A HAND-HELD SPEECH-TO-SPEECH TRANSLATION SYSTEM

Bowen Zhou, Yuqing Gao, Jeffrey Sorensen, Daniel Déchelotte and Michael Picheny

IBM T. J. Watson Research Center
Yorktown Heights, New York 10598
{zhou, yuqing, sorenj, ddechel, picheny}@us.ibm.com

ABSTRACT

Recent advances in the processing capabilities of Personal Digital Assistants (PDAs) have enabled the creation of an end-to-end speech translation system which is completely hosted on an off-the-shelf PDA. Specifically, the system presented includes an HMM-based large vocabulary continuous speech recognizer using statistical n-grams, a translation system, and, a speech synthesis system that outputs speech in the target language. This paper describes the detailed architecture of our system and the functionality of its main components, focusing on the optimization efforts employed to achieve real time results on such a limited platform. In a preliminary evaluation, the PDA system is found to achieve comparable performance with the corresponding system developed for desktop computers.

1. INTRODUCTION

With increasing international travel and commerce, automatic speech-to-speech translation systems have attracted a significant interest from both theoretical and developmental communities. In recent years, a number of projects, including C-STAR [1], VERBMOBIL [2, 3], BABYLON [4], NESPOLE! [5] and EU-TRANS [6], have been devoted to addressing the issues of automatic speech translation. There have been numerous efforts to achieve reliable and satisfactory translations between languages using powerful resources such as desktop servers and laptop computers. However, it is noted that such devices are not compact. Thus, such systems are not convenient for mobile applications. This limits the usefulness of such translation technologies. Many realistic circumstances, such as business trips and tourism, can only be effectively aided by a truly mobile device.

Independently, the development of increasingly powerful mobile computing platforms is reaching levels that are comparable to the power of desktop systems of only a few years ago. In order to bridge the gap between contemporary translation systems and the current mobile computing platforms, we have employed a number of optimizations, significantly enhancing the accessibility of our automatic speech translation technology.

Automatic speech-to-speech translation is a highly complex task. A large amount of computation is involved to achieve reliable transformation performance. Resources are not just computation limited, but the memory and storage requirements, and the audio input and output requirements all tax current systems to their limits. When the resource demand exceeds the computational capability of available state-of-the-art hand-held devices, a common technique for mobile speech-to-speech translation system is to use a client-server approach [3, 7]. Here, the hand-held device (either a mobile phone or PDA) is treated simply as a system client, and the speech input is compressed and transmitted from this client to a back-end server that is much more powerful, either over a wireless telephone network or a wireless LAN connection such as Wi-Fi (IEEE 802.11b). The entire end-to-end speech translation task is conducted at the server. Finally, the spoken utterance in the target language is sent back to the hand-held device, thus providing the user audio output on location.

There are several obvious disadvantages of the client-server based approach. First, the service area is limited to locations where wireless connection are available. Second, large vocabulary speech recognition over conventional telephone channels, especially unreliable wireless channels, will degrade the quality of the translation. Third, this approach limits the flexibility of the user's control over the overall translation system, making highly customized applications much more difficult to design and deploy.

To address above issues, a fully functional translation system needs to be entirely hosted on a hand-held device. Recently, Waibel et. al. presented a speech-to-speech translation system running on a Windows-based PDA for a limited application domain [8], demonstrating the feasibility of such ideas. In this paper, we present a complete speech-to-speech translation system that is deployed on a standard PDA on an embedded Linux operating system that translates spoken English to Mandarin Chinese. The translation

This work is founded by the DARPA BABYLON project.

system is completely hosted on an off-the-shelf PDA, but employs the same architecture as its desktop counterpart, the MASTOR [9, 10] system.

Our PDA-based system achieves comparable translation performance and speed to that found in MASTOR system. Numerous optimizations were employed to improve translation speed and to reduce resource demands. However, the system still maintains a large vocabulary continuous speech recognizer that operates in real time, or near-real time. The typical response time for an end-to-end translation is under 5 seconds.

This paper is organized as follows: Sec. 2 describes the system architecture, as well as the hardware and software specifications. Sec. 3 describes major functional components, with particular focus on optimizations. Preliminary evaluations of some major components are also provided in Sec. 3. Sec. 4 includes a discussion and future work, followed by a summary conclusion in Sec. 5.

2. SYSTEM OVERVIEW

2.1. System Architecture

Our previous work in speech-to-speech translation systems [9, 10] included the development of a complete system based on a desktop computer. We have maintained the same basic architecture of this earlier system while adapting it to the capabilities of a handheld computer.

Figure 1 shows the architecture of our speech translation system. The input speech is recognized using an automatic speech recognizer and then parsed by a statistical natural language understanding (NLU) module. An information extraction component is responsible for analyzing the semantic tree obtained from the NLU. This component is responsible for representing the (recognized) spoken sentence information in a language independent "interlingua" representation. This is combined with the canonical representation of "named entities" such as numbers and other attributes detected by our semantic model. The resulting representations are sent to an natural language generation (NLG) engine to render in the target language. The two types of information are translated using distinct models, with the the specific attributes of items, such as times and dates, using conventional techniques familiar to the machine translation community. The interlingua translation, however, uses statistical techniques and can perform considerable surface changes when required for the target language. Finally, when a textual representation of the utterance in the target language is complete, a text-to-speech synthesizer is used to produce spoken output.

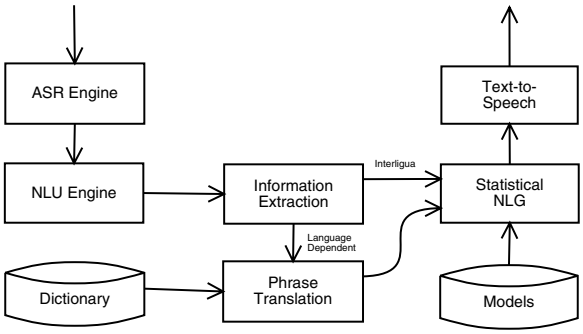

Fig. 1. System Architecture for a Speech-to-Speech Translation System on PDA

2.2. Hardware and Software Specification

To demonstrate the feasibility of building a system with our speech translation architecture on a standard PDA, we select the target hardware platform as HP (né Compaq) iPaq PDA model H3800, a popular and widely available device. The processor shipped with this product is Intel's StrongARM CPU based on ARM architecture with a frequency at 206 MHz. The system has 64 MB SDRAM and 32 MB flash ROM memory. The available memory for running our end-to-end speech translation system is about 35 MB. For audio I/O, iPaq products provide a push-and-talk button and an integrated microphone for speech input, as well as an integrated speaker for audio output. The PDA is further expanded during development using either a IBM CompactFlash Microdrive, or a MultiMediaCard to provide additional storage, mainly for the n-gram language model and dictionaries.

The original iPaq is shipped with Microsoft's Pocket PC 2002 operating system, In our system, the operating system is replaced with Familiar [11], a full featured Linux distribution for the iPaq series, based on the embedded Linux kernel. To sufficiently make use of existing system components found in the MASTOR desktop system, all of the major components are ported to PDA platform and a number of portability issues have to be addressed. In addition, numerous optimizations are required to make the applications fitting in the low footprint of hand-held devices. To construct applications for such an ARM-Linux based handheld device, a cross-compiler tool chain is employed using a Linux based x86 host.

2.3. System Interface

Figure 2 shows a screenshot of this system running on iPaq Linux. The users are allowed to configure the system working in two modes. One is the automatic mode, in which the user pushes the push-and-talk button to speak in his lan-

Fig. 2. Screenshot of IBM Speech-to-Speech Translation System Running on Linux-based iPaq

guage. When he releases the button, the system will automatically present the recognized spoken input and synthesize the translated utterance in target language. Alternatively, an interactive mode allows the user to correct the speech recognition errors if any, repeat the utterance, or make other changes before translation and synthesis is performed.

3. SYSTEM COMPONENTS

This section describes some major modules embedded with this hand-held speech translation system. To illustrate the performance of each module, we evaluate these components using the data-set in the domain of force protection and medical triage, which is designed and collected for DARPA's BABYLON project.

3.1. Large Vocabulary Continuous Speech Recognizer on PDA

The recognition module used in our mobile speech translation system is a HMM-based large vocabulary continuous speech recognition engine using statistical n-grams. Unlike most grammar based embedded speech recognition systems, our system has the vocabulary coverage and flexibility to switch new application domains typically only found on desktop or server based systems. To accomplish this, IBM's large vocabulary speech recognition system, as featured in the popular ViaVoice products, was ported to the ARM processor architecture.

The speech recognition system takes as input speech sampled at a rate of 22 KHz. The acoustic front-end uses a 24-dimensional cepstral feature vector extracted every 10 ms. Blocks of 150 ms features are transformed using linear discriminant analysis into a 40-dimensional feature vector.

The English speech recognition system uses an alphabet of 52 phones. Each phone is modeled with a 3-state left-to-right hidden Markov model. This system has approximately $3,500$ context-dependent states modeled using more than $40,000$ Gaussian distributions. The context-dependent states are generated using a decision-tree classifier.

On the porting this large scale system to ARM architecture, it is first noted that the StrongARM platform, unlike the Intel x86 series, has no integrated floating point (FP) hardware. It depends entirely on software that emulates the floating point co-processor. Despite much of the IBM recognizer being developed to use mostly integer computations, our initial profiling experiments showed that substantial amounts time were consumed by floating point calculations. Therefore, significant efforts were made to completely integerize the majority of the signal processing front-end and search components of this system. This includes a fixed point math implementation of the following major recognizer components: the Mel-cepstrum feature extraction, the Gaussian likelihood computation of the context dependent phone models, as well as the procedures of fast match and detailed match during the decoding process. Floating point calculations are still used in some portions of the recognition engine, but all of the the computationally expensive portions of the code have been fully integerized.

Although not a major portion of the compute time, some gains were achieved by making use of the Intel's Integrated Performance Primitives (IPP) library which is optimized for the ARM architecture, for the fast Fourier transform calculations used prior to Mel-band energy calculation.

Fixed point code necessarily involves scaling to effectively use the available dynamic range of the processor word size. In most cases, the amount of shifting was empirically determined using reference recordings. Thus the embedded speech recognition system will not have the same flexibility in dealing with signals of unusually high or unusually low signal gain.

The remaining floating point calculations are handled by the use of software floating point routines. By default, on this architecture, the ARM FP instructions fault with an undefined instruction trap. This is caught and handled by the ARM Linux kernel. The use of traps has high overhead and is typically slow. Alternatively, software implementa-

Table 1. Speaker independent recognition performance in the general dictation domain

Speaker	Word Error Rate
ALD	11.09%
CFD	18.60%
JMD	14.40%
RGB	15.38%
Average	14.87%

Table 2. Speaker independent recognition performance in the domain of force protection and medical triage

Speaker	Word Error Rate
TIM	14.45%
ALL	11.51%
Average	12.98%

tions are available to emulate floating point calculations as library calls. In this approach, a "soft-fp" enabled compiler does not generate the FP instructions, instead library calls are generated for each floating point computation. Typically, a "soft-fp" compiler generates much faster binaries than those applications with FP instructions handled by kernel. To enable the "soft-fp" support for our ported system, a customized tool chain is built in this work, where the libraries in this tool chain are completely recompiled with the "soft float" option turned on.

A statistical tri-gram language model with a large vocabulary of $22,430$ words was built for this PDA-based continuous speech recognizer. The language model is highly compressed yet still requires 12 MB of storage. Currently, the continuous speech recognition system functions in real time, but a significant portion of the time is dedicated to trigram language model lookups. This is due to the need to access this large and sparse data structure which is stored in virtual memory. Reducing the storage requirements and the access time for the language model is ongoing work.

The large vocabulary speech recognizer for hand-held devices was evaluated on the iPaq H3800 using multiple test sets. Table 1 depicts the speaker independent recognition performance using the test data from the general dictation domain. The test set includes 4 male speakers, each of them provides 61 utterances. Table 2 shows the initial speaker independent evaluation performance in the domain of force protection and medical triage. In this experiment, 2 male native speakers were dictating to the iPaq from a script that contains 150 utterances. These results indicate that despite the optimizations for fixed point calculations and the reduced vocabulary language model, the recognition performance is comparable to that seen on desktop systems.

The recognizer as ported to the iPaq platform includes all of the support of the custom acoustic enrollments and domain specific language models as featured in the ViaVoice recognizer. Use of these enhancements is part of our future work. Currently, support for enrollment on the iPaq has not been ported, so another desktop computer must be used to complete the model building for a specific speaker. However, the recent addition of non-interactive training should make implementation of speaker dependent adaptation much more straightforward.

3.2. Statistical Natural Language Understanding

The natural language understanding (NLU) module embedded within this hand-held speech translation system is based on the statistical parser employed in our telephony natural language dialogue systems [12]. This component uses statistical decision-tree models to determine the meaning and structure of the input spoken sentence. This is done by assigning a hierarchical tree structure to the recognized sentences as predicted by the statistical model. Trained from hand-annotated spoken sentences in the source language, the NLU model does not rely on any handcrafted grammars or rules. Unlike the system in [10], the two separate components, the "classer" and the "parser" have been combined into a single module for this task.

While the NLU module is not a significant computational bottleneck, it is important to improve the runtime speed of this module to lower the overall response time of the system. An effort was made to reduce the runtime memory requirements and to improve the parsing speed. Primarily, this involved reorganizing the statistical decision-tree models. In addition, the "soft-float" emulation libraries, as described in Sec. 3.1, were used as well to improve the speed of floating point computations. Due to the dynamic range of the probabilities in the NLU models, it is unlikely that an integerized version of this search would work as well. Presently, the parsing speed is about 1.5 second per utterance for our current application domain.

For application in the domain of force protection and medical triage, a set of spoken sentences with $10,004$ utterances is used to evaluate the NLU performance. Approximately, 85% of the collection is used as the training data, 10% is used as the evaluation test data to tune model parameters, and the remaining 5% is used as the test data. Table 3 summarizes the performance of this module running on PDA.

3.3. Statistic Natural Language Generation

High level semantic translation is performed by a natural language generation system using the semantic representation obtained from NLU module. The statistical natural language generation module is based on the algorithms intro-

Table 3. Performance of the decision-tree based NLU in the domain of force protection and medical triage

Data	Training	Evaluation Test	Test
Size	8445	1023	536
Accuracy	95.31%	75.07%	70.70%

duced in our previous study [9], and has merged the methods discussed in [13]. The language generation module uses maximum likelihood prediction based on maximum entropy modeling [14, 15], where the models parameters are estimated from a training corpus. No manually designed grammars or knowledge bases are used in the building of the NLG system.

In this implementation, several modules showed in Figure 1, i.e., the information extraction, phrase translation, and statistical NLG, are integrated as a unified component to accomplish the task of generation based on semantic representations. Combined with the optimized NLU module, these components compose the translation function of this speech translation system. On porting this component to ARM platform, this NLG module is reimplemented to fit with low computational resources available on PDA. This includes a more efficient implementation of search procedures, as well as significantly reduced I/O routines.

After optimizations, the NLG component is able to run much faster, which enables the entire translation procedure of this system take less than real time (see Sec. 4), while maintains the same accuracy level of the desktop based system (i.e., with a concept error rate of 13.5% as reported in [13]).

3.4. Text-to-Speech Synthesizer

The translated utterance generated from NLG module is synthesized by a text-to-speech engine. Considering the limited resources available for a mobile device, the current TTS system is based on IBM's formant TTS technology [16].

The formant based TTS system can synthesize an unlimited number of voices, and allows flexible customizations by modifying vocal characteristics such as gender, pitch, volume and speed. It also has the advantage of supporting unlimited vocabularies. In effect, the TTS software can pronounce any text that it is given. More importantly, format-based TTS system has a small footprint and requires less memory (about 3 MB for each language), which makes it appropriate to be deployed in embedded applications.

To port the IBM TTS engine from x86 to StrongARM Linux, the computations are heavily integerized. In addition, a number of portability issues, such as pointer alignment and memory overlays, as well as other performance optimizations, are addressed in this work.

For this PDA-based speech translation system, male, female and child voices are provided in its TTS component. Therefore, the output voice can be easily switched according to the needs of user.

4. DISCUSSION AND FUTURE WORK

The speech-to-speech translation system presented in this paper is completely hosted on an off-the-shelf PDA. This system includes an HMM-based large vocabulary continuous speech recognizer using n-grams. Moreover, the entire translation is performed in near-real time. Specifically, the entire translation procedure in this system, from the end of speaking up until the start of playback of the synthesized speech in target language is typically between 1 and 4 seconds, depending on the complexity of the spoken input.

At the time of writing this article, we are also working to provide translation from Chinese to English on the same handheld device. Future work will be focused on improving the overall end-to-end translation accuracy. This includes adapting an acoustic model for iPaq's integrated microphone, creating a more compact language model, as well as making improvements in the translation models themselves.

5. SUMMARY

In this paper, we present a speech-to-speech translation system that is deployed on a standard PDA for translating from spoken English to Mandarin Chinese. This presented end-to-end speech translation system is completely hosted on an off-the-shelf PDA running embedded Linux. This PDA-based system maintains comparable translation accuracy and speed found in its desktop counterpart. This mobile translation system distinctively includes an HMM-based large vocabulary continuous speech recognizer using n-grams. Furthermore, the entire translation procedure is able to be performed in near-real time.

6. ACKNOWLEDGEMENTS

The authors thank Jerry Quinn, Mike Monkowski, and other IBM colleagues at Speech-to-speech translation group, ViaVoice engine group and TTS group for their help with this work.

7. REFERENCES

[1] "http://www.c-star.org," website.

[2] Federal Ministry of Education, Science, and Technology, "http://verbmobil.dfki.de," website.

[3] W. Wahlster, "Mobile speech-to-speech translation of spontaneous dialog: an overview of the final verbmobil system," *Verbmobil: Foundations of speech-to-speech translation*, pp. 3–21, 2000.

[4] "http://www.darpa.mil/ipto/research/babylon.html," website.

[5] "http://nespole!.itc.it," website.

[6] "http://prhlt.iti.es/projectes/eutrans/eutrans.html," website.

[7] K. Yamabana et al., "A speech translation system with mobile wireless client," in *ACL 2003*, Sapporo, Japan, July 2003.

[8] A. Waibel et al., "Speechalator: two-way speech-to-speech translation in your hand," in *HLT-NAACL 2003*, Edmonton, Canada, May-June 2003.

[9] B. Zhou et al., "Statistical natural language generation for speech-to-speech machine translation systems," in *ICSLP-2002*, Denver, CO, Spetember 2002.

[10] Y. Gao et al., "Mars: A statistical semantic parsing and generation based multilingual automatic translation system," *Machine Translation*, 2003, To appear.

[11] "http://familiar.handhelds.org," website.

[12] K. Davies et al., "The ibm conversational telephony system for financial applications," in *Eurospeech*, 1999, vol. 1, pp. 275–278.

[13] L. Gu et al., "Improving statistical natural concept generation in interlingua-based speech-to-speech translation," in *Eurospeech-2003*, Spetember 2003.

[14] A. Ratnaparkhi, "Trainable methods for surface natural language generation," in *Proc. of the 1st Meeting of the North American Chapter of the Association of Computational Liguistics*, Seattle, WA, 2000, pp. 194–201.

[15] A. Berger et al., "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39–71, 1996.

[16] "http://www.wizzardsoftware.com/voice/voicetools /dictationfortts.htm," website.