Real-time Speaker Adapted Speech to Speech Translation System in Mobile Environment

Yong Guan[1,2], Lin Zheng[2] ,Jilei Tian[2]
School of Information and Communication Engineering, Beijing University of Post and Telecommunication, Beijing, China
Nokia Research Center, Beijing, China
{Ext-yong.guan, ext-zheng.lin, Jilei.tian}@nokia.com

Abstract—**In this paper, a real-time speech to speech translation (S2ST) system in mobile environment is designed and implemented as a client-server architecture. Particularly, we apply cross lingual speaker adaptation to adapt synthesized speech to enrolling speaker to ensure personalization. This real-time S2ST system provides streaming way, multi-threading and speaker adapted speech to speech translation for mobile user. It makes it available that mobile users get personalized real-time S2ST service through 3G/WIFI network in mobile environment.**

Keywords - speech to speech translation; speaker adaptation; cross lingual speaker adaptation; AMR coding; mobile application

I. INTRODUCTION

Generally speaking, there are two kinds of Speech to Speech Translation (S2ST) systems, of which, one is embedded system or application into a hand-held device (such as a mobile phone)[1], and another is client-server system which makes the S2S translation services available to those hand-held devices connected via a 3G mobile phone networks[2]. The client-server system makes it available that we put more resources and more powerful application in the server end and a slim application in mobile client. And also, it is possible that we make S2ST system work like a service in the client-server mode. Moreover, nowadays, the 3G and WIFI network make transmission delay acceptable for client-server mode.

In this paper, we develop the real-time S2ST system in a client-server mode, in which the enabling components, including ASR (Automatic Speech Recognition), MT (Machine Translation) and TTS (Text to Speech), are implemented on the server. The client end software is mainly responsible for the user interface and interaction as well as some audio pre-processing. The real-time S2ST system includes ASR, MT and TTS online modules and we train the speaker adaptation for ASR and cross-lingual speaker adaptation for TTS as an off-line task to ensure real-time performance. With the main computation being performed by the server, the client will be a stand-alone application running on the mobile device. An utterance from one speaker is recorded from the mobile client and then transmitted to the server. After generating translated and synthesized voice in the server, synthesized speech is transmitted to the mobile device of another speaker.

Personalization is necessary for natural interaction and to make the device less obtrusive, in what is essentially a human-human interaction. With cross lingual speaker adaptation, a user's spoken input in one language is used to produce spoken output in another language, while continuing to sound like the user's voice. In this S2ST system, we use the streaming mechanism to make it work with real time, use the multi-threading mechanism to make it work as a service, and use the speaker adapted algorithm to make it personalized so that we get a real-time speaker adapted S2ST translation service system running in client-server mode and mobile environment.

The remainder of this paper is organized as follows: first, we briefly overview the real time S2ST system, including architecture design, network communication protocol and data handling. Next, a detailed description is given for the server engine and experimental analysis, including the primary components: ASR, MT and TTS. This is followed by the user interaction design in the client end. Finally, we conclude this report and discuss the future directions.

II. OVERVIEW OF THE REAL-TIME S2ST SYSTEM

In order to develop a mobile cross-lingual S2ST system, we need to provide an infrastructure that is able to flexibly integrate the selected components. Basically, we design a framework that can be used to easily and flexibly test and evaluate different components that have been developed for the real-time S2ST system. More precisely, we design an infrastructure based on client-server architecture consisting of clients that can run on the Symbian mobile devices. The server runs Linux OS (Ubuntu system with internet support).

A. Network settings

The real-time S2ST system is intended to run on realistic scenarios that can integrate and evaluate different ASR, MT and TTS algorithms. Once the client software has been installed on smart phones, each client is allocated an IP address by the DHCP server in the network. The client sends/receives IP packets to/from the server running on Linux OS. The client can use 3G or Wi-Fi to connect to internet and communication between two clients is similar to voice over IP (VoIP).

B. Architecture

The client/server architecture is selected for building the system. Three types of data are transmitted between the mobile client and Linux server, including control messages, audio signal, and text. TCP protocol is adopted for control messages, texts and audio to ensure the quality of transmission.

978-1-4244-5899-8/10/$26.00 ©2010 IEEE

At the client end, the audio engine is used to play back audio where audio send/receive module is in charge of audio and text transmission. The control module processes the interaction between client and server.

At the server end, control module processes the interaction between client and server and maintains the session between two mobile clients. The internal core engine is executing the kernel task of speech-to-speech translation, including cross-lingual and unified speaker adaptation.

C. Audio streaming

In the current Audio Services (VAS) API, only 8KHz sampling is supported for encoding and playing audio data in the same session. The raw audio data is originally recorded at 8KHz / 16bits PCM format in the mobile device. As shown in Figure 1, the real time S2ST system encodes PCM data into AMR narrow band (AMR-NB) and then transmits to server via TCP/IP protocol. Once the audio data is received at server end, it decodes the AMR coded audio into 8KHz/16bits PCM data. To ensure compatibility with the TTS module and cross lingual speaker adaptation (which requires the user's speech to adapt the TTS model) the audio is up-sampled to 16KHz. Following this, PCM data are fed into the core engine of the speech-to-speech translation module to translate to another language. After generating translated and synthesized speech in 16KHz, an inverse processing is performed and played in the other mobile client.

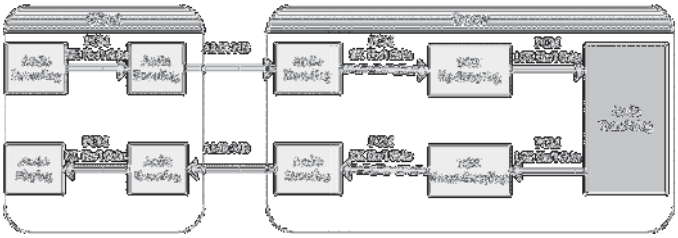

Figure 1. Audio data processing flows

III. SERVER OPERATION

The server is functionally divided into two separate parts: the core engine of the S2ST system and the peripheral control module. The peripheral control module has several functions of control, audio transmission, and audio reception corresponding to the client. The core engine of the S2ST system is the kernel and the most important part of the system. It can process the given voices in the pipeline through the ASR, MT, TTS modules. In order to make synthetic voice sound like the input user's voice, we adopt an adaptation module that can extract a user's individual voice characteristics when handling audio data in the ASR module and then apply this to the TTS module to produce personalized voice characteristics.

A. Core engine of speech-to-speech translation

The core engine of S2ST system in the server end is basically composed of ASR, MT and TTS modules. For real time purposes, we have performed some improvements to algorithms and scripts. Firstly, we carry out the training of speaker adaptation in speech recognition and cross-lingual speaker adaptation as an off-line process. This is necessary as speaker adaptation is one of the more computation expensive operations. Secondly, we revise the ASR engine "Juicer" decoder [3], which is initialized during the server start-up and then waits for speech input. This way, we can avoid having to repeatedly load models. The same process is taken to speed up the TTS engine.

1) ASR module

The ASR module incorporates steps for the generation of component weighted finite state transducers (WFSTs), composition and optimization with the AT\&T FST toolkit to build the Juicer decoder.

To speed up the real-time S2ST system, the Juicer decoder is modified into three parts, including initialization, decoding and release. In the current version of the real time S2ST system, two instances of the ASR engine are started during the server start-up. These instances use speaker independent models from the language pairs. When user login, new Juicer instances are started with these models if speaker adapted models exist for the specific user, otherwise the pre-initialized speaker independent instances are used according to the user's predefined input language.

Single pass decoding is performed in ASR module due to online requirement, whereas multi-pass decoding is used to do unsupervised speaker adaptation in off-line to generate speaker dependent ASR acoustic models. In real time S2ST system, we put the speaker adaptation part off-line and directly use the adapted models or speaker independent models for decoding.

a) AMR codec and up-sampling experiments

We have used 16KHz speech data for ASR and TTS experiments. In order to achieve good quality of synthesized speech, 16KHz sampling is regarded as a reasonable choice in HTS generation. In the current VAS, only 8KHz sampling is, however, supported for encoding and playback of audio data in the same session. Therefore, to enable operation of the TTS module and cross-lingual speaker adaptation which requires the user's speech to adapt HTS model, we carry out up-sampling of 8KHz audio data from the mobile device to 16KHz. The preliminary experiments have been conducted to assess the impact of AMR codec and up-sampling on the ASR module.

For Mandarin Chinese, we train Mel-Frequency Cepstral Coefficient (MFCC) speaker-independent (SI) and speaker adaptive (SAT) models for two pass decoder using the SPEECON database as training set. For testing, the in-house personal communication (PCOM) database is used For evaluation. For US English, MFCC SI and SAT models are trained using the WSJ0 database and evaluated on the November 1993 CSR 5k hub task.

Table 1 shows the recognition performance for both Mandarin and English. There, ACC means word accurate rate; AMR UP means AMR codec and up-sampling. In matched training and test conditions, about 4%-5% performance degradation is observed in both first pass and second pass decoding. The results of mismatched condition experiments, in which AMR codec and up-sampling MFCC features are tested on normal MFCC models for Mandarin, less than 15% recognition accuracy is obtained for both first pass and second

pass decoding. It is apparent, given present constraints, we have to adopt the AMR_UP models for decoding in the real time S2ST system, but the performance improvement for the AMR codec and up-sampling speech is expected in the future research.

Table 1: Experimental results of AMR codec and up-sampling.

Experiments conditions		Mandarin(ACC%)		English(ACC%)	
Models	Features	First pass (SI)	Second Pass (SAT)	First pass (SI)	Second pass (SAT)
Normal	Normal	76.4	81.3	89.1	94.0
	AMR-UP	8.0	14.0	N/A	N/A
AMR-UP	AMR-UP	72.8	75.9	84.3	89.5

2) MT module

For the MT part, we simply take use of Google's AJAX language API [4].

3) TTS module

In the TTS module, the hts_engine is used and acoustic feature parameters are generated from the adapted TTS models using a parameter generation algorithm that considers both the global variance of a trajectory to be generated and the trajectory likelihood. An excitation signal is generated by using mixed excitation of pulse plus band-filtered noise components with pitch-synchronous overlap and add (PSOLA) algorithm [5]. This signal is used to excite a mel-logarithmic spectrum approximation (MLSA) filter corresponding to the STRAIGHT mel-cepstral coefficients to generate the speech waveform. [6]

To speed up the real time S2ST system, hts_engine is also modified into three parts including initialization, synthesis and release. Four general TTS engines are started up by the server, one for each of combinations of language and gender. While user enrolling, if the speaker dependent models are available, they are started and used. Otherwise one specific hts_engine is used according to user's speaking language and gender.

4) Speaker adaptation module

In the real time S2ST system, two kinds of speaker adaptation are included. One is speaker adaptation for ASR and the other is cross-lingual speaker adaptation for TTS. In both cases, adaptation can be performed in a supervised or unsupervised manner.

The adaptation for ASR is conducted after a certain number of voice samples are collected. With the available voice samples, we can do unsupervised speaker adaptation to improve adapted models. The adapted models can be initialized and used to improve the decoding performance. The real time S2ST system also supports error correction scheme. So it could provide more accurate utterance text for the input speech, and supervised speaker adaptation could be applied accordingly.

The adaptation for TTS is conducted in the cross-lingual speaker adaptation module. A cross-lingual adaptation method based on a state-level mapping is used [7][8]. This mapping is derived from the Kullback-Liebler divergence (KLD) between pairs of model states from the source and target languages.

For supervised adaptation of TTS, TTS models from both input and output languages are used directly. We can extend this method to unsupervised adaptation by automatically transcribing the input data using the ASR module. Then, TTS models can be adapted directly from the ASR transcription in the same way as is done for the supervised case. Alternatively, we may use ASR models that utilize the same acoustic features as the TTS system. In this case, no other constraints need to be placed on the ASR. In particular, it is not necessary to use prosodic context dependent quinphones questions which would be necessary for TTS models.

In the current version of the real time S2ST system, the processes of speaker adaptation for ASR and TTS are both off-line. When the number of input sentences from mobile client reaches a preset number, the server informs the mobile client user if s/he is willing to start the adaptation. Once model adapted, the adapted models can be retained on the server and can be applied for this user.

B. Peripheral module

The peripheral module is actually supporting parts. It includes control module and audio process module. As illustrated in Figure 2, the main functions of the control module in the server is to manage information about the users, to maintain sessions between the pairs of clients, and to control the peripheral and core interaction module. In audio process module, audio decoding, receiving and sending are processed. Also, utterance text is processed in this module.

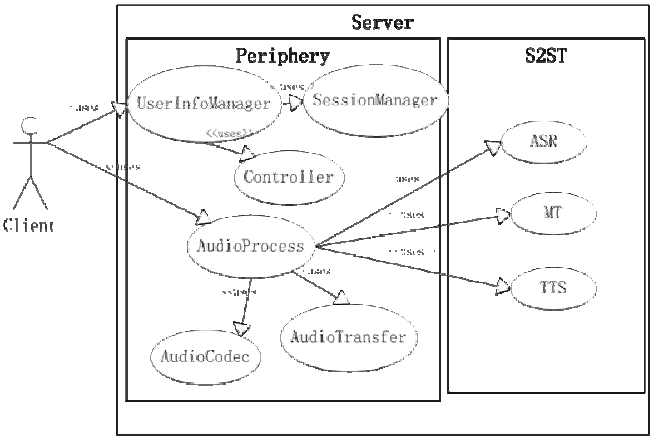

Figure 2: Peripheral control in Server End

C. Streaming and multi-threading support

We provide streaming and file-based ways for the information transmission between different modules in server end. Streaming makes it available that the S2ST system working in a real time way. In the same time, we can choose property to store the audio and text information in the server

end to make off-line ASR and cross lingual speaker adaptation for TTS doable.

In the same time, we improve the core S2ST system to support multi-threading. So, the S2ST system can work as a service to ensure multiple pairs of user to use it in parallel.

IV. User Interaction at the Client End

A. User interaction in mobile client

The client software consists of four parts: control module, audio engine, and the audio send/receive modules. The control module is in charge of sending and receiving control messages between the client and server. The audio engine mainly records, encodes, decodes and plays back audio data. Audio transmit and receive modules are used for sending and receiving audio data. In order to avoid conflicts, we use different socket ports for each module such as control, audio send and receive, with each module being run separately.

In the client end, some interactions associated with speaker adaptation are provided. Mobile user is asked if s/he wants to save his/her voices in the server for the speaker adaptation and if s/he wants to use his/her voices for speaker adaptation when sufficient voice data is collected.

Additionally, the mobile user can correct the recognized utterance text and update it to the server, thus providing functionality for supervised speaker adaptation in the ASR, as well as a means to improve the system performance and user experience.

B. User interface design in mobile client

The user interface is shown in Figure 3, in which the texts that the server recognized and translated are shown in the mobile display. The user can edit their recognized utterance then resend the corrected text to the server. When receiving messages from their "buddy", the user will receive the utterance text along with the synthesized speech.

Figure 3: User Interface in Mobile Client

V. Conclusion and Future Work

In this paper, we have developed the real time S2ST system that can easily and flexibly integrate components to implement speech-to-speech translation from end-to-end in mobile devices. The real time S2ST system is designed using a client-server architecture. On the client end, we implement the application for the NOKIA device from which mobile users can record input speech and play translated speech in the same session. The users can also edit the recognized utterance text to improve the quality of the machine translation. In the server, the application is implemented to manage communication between clients and to execute the core engine of the speech-to-speech translation. In the real time S2ST system, speaker adaptation for ASR and cross-lingual speaker adaption for TTS are implemented in off-line mode due to computational constraints as well as the necessity of collecting multiple utterances before adaptation can be performed. In its present state, the real time S2ST system can run in real time.

Concerning future work, first of all, cross-lingual speech adaptation shall be improved in further research so that the translated synthesized voices are of higher quality and better resembles that of the original speaker. And online adaptation will be integrated into the real time S2ST system while the rapid adaptation algorithms are available.

Secondly, ASR performance requires further attention, since the AMR-codec and up-sampling of input voices have degraded the performance compared the normal 16KHz speech data. Likewise, the TTS quality is degraded by the lower bandwidth of the adaptation speech.

Thirdly, we will continue to optimize the real time S2ST system in terms of complexity and memory. It is also possible to have entire S2ST in one mobile device stand alone, so that server is not necessary.

VI. References

[1] B Zhou, Y Gao, J Sorensen, et al. , "A hand held speech to speech translation system", Automatic Speech Recognition and Understanding, 2003.

[2] T Shimizu, Y Ashikari, et al., "Developing client-server speech translation platform", Proceedings of the 7th International Conference on Mobile Data Management, 2006.

[3] http://code.google.com/intl/ja/apis/ajaxlanguage/

[4] http://juicer.amiproject.org/juicer/

[5] E. Moulines and F. Charpentier. Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. Speech Communication, 9(5-6):453–468, 1990.

[6] H. Kawahara, I. Masuda-Katsuse, and A. Cheveign´e. Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: possible role of a repetitive structure in sounds. Speech Communication, 27:187–207, 1999.

[7] Y-J. Wu, Y. Nankaku, K. Tokuda, "State Mapping Based Method for Cross-Lingual Speaker Adaptation in HMM-Based Speech Synthesis," Interspeech 2009, pp.528-531, Brighton, U.K., 6-10 September, 2009

[8] K. Oura, J. Yamagishi, K. Tokuda, S. King and M. Wester "Unsupervised English-to-Japanese speaker adaptationfor HMM-based speech synthesis." ICASSP 2010, pp. 4954-4957 Dallas, USA, 14-19, Mar,2010