique de Grenoble (INPG), France, in 1991. She obtained her Master's degree in Signal, Image, Speech Processing and Telecommunications from the INPG in 1992 and her PhD from the INPG in 1995. Since 1997, she is teaching at the École Nationale Supérieure d'Électronique et de Radioélectricité de Grenoble (ENSERG) of the INPG and is a permanent researcher at the Laboratoire des Images et des Signaux (LIS) in Grenoble. Her main interest concerns the analysis and the interpretation of human motion. She works in the domain of facial expressions classification, human postures recognition, Cued Speech language classification and head rigid or non rigid motion analysis.

Y. Damousis was born in 1974 in Thessaloniki-Greece. He received the Dipl. Eng. Degree and a Ph.D from the Department of Electrical and Computer Engineering at the Aristotle University of Thessaloniki in 1997 and 2003 respectively. Currently he is a senior researcher at the Informatics & Telematics Institute of the Centre for Research and Technology Hellas in Thessaloniki. His research interests are in the areas of expert systems, optimization and fusion in AI applications. He is a member of the Technical Chamber of Greece.

D. Tzovaras received the Diploma in electrical engineering and the PhD degree in 2D and 3D image compression from Aristotle University of Thessaloniki, Thessaloniki, Greece, in 1992 and 1997, respectively. He is a senior researcher in the Informatics and Telematics Institute of Thessaloniki. Prior to his current position, he was a senior researcher on 3D imaging at the Aristotle University of Thessaloniki. His main research interests include virtual reality, assistive technologies, 3D data processing, medical image communication, 3D motion estimation, and stereo and multiview image sequence coding. His involvement with those research areas has led to the coauthoring of more than 35 papers in refereed journals and more than 80 papers in international conferences. He has served as a regular reviewer for a number of international journals and conferences. Since 1992, he has been involved in more than 40 projects in Greece funded by the EC and the Greek Secretariat of Research and Technology. He is an associate editor of the EURASIP Journal of Applied Signal Processing and a member of the Technical Chamber of Greece..

F. Jourde was born in 1981 in France. He graduated in computer science from the University of Grenoble 1. He is currently working as a research associate at the CLIPS laboratory of Grenoble. His research interests focus on Computer-Human Interaction (HCI) and in particular his research studies centre on formal specification of multimodal user interfaces and formal tests of multimodal interaction based on Lustre, a synchronous programming language.

L. Nigay was born in 1965 in France. She is a Professor at Université Joseph Fourier (UJF, Grenoble 1) and at Institut Universitaire de France (IUF). Her research interests focus on the design and development of user interfaces. In particular her research studies centre on Multimodal and Augmented Reality (AR) user interfaces such as the component-based approach named ICARE (Interaction Complementarity, Assignment, Redundancy and Equivalence) for the development of multimodal and AR interfaces and new interaction modalities combining the real and the physical worlds such as tangible user interfaces, embodied user interface and mobile augmented reality. She has published more than 130 articles in conferences, journals and books. L. Nigay has received several scientific awards (including the CNRS Bronze medal in 2002 and the UJF gold medal in 2003 and again in 2005) for excellence in her research and is involved in many international scientific societies and events, as well as European research projects.

M. Serrano was born in 1981 in Spain. He graduated in computer science both from the Facultad de Informatica of the Universidad Politecnica de Madrid in Spain and from the École Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble (ENSIMAG) in France, in 2005. He is currently working as a research associate at the CLIPS laboratory of Grenoble. His research interests focus on Computer-Human Interaction (HCI) and in particular his research studies centre on output multimodal interaction for augmented surgery and multimodal interaction on mobile devices such as phone and PDA.

L. Lawson was born in 1982 in Bénin. He graduated from the Engineering School of Université Catholique de Louvain (UCL) and obtained his Master degree in Computer Science and Engineering in 2004. He is currently working as a research associate at the Communication and Remote Sensing Laboratory (TELE) of Université Catholique de Louvain on the development of OpenInterface, an open source component-based platform.