Speech Interactive Agent on Car Navigation System Using Embedded ASR/DSR/TTS

Heungkyu Lee* · Ohil Kwon · Hanseok Ko*****

ABSTRACT

This paper presents an efficient speech interactive agent rendering smooth car navigation and Telematics services, by employing embedded automatic speech recognition (ASR), distributed speech recognition (DSR) and text-to-speech (TTS) modules, all while enabling safe driving. A speech interactive agent is essentially a conversational tool providing command and control functions to drivers such as enabling navigation task, audio/video manipulation, and E-commerce services through natural voice/response interactions between user and interface. While the benefits of automatic speech recognition and speech synthesizer have become well known, involved hardware resources are often limited and internal communication protocols are complex to achieve real time responses. As a result, performance degradation always exists in the embedded H/W system. To implement the speech interactive agent to accommodate the demands of user commands in real time, we propose to optimize the hardware dependent architectural codes for speed-up. In particular, we propose to provide a composite solution through memory reconfiguration and efficient arithmetic operation conversion, as well as invoking an effective out-of-vocabulary rejection algorithm, all made suitable for system operation under limited resources.

Keywords: speech interactive agent, embedded ASR, embedded TTS.

1. Introduction

As the quality of automatic speech recognition (ASR) and text-to-speech (TTS) is steadily improving, a variety of multimedia application services using embedded ASR (eASR), distributed speech recognition (DSR) and embedded TTS (eTTS) [1] are being introduced for commercial use. In particular, since the demand of Telematics services is surging, the speech interface to interact with human users has become an essential means of the multimodal interface. Tasked as a Telematics client service interface, the eASR, DSR and eTTS combined

* Dept. of Visual Information Processing, Korea University, MediaZen Corporation, ASR team.
** Hyundai Autonet, CT(Core Technology) team.
*** Dept. of Electronics and Computer Engineering, Korea University, Coresponding author: Hanseok Ko (hsko@korea.ac.kr)

as a stand-alone unit provides an easy manipulation interface for command and controlling a car navigation system while the driver can pay attention to safe driving.

Human-to-computer interfacing can be made more accessible and convenient if a conversational agent is applied [2][3][4]. Conversation is one of the most important interactions that facilitate dynamic knowledge interaction. People can have a conversation with a conversational agent that can talk with people by using the eASR and eTTS as a combined unit. In this paper, we propose a speech interaction agent (SIA) as a conversational agent which achieves command and control tasks while interacting with the human driver according to the given scenarios on the car navigation system. The conversational agent is classified into a "task-oriented" agent and an "interaction-oriented" agent. The role of the task-oriented agent is to realize a decision-making function while assuming that the agent understands its environments. The role of interaction-oriented agent is to enable understanding of the environments. The proposed agent integrates task-and-interaction-oriented functions. To do this, the SIA also has communication protocols that are essential to efficient coordination among agents. According to the given communication protocols, the SIA sends the command to the specific application program and performs the user's requests after the agent recognizes the words using the eASR.

The proposed system has to be executed on an embedded car navigation system that has several external buttons to operate. Thus, the SIA not only should cope with the multiple inputs received from the users and applications at any time, but it should also provide priority control and scheduling. The embedded hardware system has lower memory, CPU and performance, compared to its server hardware counterpart. To overcome the resource limitation, we implement and optimize the agent by using algorithmic and architectural approaches.

The content of this paper is as follows. The design concept of SIA using a combination of eASR, DSR, and eTTS is presented in Section 2. The implementation issues for low power optimization are discussed in Section 3. In Section 4, we implement the design reflecting the implementation issues and evaluate the proposed SIA. Finally, in Section 5, we discuss results and provide conclusive remarks including future research issues.

2. The Design of Speech Interactive Agent

2.1 The whole structure definition

The speech interactive agent (SIA) system structure for embedded car navigation system is shown in Figure 1. It is essentially composed of multimedia application services, middleware and multimodal interfaces. Application services provide car audio/video, car maintenance, safety and security, driving assistance, traffic information and e-commerce. Middleware as a net

broker provides transparent transmission service to the internal/external processes for communicating protocols. Multimodal interfaces include control and presentation media such as touch screen, hands-free, mute button, push-to-talk button and SIA using a combination of eASR, DSR, and eTTS.

A driver can either push the buttons in order to operate the car navigation system or just say the words that the SIA is able to recognize after the SIA utters the guidance information according to the operational scenarios. One issue to be resolved is the user's input event that may be transmitted simultaneously by the SIA and by pushing the button. In this case, the broker-process as middleware solves the problem by passing the event message to the SIA transparently and negotiating the resource usage with the SIA. The broker-process provides the network transmission interfaces such as name service, thread processing, security, concurrency, event control, DTS (Distributed Time Service), and DTP (Distributed Transaction Service) via a wireless network.

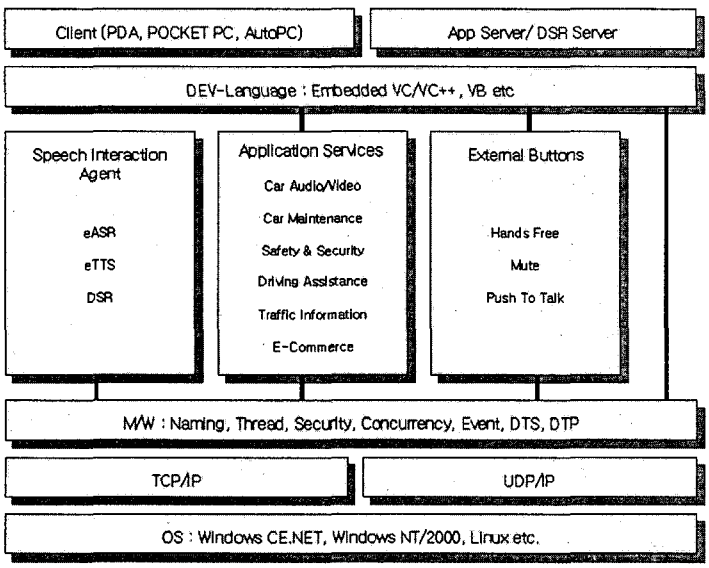


Figure 1. Embedded car navigation system

2.2 Speech Interactive Agent

As shown in Figure 1, speech interaction function provides a convenient means of manipulating the application services without requiring the driver's physical touch on the touch-screen dashboard even while driving a car. Because SIA is a stand-alone process functioning as a conversational agent equipped with eASR, DSR and eTTS modules, people can talk conveniently through the agent, and can engage in the rendered conversation by using speech.

As shown in Figure 2, the agent is composed of feature extractor, speech interaction helper, speech interaction watch-dog, embedded ASR/TTS and the network interface. The speech recognition system is classified into the embedded ASR and distributed speech recognition (DSR) system that is used via the wireless network using a CDMA 2000 terminal. Thus, the feature extractor has the front-end role of passing the mel-cepstral features to eASR or DSR according to the scenarios. The eTTS utters the information when the event is requested by the user and application programs. The "speech interaction helper" provides the helper scenarios to the user when a recognition error occurs or an out-of-vocabulary is encountered. The scope of this function can be extended by including the topic detection service. The "watch-dog" function monitors the service situation and status of the eASR/eTTS in order to cope with the exception handling event which can occur when a user pushes the external buttons during the service interval.

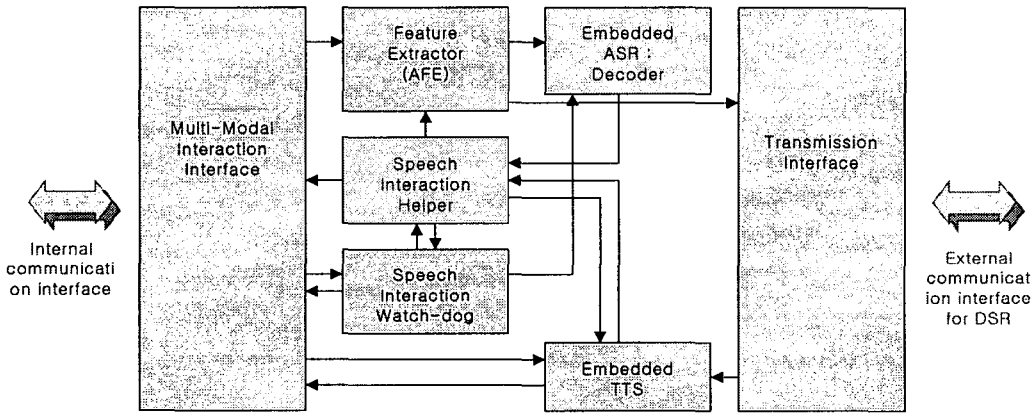


Figure 2. Speech Interactive agent(SIA) block diagram

2.3 Protocols between internal processes using priority control service

Speech interaction transmission flow is shown in Figure 3. The transmission includes handling and relaying the request/response messages to be processed and negotiation message to pipeline the permission for use of the speech resource under negotiated rule-based scenarios [7]. At this point, priority control messaging with the purpose of negotiation is proposed as a method to limit and share the input/output channels of speech because the request messages can be received from the multimodal interfaces simultaneously. For example, while the eTTS utters the traffic information, a user can push the mute button to dial a phone. At this time, the eTTS has to pause or stop. Another case is when a user continues to push the service button on touch-screen while the service guidance is being broadcasted continuously even before the utterance is not completed by the eTTS. In the case of eASR and DSR scenarios, it is the

same. Thus, the "priority control" request message is sent to the SIA every time initially in order to notify the priority of resource request and the user's prompt action.

The data format is delineated in Figure 4. "Application code" represents the index number of applications. "Message code" follows the request-and-response format. "Scenario code" is the index number of the recognizable word list for the eASR and DSR functions. "Priority level" indicates the amount of attention about whether a prompt execution is required or not. Data format is used to transmit either the eTTS utterances received from an application program or the words to be recognized.

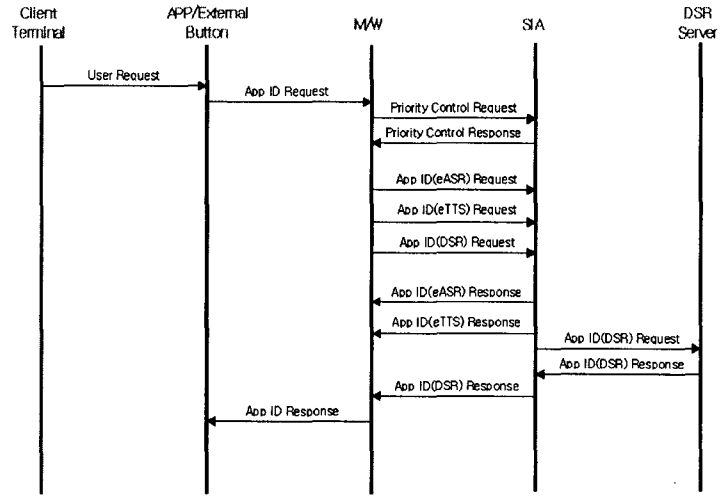


Figure 3. Speech Interactive agent(SIA) protocol flow

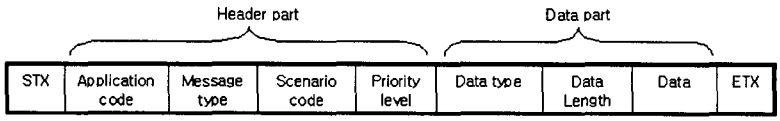


Figure 4. Data format

2.4 Threads and Exception Handling

The SIA creates and handles threads to receive messages or execute the eASR/DSR while the sentences including the information requested from a user are spoken by the eTTS. The thread processing provides simultaneous processing capabilities to execute the prompt request even while another process is running. However, many simultaneous requests received from each multimodal interface may result in an exceptional event that causes non-existing operations. In this case, the SIA needs to handle "exception" events. In order to cope with an exception handling, the agent performs the "stop" or "pause" operation of the eASR/DSR/eTTS

by means of scheduling control. During the exception handling, the "priority control" message has an agent help to decide on the response behavior if an "exception" event is occurred.

3. Implementation Issues of Speech Interactive Agent for Low-Power Optimization

Implementing the SIA on an embedded platform such as the StrongArm simulator requires speed and power optimization because the battery life is very limited. Thus, source code and run-time optimization are required. Relevant previous works in code and run-time optimization for low power can be found in [5][6].

Two categories of source code optimization can be used: architectural and algorithmic. Architectural optimization reduces the power consumption for a particular system. Algorithmic optimization is more general and involves changes in the algorithmic implementation of the source code to run faster and consumes less power.

3.1 Architectural optimization

Profiling of the source code on the StrongArm simulator reveals that over 90% of the time is spent in floating-point emulation because StrongArm has no on-chip floating-point processor. Thus, the floating-point algorithms have to be changed into the form of fixed-point arithmetic. Fixed-point arithmetic uses scaled integers designated by the scaling factor Q*n*, where *n* is the number of bits to the right of the decimal, in order to perform basic arithmetic functions using the existing integer hardware. The basic rules are as follows. Adding and subtracting two numbers in the Q*n* format yield a number in the same Q*n* format. Therefore, this computation requires decimal points that are lined up. Multiplying two numbers in the Q*n*format yields a number in the Q2*n* format. Dividing two variables in the Q*n* format yields a number in the -Q*n* format.

A fixed-point DSP processor usually provides a hardware multiplier so that addition and multiplication can be computed in one CPU cycle. However, there is no hardware for division. Thus, it takes more than 20 cycles to compute division. To avoid this, we have to precompute the inverted data and store it.

3.2 Algorithmic optimization

In a fixed-point prepressor, the usual format is 16-bit or 32-bit integer. Thus, in order to make use of the full 16-bit or 32-bit dynamic range, the algorithm has to carefully normalize its numerical behaviors to within the dynamic range of a 16-bit or 32-bit integer at every stage of

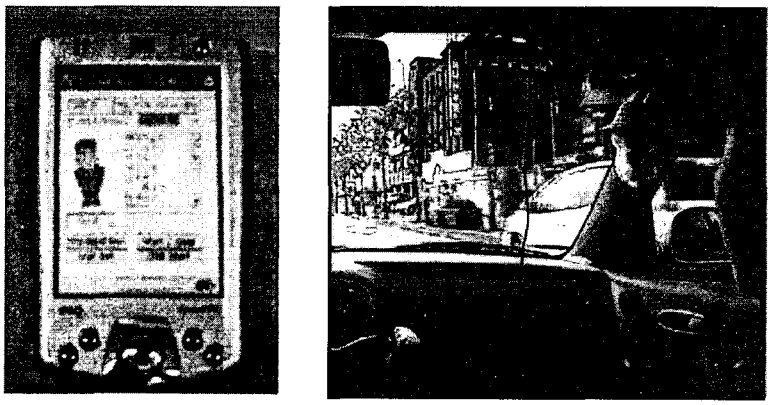
the computation. The design principles are as follows. At first, normalization takes advantage of the full dynamic range. Next, we remember the normalization factors and then compensate them before output by saving step by step.

As usual, we use the look-up table using the fixed Q*n*-format to speed up the algorithms such as pre-emphasis filter, Hamming window, mel-filter bank and so on. The fixed Q*n*-format of the look-up table has to be determined after the simulation for checking dynamic ranges is completed.

The computation time of the DFT requires a lot of time. Because speech is a real-valued signal, an N-point complex FFT can be reduced to an N/2-point real FFT. Also, the computation time can be reduced furthermore when the code is changed into fixed-format algorithm using the look-up table.

4. System Implementation and Evaluation

The SIA and virtual simulator are implemented on a Compac PDA, whose operating system is Windows CE.NET. As shown in Figure 5, to test the SIA, a virtual simulator is implemented. The virtual simulator has only a message sending function and the SIA is a hidden background process having the role of communicating and responding to the request message by eASR/DSR/eTTS. The virtual simulator first sends the request message to perform eASR and then SIA starts to recognize the spoken words. When eASR finishes recognition, eTTS utters the resulting word. The virtual simulator also sends texts about car or traffic information to the agent. In response, SIA utters the text sentences received from the simulator.



(a) The virtual simulator (b) real test on a car

Figure 5. The virtual simulator and real test

The SIA has one eASR thread, one DSR front-end thread and two eTTS threads. In the case of eTTS, one channel for eTTS output is related with car navigation system, and the other is related with application services. The eTTS output related with car navigation system has higher priority than others. Thus, the eTTS output related with car navigation system never stops or pauses. The total number of recognizable words is more than 1,000 words. However, the tree-based recognition approach is applied according to the operational scenarios in the car navigation domain.

An embedded speech recognition engine is developed as a part of car navigation system and optimized in car noises. It is an isolated word recognizer with dynamic vocabularies. A driving test is performed on a car as delineated in Table 1. A total of 40 men and women are tested on a Hyundai EF-Sonata and Samsung SM5 car respectively. The driving test was done at a low speed between 20 and 60 Km/H while the high speed was between 70 and 110 Km/H.

Embedded Text-To-Speech engine is evaluated to check the sound quality as in Tables 2 and 3 for men and women in terms of mean-opinion-score. eTTS 1, 2, 4 and 5 are the various versions of the engines developed for embedded environments. eTTS 1 is 32M in size with a young woman's voice while eTTS 2 is 64M in size also of a woman's voice. eTTS 4 is 32M with a man's voice. eTTS 5 is 64M. eTTS 3 is 32M with a woman's voice developed by a benchmark developer.

Table 1. Test Results on a car

	office	low-speed	high-speed	average	car
off-line	99.69	94.44%	82.10%	-	Avante
man	-	95.4%	96%	95.7%	EF Sonata, SM5
woman	-	89.5%	89.2%	89.3%	EF Sonata, SM5
average	-	92.5%	92.6%	92.5%	EF Sonata, SM5

Table 2. MOS(Mean Opinion Score) test for man

	TTS1		TTS2		TTS3		TTS4		TTS5	
	naturalness	clearness	naturalness	clearness	naturalness	clearness	naturalness	clearness	naturalness	clearness
man1	33	29	32	35	34	34	34	35	36	35
man2	26	31	32	32	36	32	29	32	31	32
man3	24	22	28	31	32	31	28	34	27	32
man4	23	24	22	25	33	30	26	25	32	34
man5	22	23	26	24	32	31	28	27	33	33
man6	24	19	23	27	34	29	27	26	31	30
man7	14	19	23	27	26	31	27	32	32	35
man8	19	21	16	25	29	30	27	27	26	27
man9	24	27	29	32	36	29	31	37	36	37
man10	21	24	29	29	34	38	25	28	32	32
	230	239	260	287	326	315	282	303	316	327
	469	2.93125	547	3.41875	641	4.00625	585	3.65625	643	4.01875

Table 3. MOS test for woman

	TTS1		TTS2		TTS3		TTS4		TTS5	
	naturalness	clearness	naturalness	clearness	naturalness	clearness	naturalness	clearness	naturalness	clearness
woman1	21	23	23	29	37	28	28	30	27	30
woman2	24	26	20	23	26	21	24	27	27	28
woman3	19	23	23	23	28	23	23	27	29	30
woman4	11	16	20	22	19	19	16	17	21	18
woman5	23	23	22	26	25	22	22	24	29	30
woman6	24	21	26	28	24	25	21	24	24	28
woman7	20	21	22	29	28	27	21	27	26	30
woman8	21	24	21	24	28	26	22	28	27	30
woman9	22	28	29	34	31	31	29	33	29	34
woman10	18	19	22	21	28	24	23	24	29	25
	203	224	228	259	274	246	229	261	268	283
	427	2.66875	487	3.04375	520	3.25	490	3.0625	551	3.44375

We optimized the algorithms in terms of algorithmic and architectural views as described in Section 3. As a result, the execution time and code size are optimized as in Table 4 and 5 in the case of the eASR. The eASR having the form of floating point arithmetic is very faster on personal computer, but on PDA, the eASR is very slow. It spends over 20 seconds to finish the speech recognition. So, we changed the codes into the fixed-point form that is hardware dependent. In the case of the eTTS, the execution time and code sizes are also optimized as in Table 6. However, the access time of storage to get the specific tri-phone wave spends a lot of time. It can be resolved using fast accessible memory storage. In Table 3, the language processing part includes natural language processor, a prosody generator and grapheme to phoneme conversion. Speech synthesizer I includes decision maker to search and get the proper tri-phone in the index database. Speech synthesizer II is related with input/output channel access between speech synthesizer database and eTTS process, and speech concatenation.

Table 4. The eASR average speed; average 100 frames

Environment	Average Speed(Milliseconds)		Arithmetic
PC : Pentinum 4, CPU 2GHz, 512M RAM	MFCC	32	Floating-point
	Decoder	93	
Compac PDA : Intel StrongARM 206MHz, 64MB SDRAM, 32MB Flash ROM	MFCC	714	Fixed-point
	Decoder	1462	
	MFCC	659	Fixed-point, optimized version
	Decoder	1296	

Table 5. The eASR memory size

Components	Floating-point	Fixed-point	Fixed-point, optimized version
Front-End (Feature extraction)	17	8.5	8.3
Noise suppression & Feature compensation	6	2.8	2.6
HMM model	1400	1035	774
Total (Size : Kbyte)	1423	1046.3	784.9

Table 6. The eTTS average speed to synthesize the sentences (11 Khz, 40 M DB)

Input Text (Bytes)	Output Sound (Bytes)	Tri-phone number	Response time (milliseconds)			
			Language Processing	Speech Synthesize I	Speech Synthesize II	Total
51	135916	48	142	260	1177	1579
95	255196	92	332	489	2348	3169
111	270756	98	232	558	2225	3015
152	449662	150	391	727	3980	5098
222	531512	196	454	1096	4023	5573

5. Conclusions

In this paper, we presented an implementation of an efficient speech interactive agent rendering smooth car navigation and Telematics services, by employing embedded automatic speech recognition (ASR), distributive speech recognition (DSR) and text-to-speech (TTS) modules. In particular, we proposed to solve the hardware resource limitation problem by providing a composite solution through memory reconfiguration and efficient arithmetic operation conversion, as well as invoking an effective out-of- vocabulary rejection algorithm.

6. Acknowledgement

This work was supported by grant No. A17-11-02 from Korea Institute of Industrial Technology Evaluation & Planning Foundation.

References

[1] Huang, X., Acero, A., Hon, H. 2001. *Spoken Language Processing.* Prentice Hall PTR.

[2] Takata, S., Kawato, S., Mase, K. 2002. "Conversational agent who achieves tasks while interacting with humans based on scenarios." *Robot and Human Interactive Communication Proceedings: 11th IEEE International Workshop*, 25-27 Sept.

[3] Aakay, M., Marsic, I., Medl, A., Guangming Bu. 1998. "A system for medical consultation and education using multimodal human/machine communication." *IEEE Trans-Information Technology in Biomedicine, Vol 2*, Issue: 4, Dec.

[4] Prendinger, H., Ishizuka, M., 2001. "Let's talk! Socially intelligent agents for language conversation training." *IEEE Trans-Systems, Man and Cybernetics, Part A, Vol 31*, Issue: 5, Sept.

[5] Delaney, B., Hans, M., Simunic, T., Aquaviva, A. 2001. "A Low-Power, Fixed-Point Front-End Feature Extraction for a Distributed Speech Recognition System." HP Technical Report, HPL-2001-252.

[6] Yifan Gong, Yu-Hung Kao, 2000. "Implementing a high accuracy speaker-independent continuous speech recognizer on a fixed-point DSP." *ICASSP 2000 Proceedings, Volume: 6*, 5-9 June.

[7] Gerard J. 1991. Holzmann, *Design and Validation of Computer protocols.* Prentice Hall.

received: 2004. 4. 24
accepted: 2004. 6. 15

▲ Heungkyu Lee
Department of Visual Information Processing, Korea University,
MediaZen Corporation
5Ka-1, Anam-dong, Sungbuk-ku, Seoul, 136-701, Korea
Tel: +82-2-953-8003(211) (O), 017-326-9067 (H/P)
Fax: +82-2-923-8830
E-mail: hklee@ispl.korea.ac.kr, hklee@mediazen.co.kr

▲ Ohil Kwon
Hyundai Autonet, Core Technology Team.
San 136-1, Ami-ri, Bubal-eub, Ichon-si, Kyoungki-do, 467-701, Korea
Tel: +82-31-639-7817 (O), 016-352-5150 (H/P)
Fax: +82-31-639-6695
E-mail: koi@haco.co.kr

▲ Hanseok Ko
Dept of Electronics and Computer Engineering, Korea University
5Ka-1, Anam-dong, Sungbuk-ku, Seoul, 136-701, Korea
Tel: +82-2-3290-3239 (O), 011-9001-3239 (H/P)
Fax: +82-2-3291-2450
E-mail: hsko@korea.ac.kr