Voice Interaction System with 3D-CG Virtual Agent for Stand-alone Smartphones

Daisuke Yamamoto

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
daisuke@nitech.ac.jp

Keiichiro Oura

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
uratec@nitech.ac.jp

Ryota Nishimura

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
nishimura.ryota@nitech.ac.jp

Takahiro Uchiya

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
t-uchiya@nitech.ac.jp

Akinobu Lee

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
ri@nitech.ac.jp

Ichi Takumi

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
takumi@nitech.ac.jp

Keiichi Tokuda

Nagoya Institute of Technology
CREST, JST
Gokiso, Showa, Nagoya, Japan.
tokuda@nitech.ac.jp

ABSTRACT

In this paper, we propose a voice interaction system using 3D-CG virtual agents for stand-alone smartphones. Because the proposed system can handle speech recognition and speech synthesis on a stand-alone smartphone differently from the existing mobile voice interaction systems, this system enables us to talk naturally without encountering delays caused by network communications. Moreover, proposed system can be fully customized by dialogue scripts, Java-based plugins, and Android APIs. Therefore, developers can make original voice interaction systems for smartphones easily based on proposed system. We have made a subset of the proposed system available as open-source software. We expect that this system will contribute to studies of human-agent interaction using smartphones.

Author Keywords: Voice interaction system; Mobile application; Open-source software

ACM Classification Keywords

H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s). Copyright is held by the author/owner(s).

HAI 2014, October 29–31, 2014, Tsukuba, Japan.

ACM 978-1-4503-3035-0/14/10.

<http://dx.doi.org/10.1145/2658861.2658874>

INTRODUCTION

Recently, voice interaction systems for mobiles, such as Apple's Siri [1], have become widely popular. These systems enable users to obtain information—from map navigation to weather forecasting—by talking with virtual agents. These systems adopt a server-side speech recognition method to achieve a high speech recognition accuracy.

On the other hand, these systems have the following problems. One is that these system don't display 3D-CG virtual agents. We believe that 3D-CG virtual agent is effective for voice interaction systems to archive more user-friendly interface. The other is that the delay in voice interaction is considerably long because of both network communication and server-side processing costs. In general, the delay is an important factor for facilitating a natural voice interaction. Importantly, the more natural the voice interaction system is (such as in a voice interaction system with virtual agents that adopts real-time 3D-CG rendering), the more serious the delay problem becomes.

The purpose of this study is to develop a voice interaction system that can function on stand-alone smartphones with a virtual agent. They enable users to talk with 3D-CG virtual agents more naturally and smoothly. Moreover, the proposed system can be fully customized by dialogue scripts and Java-based plugins using the smartphone's functions effectively. Therefore, users can develop original voice interaction systems for smartphones freely and easily

based on proposed system. Concretely, we ported the existing toolkit for building voice interaction systems MMDAgent [2] to the Android OS and extended it for smartphones. MMDAgent archives not only advanced speech technologies but also various and detailed managements of virtual agents.

In order to meet the goals of this study, we need to solve the following problems.

Problem 1. The response time for the voice interaction system must be minimized. Response time is an important factor in natural voice interaction systems [3]. This hypothesis must be confirmed with subjective experiments.

Problem 2. The proposed system must be able to be extended easily by not only dialogue scripts but also plugin mechanisms to utilize smartphone-specific functions, such as email, calendar, and networking.

Problem 3. The power consumption for the proposed system must be minimized because it is a crucial issue for smartphone users.

In the PROBLEMS section, we describe in detail the problems outlined above. Next, in the PROPOSED SYSTEM section, we explain our method. The PROTOTYPE SYSTEM section discusses the prototype for the proposed system. In the EXPERIMENTS section, we evaluate the prototype in terms of its response time, power consumption, and subjective experiments. The RELATED WORK section describes comparable systems. Finally, the CONCLUSION section summarizes our research with concluding remarks.

Furthermore, we unlocked a subset of the proposed system to present it as open-source software¹. We expect that the proposed system can contribute to studies of human-agent interaction systems for smartphones.

PROBLEMS

Comparison of voice interaction methods for mobiles

First, in order to address **Problem 1**, we compared the various methods adopted by mobile voice interaction systems.

Many existing voice interaction systems for smartphones adopt server-side speech recognition (we call it the “cloud method”). The cloud method requires transfer time to record a voice into the smartphone and then upload it to the server before processing it in the recognition server. The cloud-method thus has the disadvantage of a response time for voice interaction that may be longer than one might expect in, say, a natural human dialogue.

In general, response time is an important factor for making natural interaction systems [4],[5]. Shiwa [6] suggests that

the response time should be within 2 s. Although the response time of existing cloud-method systems may take several seconds, the delay might not be noticeable because these systems adopt a question-and-answer type interface instead of a natural voice interaction interface. However, we believe that the delay will become significant as more natural voice interaction interfaces are developed, such as in using 3D-CG virtual agents.

Therefore, we compared and studied the following three methods to examine the delay.

Cloud method: this method records a voice in the smartphone and then uploads it to the server before recognizing it.

Streaming method: this method recognizes a voice in the server in real-time because the smartphone transfers the voice to the server in a streaming manner.

Stand-alone method: this method recognizes a voice in the smartphone in real-time without using any servers or networks.

Figure 1. Response time for voice interaction of each method. “Network” refers to the delay in network communication.

	Cloud	Streaming	Stand-alone
Cost for client	good	good	poor
Cost to network	fair	poor	good
Cost to server	poor	poor	excellent
Delay in interaction	poor	good	excellent

Table 1. Costs and delay in various voice interaction methods for smartphones.

Figure 1 and Table 1 show the features for each method. The cloud method has certain advantages. For instance, the computational cost for the smartphone itself is low because voice recognition is not performed by the smartphone. However, the cost to the server is high and the response time is long. Apple’s Siri adopts this method.

¹ <http://www.mmdagent.jp/>

The streaming method has the advantage that the computational cost to the smartphone is low while the response time is shorter than with the cloud method. However, that comes at a cost to both the server and network. This method requires constant communication with the server while talking with agents. Mobile Mei-chan [7] adopts this method.

The stand-alone method has the advantage that both the cost to the server and the delay in communicating is nil. Although the calculating cost to the smartphone is high, the processing speed of current smartphones is sufficiently high for recognizing a voice in real-time.

A toolkit for building voice interaction systems

We have developed MMDAgent in our past research. MMDAgent is the toolkit that includes advanced speech and graphics technology such as speech recognition, speech synthesis, 3D-CG rendering, dialogue management, and a physics engine. MMDAgent adopts Julius [8] as a speech-recognition engine, Open JTalk [9] for speech synthesis, OpenGL for 3D-CG rendering, MikuMikuDance [10] format for 3D modeling, and Bullet Physics [11] for the physics engine.

MMDAgent can manage voice interaction scenarios based on FST script. FST script, which is based on the FST (Finite State Transducer) format as shown in Figure 2, can handle dialogue scenarios by triggering various events including speech keywords, sensor values, timers, and so on. MMDAgent performs real-time voice recognition with a slight delay.

a) FST script

```

1 10 RECOG_EVENT_STOP|Hello <eps>
1 10 RECOG_EVENT_STOP|Hi <eps>
10 11 <eps> MOTION_ADD|mei|greet|greet.vmd
11 12 <eps> SYNTH_START|mei|normal|Hello
12 1 SYNTH_EVENT_STOP|mei <eps>

```

b) FST diagram

Figure 2. a) A sample FST script for MMDAgent and b) its diagram. In this FST, if the agent recognizes voice “Hello,” the agent greets and speaks “Hello.”

Although the existing MMDAgent was compatible with PC platforms including Windows, Linux, and Mac OS X, it did not support smartphone platforms such as Android OS. Therefore, we needed to port MMDAgent for PCs to Android OS in order to render it compatible with smartphones.

PROPOSED SYSTEM

Porting MMDAgent to Android OS

There were many challenges in porting the existing MMDAgent for PCs to Android OS. First, Android applications are expected to be written in Java whereas the existing MMDAgent was written in C++. Fortunately however, Android OS can support the C++ language by using Android NDK (Native Developer Kit) [12]. Therefore, we adopted Android NDK for porting MMDAgent. We used the OpenSL ES [13] library for Audio I/O.

Second, we changed Julius, the acoustic model, because the existing acoustic model was too demanding for smartphones. We adopted IPA’s PTM model, which is smaller in size than the existing acoustic model. However, we did not change the language model, which can handle about 60,000 words [14]. This language model is the same as that in the existing MMDAgent. The text-to-speech engine, Open JTalk, is also the same as that in the existing MMDAgent. Since a microphone of a smartphone is close to its speaker in general, moreover, we developed the echo suppressor to cut off microphone when playing voices.

Although we changed some of the specifications for MMDAgent, both MMDAgent for PCs and MMDAgent for smartphones are based on the same scripts and materials. Therefore users can develop a common dialogue content for both PCs and smartphones.

Plugin extension based on Android OS APIs

Android NDK doesn’t support some Android OS APIs such as mail management, calendar, live wallpaper, and application launcher. To address **Problem 2**, we developed a bridge module to connect existing C++ modules with Android’s Java modules, as shown in Figure 3.

To explain in more detail: with MMDAgent, all internal messages between modules are communicated via the Global Message Queue. The bridge module transfers the internal messages in the Global Message Queue to PluginListener, written in Java. The bridge module adopts JNI (Java Native Interface), which can communicate between C++ and Java. Similarly, internal messages based on the Java language can be transferred to the Global Message Queue. This mechanism provides an easy way to communicate easily between modules written in C++ and those written in Java.

Therefore, developers can extend the proposed system by implementing PluginListener class of Java. A sample implementation of PluginListener is shown in Figure 4. In this sample, if the plugin receives the internal message ‘Hello’ command, the plugin returns the internal message ‘Hello’ event. Moreover, the proposed system can support plugins based on Android’s standard GUI by using PluginListener. Figure 5 shows a sample of the GUI. In that sample, users can add text, select items, and push buttons.

Figure 3. System configuration of MMDAgent for Android. To use Android OS functions, we implemented the Java module and Audio I/O module using OpenGL ES.

```
public class Sample implements PluginListener{
    public void onCommandMessage(String message){
        if(message.equals("Hello"))
            MMDAgentJNI.sendMessage("Hello");
    }
    public void onEventMessage(String message){
        ...
    }
}
```

Figure 4. Sample implementation for PluginListener. Here, if the plugin receives the internal message 'Hello,' the plugin returns the internal message 'Hello.'

Figure 5. Sample plugin based on the standard Android GUI. In this sample, users can input using standard GUIs.

MMDAgent in live-wallpaper mode

The proposed system enables the live-wallpaper mode, as shown in Figure 6. Live-wallpaper is resident software operating in the background of Android smartphones. In live-wallpaper mode, users can use the voice interaction system without launching any applications because the system is already working as resident software. We expect that in the live-wallpaper mode, our proposed system will work more quickly and easily.

There may be a concern with power consumption because the proposed system in live-wallpaper mode requires running the speech-recognition process at all times. In order to reduce power consumption, the proposed system can halt the speech recognition process while other applications are being used. This simple technique will be extremely effective for reducing power consumption because wallpaper is not visible at all times.

Figure 6. The proposed system in the live-wallpaper mode.

PROTOTYPE SYSTEM

We developed two prototypes. One is the application mode for smartphones shown in Figure 5. The second is the live-wallpaper mode shown in Figure 6. Live wallpaper is resident software working in the background of Android smartphones.

This section describes the prototype system and its interfaces in detail.

Virtual agent

For this proposed system, we created a disproportionately shaped woman as the virtual agent we call "SD Mei-chan." Because SD Mei-chan's head is so big, users can more easily understand her facial expressions even if the display is relatively small. Icons near the virtual agent show the active functions for voice interaction. For example, the weather forecast icon permits users to talk about the weather with the agent. All objects including the virtual

agent, icons, and images are rendered as 3D computer graphics.

Table 2 presents a list of the motions, expressions, and speaking styles for SD Mei-chan. SD Mei-chan expresses herself vividly by combining these motions, expressions, and speaking styles. Since these resources can be created by using free software, her motions and voices can be customized freely if necessary.

Type	Resources
Motion	breath, good-bye, greeting, guide, idle, imagine, laugh, look panel, point, self-introduction, surprise, wait
Expression	anger, bashfulness, happiness, listening, sadness, normal
Speaking style	angry, bashful, happy, normal, sad

Table 2. List of resources for motion, expression, and speaking style.

Basic voice interaction

The voice interaction function can be used in the proposed system exactly as it is in the existing MMDAgent for PCs. Dialogue scenarios can be written with FST script as well as with the existing MMDAgent.

Figure 7 provides examples of the voice interaction function showing SD Mei-chan in normal status, sleeping status, and interacting with a forecast panel.

Figure 7. Example of voice interaction with the proposed system. In order: normal status; sleep status; and interacting with a forecast panel.

Voice interaction based on Android OS API

Android OS provides some APIs such as calendar, email management, and networking. Our proposed system can use these APIs by implementing PluginListener for Java.

For example, users can run the calendar service by using the calendar API. Were the user to say “please tell me my schedule for today,” the agent will answer “you have two appointments: a conference at 10 a.m., and a planning

meeting at 1 p.m.” by referring to schedule data in the Android OS.

Voice interaction based on network services

The proposed system can retrieve weather information by connecting with forecasting web services. Users can find weather information using their voice and an image, as shown in Figure 7 (right). For example, when the user asks “how is the weather today?,” the virtual agent replies “today’s weather is sunny.” Similarly, users can ask their horoscopes and get directions.

Idle function

Because the speech-recognition process is running continuously, unintended voices and agent-performed actions from falsely recognized noises might pose a problem.

Therefore, the proposed system includes an idle as well as an active mode for preventing false recognition. In active mode, the virtual agent answers all recognized user voices. In idle mode, the virtual agent does not answer any user voices not prefaced with “hello.” In idle mode, when a user says “hello,” the state changes to active mode replacing the point of view, as shown by “normal status” in Figure 7 (left and center).

EXPERIMENTS

We evaluated the delay in voice interaction, power consumption, and usability of the proposed system by comparing the cloud and streaming methods with our proposed method. Apple’s Siri served as the representative cloud model, and Mobile Mei-chan stood as the streaming model. In Siri and Mobile Mei-chan, we used iPhone 4S. In proposed system, we used Galaxy S3. We used b-mobile²’s 3G network for network communications. Table 3 shows the applications and smartphones used in the experiment. Both of the iPhone 4S and Galaxy S3 were flagship smartphones in the summer 2012.

	Application	Smartphone
Cloud method	Apple’s Siri	iPhone 4S
Streaming method	Mobile Mei-chan	iPhone 4S
Stand-alone method	Proposed system	Galaxy S3

Table 3. Applications and smartphones for comparative analysis

Evaluation of the delay of voice interaction

First, we evaluated the delay in the voice interaction of our proposed system by measuring the response time in order to verify **Problem 1** (described in the INTRODUCTION above). In our experiment, “response time” means the

² <http://www.bmobile.ne.jp/english/>

Figure 8. Results of the usability questionnaire for each system

elapsed time from the terminus of the user’s speech to the time when we first hear the synthesized voice—as shown in Figure 1. The experimental conditions were as follows: the speech text was “What is the weather like today?” The target language was Japanese. We recorded the voice with a microphone and measured the response time by measuring the elapsed silence. We measured the response time five times for each system.

Table 4 shows the results, listing response times for each method. The response time for the proposed method (0.82 s) is faster than both the streaming method (1.1 s) and cloud method (4.1 s). The reason for why the proposed method is faster than the streaming method is that our method is not susceptible to network delays. These results suggest that the proposed method is the best approach for mobile voice interaction systems where delay is concerned.

There is some concern that, with the proposed system, the response time will be longer when the size of the language model is larger. Server-side recognition methods need not share this concern. However, we believe that this problem is easily overcome given that the processing speed of smartphones has been increasing at a rapid pace in recent years.

	Response time
Cloud method	4.1 s
Streaming method	1.1 s
Stand-alone method	0.82 s

Table 4. Response time for each voice interaction method.

Evaluation of power consumption

Next, we tested the power consumption rate for the proposed system in order to address **Problem 3** (described in the INTRODUCTION). In general, power consumption is an important factor for smartphone applications. Because a voice interaction system needs to monitor audio input continuously in order to process speech recognition, we needed to verify that the power required for our proposed system is not excessive.

The experimental conditions were as follows: The target smartphone was a Sony Xperia TX with its brightness display at 50%. We disabled the wireless function—both Wi-Fi and 3G networks. We measured the remaining battery power after keeping the terminal running at 1 h from full battery. We compared our prototype system with the system that displays only a static image.

Table 5 shows the experimental results. The remaining battery power for the proposed system (58%) is less than the system displaying only static image (93%). These results suggest that the proposed system can work continuously for 143 min. Whereas one might be concerned that the power required for the proposed system is considerably high for an average smartphone, we think that the power consumption is sufficient to operate in a usual manner because users will not be running the proposed system at all times. Because other applications, such as a web browser or an email application is active more often, the overall power consumption for the proposed system is relatively nominal. Nevertheless, we will study the power consumption issue in future work.

	Remaining battery level (60 min.)	Estimated time
Proposed System	58%	143 min.
Wallpaper	93%	857 min.

Table 5. Evaluation of power consumption. Wallpaper means a static image of background of Android OS.

Evaluation of the usability

Finally, we evaluated the usability of the proposed system. Experimental conditions were as follows: 16 university students participated in the experiment as target users. They were asked to communicate with the voice agents using Siri, Mobile Mei-chan, and the proposed system. We conducted a five-stage questionnaire to obtain our results, asking the students to judge the following:

1. **Response** time of voice interaction
2. The quality of **speech synthesis**

3. The quality of **speech recognition**
4. Did the virtual agent seem **real**?
5. Is the virtual agent **charming**?
6. Do you feel the system is **natural**?
7. How is the **graphic** quality of the virtual agent?
8. Do you feel the **virtual-agent** displayed is desirable?

Figure 8 shows the results of the questionnaire. In **response**, the proposed system (4.7 point) scored better than not only Siri (2.7 point) but also mobile Mei-chan (3.8 point). Although the difference in response time between mobile Mei-chan and proposed system is relatively small—as shown in Table 3—the proposed system is better than we expected. The **reality** (4.7 point) and **charm** (4.6 point) of the proposed system is also better than other systems tested. The **speech recognition** (3.6 point) of the proposed system is worse than that of Siri (4.8 point). We found that the high scores for both **response** and **graphic** quality affected the **reality** and **charm** in spite of the poor **speech recognition** scores. Moreover, the results for **virtual-agent** suggest that the 3D-CG virtual agent is receptive for voice interaction systems. On the other hand, in terms of whether the system is **natural**, our results show the difference between methods is relatively minor. Although the **speech recognition** of the proposed system is worse than that of Siri, we will improve it by adopting higher accuracy models for speech recognition when the processing speeds of smartphones are improved in the future.

We also obtained supplementary comments for our proposed system. Typical positive comments included: virtual agent is cute and natural; CG rendering is beautiful; synthesized voice is clear and pleasant. Typical negative comments suggested: there are fewer responses available than with Siri; it is shameful to always have women agents on smartphones.

RELATED WORK

We have developed other MMDAgent-based voice interaction systems, such as a voice interactive digital signage for campus guide [15] and mobile Mei-chan.

The voice interactive digital signage for campus guide is implemented in a digital signage placed at the main gates of Nagoya Institute of Technology. This system enables users to talk with a female CG virtual agent. Both students and staff can post their event information with synthesized voices by using a web browser, and the virtual agents can then advertise the posted events with voice, gesture, and images.

Mobile Mei-chan is a voice interaction system for mobiles based on video communication systems such as Skype or Google Hangout. With this system, MMDAgent is connected with Skype using Skype API. Users can converse with CG virtual agents using Skype-installed terminals,

such as smartphones and PCs. Although both this system and proposed system can be worked in smartphones, we confirm that proposed system is better than Mobile Mei-chan in EXPERIMENTAL RESULT section.

Talkman [16] is an early voice interaction system based on a 3D-CG virtual agent. The purpose of Talkman is to develop a virtual agent with natural human facial features and expressions based on speech synthesis and speech recognition technology.

There are other agent-based voice interaction systems such as *Takemaru-kun* and *Kita-chan* [17]. These system can be worked as an agent robot. Furthermore, there are voice interaction systems for public transit such as Let's Go bus system [18]. Using these systems, users can query bus information by speech using their phones.

Galatea toolkit [19] is a voice interaction toolkit along with MMDAgent used in this research. Galatea toolkit can develop a spoken dialogue system with a life-like animated agent. Galatea toolkit cannot be worked in smartphones differently from proposed system.

CONCLUSION

This paper proposed a voice interaction system with a 3D-CG virtual agent for stand-alone smartphones. The proposed system enables users to talk with a 3D-CG virtual agent more naturally than with existing systems because the delay in voice interaction is relatively short. Moreover, the proposed system can be fully customized by not only dialogue scripts but also Java-based plugins using the smartphone's APIs such as email management, calendar, and networking. Since the system also can run in live-wallpaper mode for Android smartphones, users can use this system easily without running any applications explicitly.

The response time, power consumption, and usability of proposed system is better than that of existing voice interaction systems such as Siri and mobile Mei-chan. We found that the response time and graphics quality affect the reality and charm of the virtual agents.

Because we have made a subset of the proposed system available as open-source software, it can be easily downloaded from the Web. A demonstration video of the proposed system is also released here³. In the future, we will develop several applications based on this proposed system. Since proposed system can be easily customized based on scripts and plugins, we expect that this system will contribute to studies of human-agent interaction using smartphones.

³ <https://www.youtube.com/watch?v=eR7aUh9RBio>

ACKNOWLEDGEMENTS

This research was partly funded by Core Research for Evolutionary Science and Technology (CREST) from Japan Science and Technology Agency (JST).

REFERENCES

1. Apple Inc. Siri, <http://www.apple.com/ios/siri/>
2. Lee, A., Oura, K., and Tokuda, K. MMDAgent - A fully open-source toolkit for voice interaction systems. *In Proc. ICASSP 2013* (2013), 8382-8385.
3. Ward, N., Rivera, A., Ward, K., and Novick, D. Root causes of lost time and user stress in a simple dialog system. *In Proc. INTERSPEECH 2005*, (2005), 10.
4. Miller, R.B. Response time in man-computer conversational transactions. *In Proc. Spring Joint Computer Conference*, AFIPS Press (1968), 267-266.
5. Starner, T. The Challenges of Wearable computing: Part 2, *IEEE Micro*, Vol.21, issue 4, (2001), 54-67.
6. Shiwa, T., Kanda, T., Imai, M., Ishiguro, H., Hagita, N., and Anzai, Y. How quickly should communication robots respond? *Journal of RSJ*, Vol.27, No.1, (2009), 87-95.
7. Uchiya, T., Yamamoto, D., Shibakawa, M., Yoshida, M., Nishimura, R., and Takumi, I. Development of spoken dialogue service based on video call named 'Mobile Mei-chan' (in Japanese). *In Proc. JAWS2012*, (2012), Interaction 1-3.
8. Lee, A., and Kawahara, T. Recent Development of Open-Source Speech Recognition Engine Julius. *In Proc. APSIPA*, (2009), 131-137.
9. Open JTalk, <http://open-jtalk.sourceforge.net/>.
10. MikuMikuDance, http://www.geocities.jp/higuchuu4/index_e.htm.
11. Bullet Physics, <http://bulletphysics.org>.
12. Android NDK, <http://developer.android.com/tools/sdk/ndk/index.html>.
13. OpenSL ES, <http://www.khronos.org/opensles/>.
14. Lee, A., Kawahara, T., Takeda, K., Mimura, M., Yamada, A., Ito, A., Ito, K., and Shikano, K. Continuous Speech Recognition Consortium — an Open Repository for CSR Tools and Models —. *In Proc. LREC*, (2002), 1438-1441.
15. Oura, K., Yamamoto, D., Takumi, I., Lee, A., Tokuda, K. On-Campus, User-Participatable, and Voice-Interactive Digital Signage (in Japanese), *Journal of Japanese Society for Artificial Intelligence*, Vol.28, No.1, (2013), 60-67.
16. Nagao, K., Takeuchi, A. Speech dialogue with facial displays: Multimodal human-computer conversation, *In Proc. ACL-94*, (1994), 102-109.
17. Shikano, K., Tobias, C., Kawanami, H., Nisimura, R., Lee, A. Development and Evaluation of Takemaru-kun Spoken Guidance System and Portability to Kita-chan and Kita-robo Systems (In Japanese). *In Proc. IPSJ SIG Notes*, Vol.2006 (107), (2006), 33-38.
18. Raux, A., Bohus, D., Langner, B., Black, A.W. and Eskenazi, M. Doing Research on a Deployed Spoken Dialogue System: One Year of Let's Go! Experience. *In Proc. INTERSPEECH 2006*, (2006), 65-68.
19. Kawamoto, S., Shimodaira, H., Nitta, T., Nishimoto, T., Nakamura, S., Itou, K., Morishima, S., Yotsukura, T., Kai, A., Lee, A., Yamashita, Y., Kobayashi, T., Tokuda, K., Hirose, K., Minematsu, N., Yamada, A., Den, Y., Utsuro, T., Sagayama, S. Open-source software for developing anthropomorphic spoken dialog agent. *In Proc. PRICAI-02, International Workshop on Lifelike Animated Agents*, (2002), 64-69.