Journal on Multimodal User Interfaces, Vol. 1, No. 1, March 2007

AN AGENT BASED MULTICULTURAL TOUR GUIDE SYSTEM WITH NONVERBAL USER INTERFACE

Hung-Hsuan Huang, Kateryna Tarasenko, Toyoaki Nishida

Graduate School of Informatics, Kyoto University, Japan
{huang;ktarasenko}@ii.ist.i.kyoto-u.ac.jp;
nishida@i.kyoto-u.ac.jp

Aleksandra Cerekovic, Vjekoslav Levacic, Goranka Zoric, Igor S. Pandzic

Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
{aleksandra.cerekovic;vjekoslav.levacic;goranka.zoric;igor.pandzic}@fer.hr

Yukiko Nakano

Department of Computer, Information and Communication Sciences, Tokyo University of Agriculture & Technology, Japan
nakano@cc.tuat.ac.jp

ABSTRACT

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

KEYWORDS

Embodied conversational agent – Distributed system – User interface – Non-verbal interaction

1. INTRODUCTION

Embodied conversational agents (ECAs) [1] 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 inputs
- Generate verbal and nonverbal outputs
- Perform conversational functions

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 and emotional modeling, natural language generation, facial expression generation, gesture generation as well as an animator may be required. These functions involve multiple disciplines include artificial intelligence, computer graphics, cognitive science, sociology, linguistics, psychology, etc. They are in a 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 purposes. Hence, if there is a common and generic backbone framework that connects a set of reusable and modulated ECA software components, the rapid building of ECA systems will become possible while the redundant efforts and resource uses can be prevented. For these reasons, our group is developing a generic ECA platform and researching the adequate communicative interfaces between ECA software blocks.

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 who speak different languages and show 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.

To investigate the usefulness of our generic ECA platform in rapid ECA prototyping and explore the issues that may occur in multicultural and multimodal communication, we proposed a short-term project to develop a tour guide application with an ECA interface which serves human users in different cultural styles based on our generic ECA (GECA) framework [2] in the eNTERFACE '06 workshop [3]. This international workshop focuses on the topic of multi-modal human computer interface and was held in Dubrovnik, Croatia in the summer of 2006. Unlike usual workshops that researches only present their research results without actual work, the nature of this workshop is to invite volunteer participants to jointly develop proposed application projects in a relatively short four-week period and then present their research results.

The title of our proposed project was "An Agent Based Multicultural User Interface in a Customer Service Application." Although only one participant does not belong to our research group, only two of the six student participants were the core developers of the GECA project and thus most of the team members were supposed not to be familiar with the GECA platform. The result of this student project is a tour guide agent of the venue of the workshop, Dubrovnik city where is specified as a UNESCO Worlds Heritage. This paper describes the work to

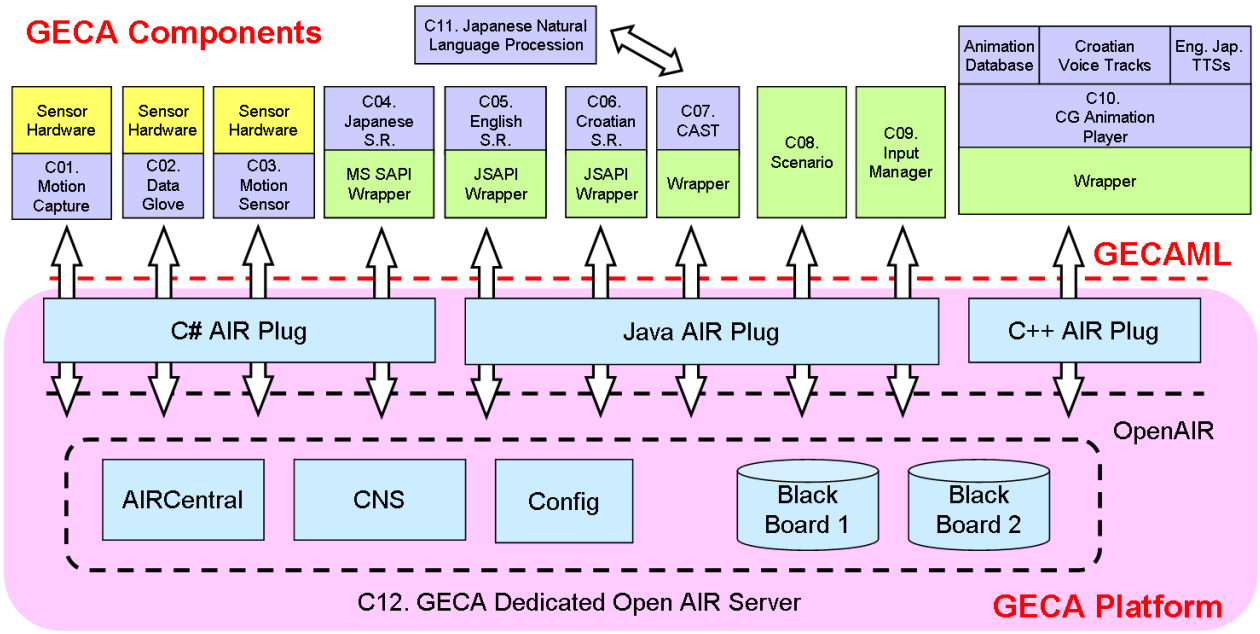

Figure 1: *The conceptual diagram of GECA framework and the configuration of the eNTERFACE06 agent.*

use the GECA Framework to build this Dubrovnik tour guide agent.

2. GENERIC ECA FRAMEWORK

To connect many heterogeneous functional components to build an integral virtual human, the consistency of all communication channels, timing synchronization of all components, and handling streaming data from sensors in real-time is indispensable. The base bone platform of GECA Framework is built upon a routing and communication protocol of cooperating A.I. programs, OpenAIR [4]. The platform mediates the information exchange of ECA software components with XML messages via blackboard model mechanism (or whiteboard in OpenAIR's context) and has 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
- Explicit timing management mechanism
- The use of shared backbone blackboards flattens the component hierarchy, shortens the decision making path and makes reflexive behaviors to be easier implemented
- Possible to use multiple logically isolated blackboards rather than traditional single blackboard
- Easy to switch or replace components which have the same function if they understand and generate messages in the same format

Figure 1 shows the conceptual diagram of the GECA Framework and the configuration of the system. Based on this framework, we are specifying an XML based high-level protocol (Generic ECA Markup Language, GECAML) 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.speech.text`", "`output.scenario`", "`output.player.control`", etc. Each message type has a specified set of XML elements and attributes, e.g., "`Intensity`", "`Duration`", "`Delay`", etc. for an non-verbal action of the character. Since the deliberate part (the thought of the agent) involves various levels of temporality / abstraction and is very complex, at present we only take account of the input (user behavior) / output (agent behavior) control of the agent system and leave it as a black box.

Currently, the inputs of the deliberate block which is mainly a scenario script interpreter are text streams recognized from the human users' utterances that are annotated with non-verbal action tags. The outputs are the utterances of the conversational agents that are going to be spoken by a TTS engine and annotated with markups to specify facial expressions, body movements, gestures, and other non-verbal behaviors of the animated character.

The message flow works like the following scenario on the platform: when a component starts, it registers its contact information (unique name, IP address, etc) to the `CNS` (CentralNaming Service) component and subscribes its interested message type(s) to the `AIRCentral` component. Then the messages in those types will be sent to the component from the specified blackboard which behaves as 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 specific message type(s).

By utilizing the communicative functions provided by the Air Plug libraries (currently C#, C++ version libraries and a customized Java reference implementation from mindmakers.org have been developed) 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 compliant. After

doing this, the heterogeneous nature of components that provide the same capability (both MS SAPI TTSs and JSAPI TTSs provide the same capability of the agent, i.e. to speak out from text) can be hided and behave identically to the other software components. For additional information about GECA Framework, please see [2].

3. THE DUBROVNIK CITY TOUR GUIDE

The planned Dubrovnik city tour guide agent dynamically changes its behaviors among American, Japanese or Croatian style according to where the visitor comes from in verbal and nonverbal interactions. One of the possible scenarios is: when a Japanese visitor comes to the system, the system recognizes that the visitor is a Japanese from the combined result of the speech and non-verbal behavior recognizers. An example for such culture-specific nonverbal behaviors is bowing for 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 nonverbal 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. Most of the functioning components are meant to be developed during the eNTERFACE workshop while the others are existing program components. In this section, we discuss the GECA components that are used to build the tour guide agent of Dubrovnik city. Section 3.1 is a brief introduction of the component configuration and sections 3.2, 3.3, 3.4, 3.5 are the main tasks dealt during the workshop period.

3.1. Software Component Configuration

This agent is composed with the component configuration depicted in figure 1. The follows are the brief descriptions of the software components.

- C01. Motion capture component. This component utilizes a simple motion capture device using IR technology [5] to approximate the position of the human user's right hand. This device was chosen because its compact physical size to be carried to the oversea workshop venue and the low cost
- C02. Data glove component. This component acquires data from a data glove device and reports recognized shapes of the visitor's fingers to the other components
- C03. Motion sensor component. This component acquires data from a three dimensional acceleration sensor [6] which is attached on the visitor's head to detect head shaking and nodding movements
- C04. Japanese speech recognition component. This component is a wrapped MS SAPI5 [7] Japanese recognition engine which recognizes Japanese spoken by the visitors by matching predefined grammar rules and sends the recognized result as a text string to the subscribed components
- C05. English speech recognition component. This has the same function as component C04 but is a wrapped Java SAPI [8] speech recognition engine. Java SAPI is used here for keeping multiple speech recognizers on the same computer so that only one microphone is required to gather the user's speech
- C06. Croatian speech recognition component. Because the lack of a good enough speech recognizer for Croatian language, Croatian is recognized by an English speech recognizer with the grammar rules which defined a limited range of Croatian vocabularies by approximating their pronunciations with English alphabets
- C07. Japanese spontaneous gesture generator. This component is a wrapper of CAST [9] engine that generates type and timing information of spontaneous gestures from Japanese text string by using a statistical model built from analyzing human presentation video
- C08. GECAML scenario component. This component is an interpreter of GECAML scenario scripts. It reads a script that specifies the agent's verbal and nonverbal responses corresponding to specific behaviors from the visitors. This script is described in an XML based language representing Question and Answer pairs and is inspired from AIML [10]. A set of ECA specific capabilities distinguish it from AIML includes conversation state, nonverbal interactions, and objects in scene descriptions. All of the three language versions of the scenario are described in the same script
- C09. Input manager component. The component combines sensor data streams which have no direct meanings to individual events representing human user's nonverbal behaviors. Currently it combines data streams from motion capture component and data glove components and detects user's nonverbal behaviors such as waving and pointing
- C10. CG Character animation player component. This component is a wrapped character animation player which is implemented by using the visage|SDK [11]. It accepts event messages from the other components and drives the CG character to speak or perform nonverbal behaviors. This player calls MS SAPI English and Japanese Text-To-Speech (TTS) engines installed in the same machine to generate the voice output of the agent and viseme events to drive the character animator to move the agent's lips. Because the lack of a good quality Croatian speech synthesizer, Croatian speech is done via prerecorded voice tracks and handled by the player. All of the nonverbal animations of the character are defined in an animation category included inside the player. The animation definitions in the category are not merely canned tracks but are parameterized and can be changed during runtime
- C11. Japanese natural language processing component. This component includes a Japanese morphology analyzer, Juman [12] and a Japanese syntax analyzer, KNP [13], the two analyzer label an input text string with linguistic information that will be used by C07. The communication between C11 and C07 is in their original way and is not included in GECAML
- C12. GECA dedicated OpenAIR server. This server mediates the communication among all of the components with a shared memory (blackboard) by subscribe-publication mechanism. When one of the components connected to the system starts up, it registers its interested message types to the server. Components process their input data and send the output messages to the server, the messages are then forwarded by the server to the components which have registered those types. The CNS, AIRCentral, and whiteboard(s) described in section 2 are included in this server

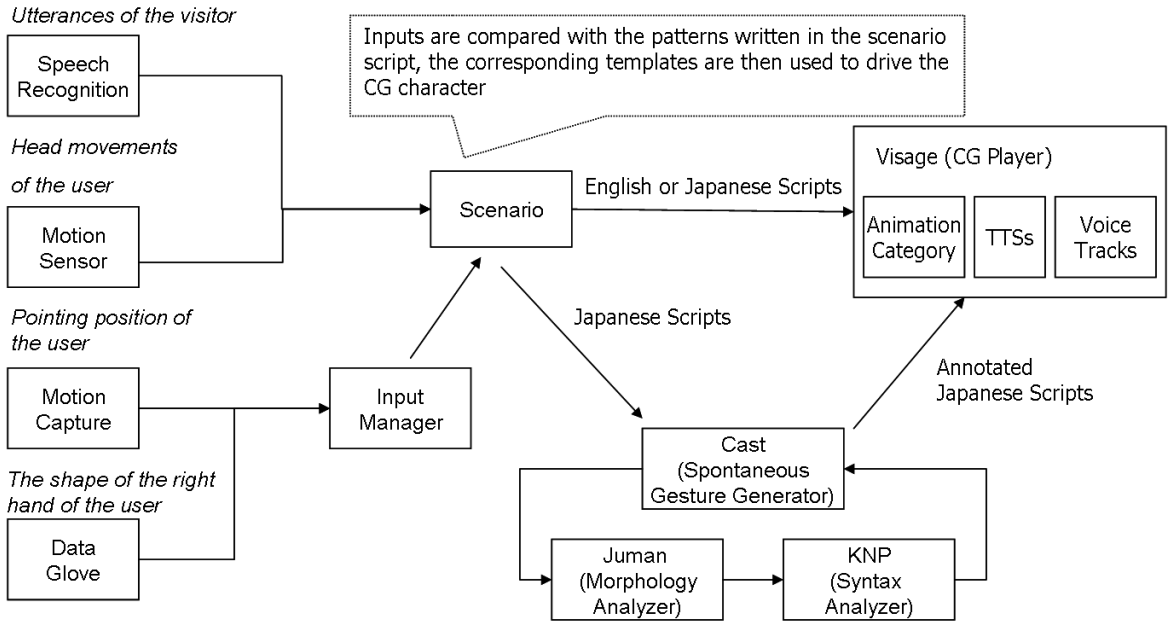

Figure 2: *The data flow of the Dubrovnik tour guide agent.*

Among these components, C01, C02, C05, C06, and C09 are newly developed while C07, C08, C10, C12 are modified and improved during the workshop period. Figure 2 shows the data flow among these components. 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 scenario component. The inputs from different modalities are combined by the script interpreter and are matched with the patterns in the scenario script. It then sends the matched response which may include utterance and action tags to speech synthesizer and animation category component. The spontaneous gesture generator inserts action tags into the utterance according to the linguistic information from its natural language tagging companion. The character animator listens to action and speech events and play them in real-time.

Nonverbal animations are looked up from the animation category and filled with the parameters defined in the event messages. Current character animator (visage) also provides TTS support. 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.

3.2. Nonverbal User Behaviors 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

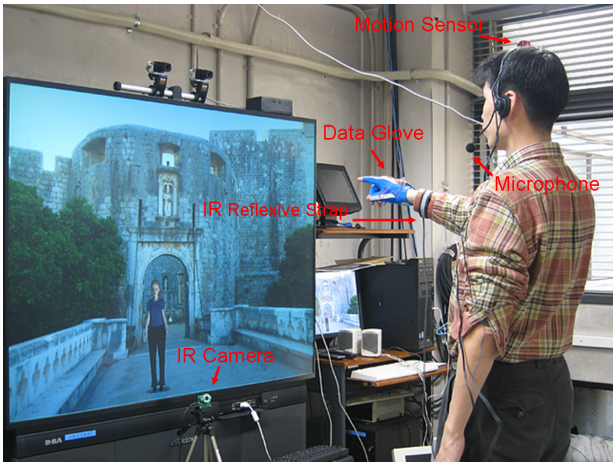

Figure 3: *The hardware configuration of the Dubrovnik city tour guide agent. Note that during the workshop, a LCD projector settled behind the user is used instead of the large display shown in the figure.*

- Interrupt the agent's utterance
- Shake head and nod to express positive and negative answers

The non-verbal behaviors are recognized by using the data from data gloves, infrared camera, and acceleration sensors. The configuration of these hardware devices is shown in figure 3.

3.2.1. *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. We

assign a threshold value of when finger becomes bent, which means that we detect only two states of the fingers. By mapping finger shapes into the gestures, this component detects two statuses of hand shapes, pointing or five fingers straight.

3.2.2. NaturalPoint OptiTrack FLEX3

Infrared reflecting 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 a 2D plane representing the screen. The problem that the number of infrared sources captured by the IR camera does not match the actual numbers due to obstacles or the user's own body is solved by K-Means algorithm to group close mass centers. This component sends the position of the user's right hand continuously to the shared blackboard which be used by input manager component to detect pointing position, hand stability and waving.

Data glove and hand movement detection programs work as separate GECA components. Each program has an OpenAir plug implementation, sending the data which `InputManager` component receives and combines into the new output. `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.

3.2.3. 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".

3.3. Croatian Speech Recognition and Synthesis

In the current implementation, English and Japanese spoken by the user are recognized by using commercial MS SAPI5 speech recognition engines. Since there is no Croatian speech recognizer neither TTS (Text-To-Speech) synthesizer exists in Croatian mode, we used English speech recognizer to recognize Croatian language speaker and prerecorded Croatian voices. At first, recognition grammar rules are created to recognize the vocabularies used by the Croatian scenario. The grammar for Croatian is defined by using English alphabet to approximate the pronunciation of Croatian. We have used such grammar rules to recognize both Croatian words and sentences. The result was very successful if the Croatian speaker pays some attention in his (her) pronunciation. However, some Croatian words are impossible to be expressed by English alphabets; therefore, some additional efforts may be required in composing the Croatian scenario script to prevent to use such words. After the Croatian scenario is created, all sentences appear in the script are recorded in a silent root with a native Croatian speaker.

The corresponding lip synchronization animation sequence is then generated by a system which is developed by one of the authors [14]. It determines appropriate motions of the character's mouth and tongue during the speech by voice signal analysis and pre-trained neural networks. The generated animation sequences are saved as MPEG-4 FBA parameters in files and are played later with the voice track at the same time by the animator.

3.4. Action Animation Catalogue

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 scenario script. A single animation catalogue query corresponds to a 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. Below is a simple example of a GECAML scenario category with non-verbal input/output descriptions:

```
<Category>
  <Pattern>What is this?
    <Action Type="Pointing" Target="Fountain"/>
  </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 responding to a combination of the user's verbal / non-verbal behavior is described. The target "`Fountain`" where the user is pointing is defined as a squared area of the background image in the scene definition block elsewhere in the scenario script. The duration of the agent's pointing action 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, the meaning of the coordinates ("`X`", "`Y`", "`Z`") is dependent on the action. For example, for the "`Walking`" action such this triad would mean the destination in the virtual 3D space where the agent is supposed to walk to. The "`Direction`" attribute specifies the direction information of the action, for example, the agent will point to its right-up direction in the code segment above.

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.

The analysis of videos to create gestures was difficult because we did not use any tool that would translate a gesture in

Figure 4: *A Japanese emblem gesture to show prohibition.*

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).

3.4.1. Cultural differences through the non-verbal behavior of the agent

We took videos of Japanese tour guides' non-verbal behaviors at some sight-seeing spots in Kyoto and those of European tour guides in Dubrovnik and then analyzed the video data. As a result, we tried to implement those 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 4 and figure 5). 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. 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.

3.5. Character-Background Alignment

In the system data flow as shown in figure 2, character animator -Visage Player uses animation database with MPEG-4 parameters to generate animation of agent guide. To reduce the system complexity and prevent the drawbacks come from an awkward 3D environment, in the prototype system, scene transitions are approximated by camerawork and the changes of realistic background photos instead of building a full 3D virtual world. Therefore, the position of the agent character relative to the real camera and the background image had to be calculated.

Figure 5: *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.*

As manual positioning of the agent character is not practical when number of the scenes increases, we have developed an automatic procedure to align the position of the character with the background images by an application developed with ARToolkit [15]. It detects real-world markers which defines a position of the agent character on the static background photograph. If the marker is detected, the parameters of camera position and orientation relative to physical markers can be calculated. These parameters, model view matrix and projection matrix, are then used to render the agent character aligned with that background picture. Two pictures of the same background thus have to be taken, one is with the mark and the other one is not. After a series of tests with various positions and orientations of the marker, we found that to use a bigger marker and to lay it on the floor without presence of any shadow on the marker gets best results. At last, the character can be successfully aligned

to background pictures with the background images at a fairly high ratio.

4. DISCUSSIONS AND FUTURE WORKS

At the end of the eNTERFACE workshop, a Dubrovnik guide agent runs in Japanese and English mode with 5 scenes (background images) and 16 QA categories is completed. It is driven by speech recognizer and provides scene changing feature. Although the ambitious planned system could not be completed before the end of the very short four-week period, nearly all of the individual components are implemented but just were not put together and fully tested. Besides, after starting the project, we realized that it is difficult to distinguish where the user comes from non-verbal inputs with the very limited hardware configuration which was not a problem of GECA framework itself. We found that there is no obvious difference in non-verbal behaviors when people greet, a bow can be used to identify a Japanese user, but there is nearly no way to distinguish an American and a Croatian user.

After the eNTERFACE workshop, we are still improving the Dubrovnik tour guide agent. The current application is running in three language modes, English, Japanese and Croatian mode. 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 and Croatian mode, the agent performs scripted gestures only. Also, due to non-persistence of Croatian TTS and usage of prerecorded voice files and corresponding lip animation files, regression of abilities in Croatian mode is bounded to the speaker whom the prerecorded voice belongs. English speech recognition engine adopted for Croatian is working well when the key words don't match what can also cause boundaries in the Croatian version of the agent. However, we hope that soon good Croatian speech recognizer/synthesizer will appear.

The experimental Dubrovnik tour guide agent system is relatively simple; however, this work is not emphasizing on how strong the built ECA system is but is trying to ease the development and provide sufficient capabilities for general ECA researches. In this preliminary evaluation, the tour guide agent proves the platform's capability to seamlessly deal with multimodal inputs and sufficient performance for smooth real-time conversation. Besides the system can be built by incorporating software tools which are not specifically designed for these systems with little efforts (usually requires only several dozen lines of code), just by wrapping those tools according to the specification of GECA framework, and then an ECA system works.

Since the undoubted needs of a generic platform or protocol in the ECA research field, some activities have been launched, for example, the SAIBA model [16]. However, most of them only focus on modeling either the user behavior or the agent behavior without trying to describe the interaction flow. Figure 6 shows the relationship of SAIBA and GECA. GECA differs to SAIBA in the description of the user behaviors and the commands to adapt the virtual environment instead of describing agent behaviors only. Further, the GECA framework does not only specifies the behavior description but tries to provide a solution including the base bone platform program and the helping libraries to develop the functional plug-ins. Besides, SAIBA/BML suffers an innate weakness that the compilation of BML is required such that action interruption and reflexive behaviors of the agent is difficult to be implemented.

This work is a preliminary attempt to apply GECA Framework to the multi cultural domain where the I/O modalities (speech recognition, speech synthesis, gesture recognition and behavior synthesis) of the agent are polymorphic and is suitable to investigate the usefulness of the GECA architecture. We observed the verbal and non-verbal behaviors of actual Japanese and European tour guides at sight-seeing spots, implemented our findings of the typical behaviors in the Dubrovnik tour guide agent application. However, the cross-culture issue in the ECA-human interaction domain itself is not explored deeply in the one-month project. We have started a joint project, CUBE-G [17], with the University of Augsburg to explore the cross-culture issues in depth.

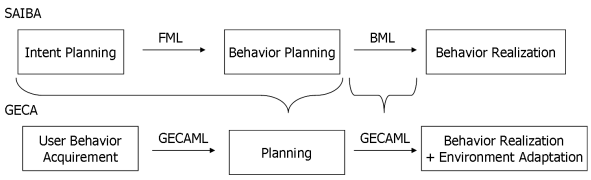

Figure 6: *The relationship between SAIBA and GECA activities. GECA differs to SAIBA in the description of the user behaviors and the commands to adapt the virtual environment instead of describing agent behaviors only.*

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. Later improvements of this system can include: Continuous scene transitions while the agent is walking, e.g. walking around circular fountain. External animation catalogue component and a script language to describe action animations instead of modifying the codes of the CG animator. Support of non MS SAPI5 TTS engines. Include more sensor hardware, for example, head tracker to allow more complex nonverbal interactions. Further, the virtual object reference is described by coordinate matching defined in the scenario script without considering complex context modeling [18], not only the current simple human / agent behavior description but also a deeper conversation modeling is required in the future to achieve more natural human / agent interactions.

5. REFERENCES

[1] J. Cassell, J. Sullivan, S. Prevost, and E. Churchill, eds., *Embodied Conversational Agents*. MIT Press, 2000. 41

[2] H. Huang, T. Masuda, A. Cerekovic, K. Tarasenko, I. Pandzic, Y. Nakano, and T. Nishida, "Toward a Universal Platform for Integrating Embodied Conversational Agent Components", in *Proceedings of the 10th International Conference Knowledge-Based Intelligent Information and Engineering Systems (KES 2006)*, (Bournemouth, UK), pp. 220–226, October 2006. 41, 43

[3] The eNTERFACE'06 workshop on multimodal interfaces. http://enterface.tel.fer.hr. 41

[4] OpenAIR protocol. http://www.mindmakers.org/openair/airPage.jsp. 42

[5] NaturalPoint OptiTrack Flex 3. http://www.naturalpoint.com/optitrack/. 43

[6] NEC/Tokin 3D motion sensor. http://www.nec-tokin.com/english/product/3d/. 43

[7] MS SAPI (Microsoft Speech Application Programming Interface). http://www.microsoft.com/speech. 43

[8] Java SAPI (Java Speech Application Programming Interface). http://java.sun.com/products/java-media/speech/. 43

[9] Y. Nakano, M. Okamoto, D. Kawahara, Q. Li, and T. Nishida, "Converting Text into Agent Animations: Assigning Gestures to Text", in *Proceedings of The Human Language Technology Conference (HLT-NAACL04)*, 2004. 43

[10] AIML (Artificial Intelligence Markup Language). http://www.alicebot.org. 43

[11] visage|SDK, Visage player, Visage Technologies. http://www.visagetechnologies.com. 43

[12] Juman. http://nlp.kuee.kyoto-u.ac.jp/nl-resource/juman.html. 43

[13] KNP. http://nlp.kuee.kyoto-u.ac.jp/nl-resource/knp.html. 43

[14] G. Zoric and I. S. Pandzic, "A Real-time Language Independent Lip Synchronization Method Using a Genetic Algorithm", in *Proceedings of ICME 2005*, July 6-8 2005. 45

[15] ARToolKit. http://artoolkit.sourceforge.net/. 46

[16] S. Kopp, B. Krenn, S. Marsella, A. Marshall, C. Pelachaud, H. Pirker, K. Thorisson, and H. Vilhjalmsson, "Towards a Common Framework for Multimodal Generation: The Markup Language", in *Proceedings of IVA2006*, August 2006. 47

[17] CUBE-G (Culture-adaptive Behavior Generation for interactions with embodied conversational agents). http://mm-werkstatt.informatik.uni-augsburg.de/projects/cube-g/. 47

[18] F. Landragin, "Visual Perception, Language and Gesture: A Model for their Understanding in Multimodal Dialogue Systems", in *Signal Processing*, Elsevier, 2006. 47