TOWARDS MULTI-MODAL INTERFACES FOR EMBEDDED DEVICES

Volker Fischer[1]*, Carsten Günther*[1]*, Jozef Ivanecký*[1]*,*
Jan Šedivý[2]*, Luboš Ureš*[2]

[1]*IBM European Voice Technology Development*
Vangerowstr. 18, 69115 Heidelberg, Germany
[2]*IBM Research, Voice Technologies and Systems*
Murmanská 4, 100 00 Praha, Czech Republic

{*vfischer, gcarsten, ivanecky*}*@de.ibm.com*
{*kleindienst, jan_sedivy*}*@cz.ibm.com*

Abstract: The paper describes our efforts towards a multi–modal interface for embedded devices. Multi-modality is becoming more important especially in an embedded scenario where the ”standard” ways of input (keyboard, mouse, stylus) as well as output (display) are constrained and less comfortable. We explore the usage of the most natural human interface – namely voice – to extend available input and output capabilities and simultaneously preserve the standard way of using such devices.

1 Introduction

VoiceXML is currently becoming a standardized way to design voice controlled applications. Whereas with HTML and additional tools such as ECMA script or Perl it is possible to design user interfaces and simple but already dynamically interacting applications, the combination of VoiceXML and HTML allows us to create a multi-modal interface that simultaneously supports also human voice. Since no standardized combination of both markup languages is available, we implement a new approach that combines VoiceXML and HTML in a newly developed multi-modal VoiceXML browser. The browser parses and interprets the combination of VoiceXML/HTML documents, controls the speech recognition engine (IBM’s embedded voice engine with low footprint), and manages the multi-modal dialog with the user.

Figure 1 shows the general architecture of our multi–modal system. The central part is the VoiceXML Browser interpreting the loaded VoiceXML pages and controlling the embedded speech recognition and speech synthesis engines. The browser architecture supports input via voice or touch screen and audio or visual output. Server-generated VoiceXML pages can be loaded via a protocol stack supporting different wireless access means.

To demonstrate the capabilities of this approach we present a SMS ”dictation” application that allows message creation either via voice or classical text input. The entire application is written in VoiceXML, HTML and ECMA script, and runs on the new multi-modal VoiceXML browser. Given the used technology, the application is completely platform independent and can be executed on each platform that is capable of running the browser.

The remainder of the paper is organized as follows: In Section 2 we give a brief overview over speech recognition techniques for embedded devices. Assuming some familiarity with the basic concepts of VoiceXML, HTML, and ECMA script, in Section 3 we focus on the interaction and synchronization of voice and typed input in the newly developed multi-modal VoiceXML browser. Section 4 introduces an example application that — amongst others —

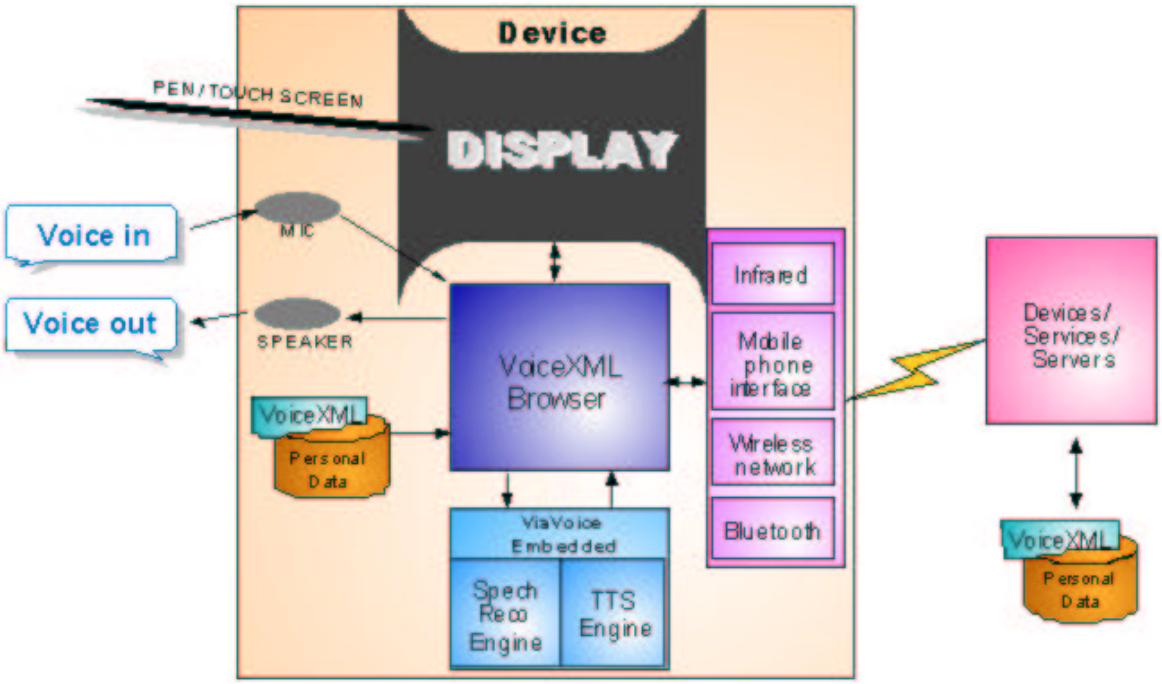

Figure 1 - Multi–modal browser architecture.

serves as a proof of concept for the ideas developed in Section 3, and a brief summary is given in Section 5.

2 Embedded Speech Recognition

Voice driven interfaces for consumer devices such as PDAs, mobile phones, smart–phones, or car navigation systems are becoming increasingly popular. While today there is no doubt that the overwhelming majority of such appliances will make use of the well established Hidden Markov Model (HMM) based approach to statistical speech recognition, it is also obvious that special needs of the embedded and mobile domain must be considered. In the remainder of this section we will review the basic requirements imposed by the scenario under consideration and will describe how these are taken into account in the training of acoustic models as well as in recognition.

Emerging standards like VoiceXML offer an up to now unknown degree of freedom in the dialog design for voice driven interfaces. Therefore, sub–word HMM based recognizers that offer a large flexibility in the vocabulary and dialog design, are becoming indispensable, whereas whole–word based recognizers, which are still predominant in many small vocabulary embedded speech recognition applications, will more and more disappear from the scene. Recent industrial joint efforts in the creation of high quality speech data bases for consumer devices (see [15]) already take this into consideration by collecting a substantial amount of rich context data.

It is well known that recognition accuracy improves significantly, if the acoustic model is trained with data that matches the target domain, e.g. in the type and amount of environment noise . Since voice interfaces for embedded devices are expected to work under a wide variety of conditions — consider, for example, the use of your PDA in your office, your car, or in a train station — the acoustic model must therefore incorporate training speech that properly reflects the characteristics of different environments.

Finally, but probably most important, the design of an embedded speech recognizer has to deal

with only limited computational resources, both in terms of CPU power and memory capacity, that today's embedded devices can offer. While some applications may run entirely on the local device and therefore require a relatively compact acoustic model, others may defer parts of the recognition process to a recognition server, which requires compatibility of at least the client's and server's acoustic front–end.

The latter is ensured by the use of a standard acoustic front–end, that computes 13 Mel Frequency Cepstrum Coefficients (MFCC) every 10 milliseconds. Utterance based cepstral mean subtraction and C0 normalization are applied to compensate for the acoustic channel and the first and second order delta coefficients are computed to capture the temporal dynamics of the speech signal. While more recently other feature extraction techniques such as MVDR [6] have been demonstrated to provide superior accuracy in noisy environments, cepstral coefficients are well suited for low bit–rate compression and transmission [14]. Moreover, speech can be reconstructed from cepstral coefficients and a simple pitch tracker [4], which we consider as a prerequisite for the text–to–speech component of future multi–modal interfaces for embedded devices.

The use of additive noise from the real environment is a well known method to increase the robustness of a speech recognizer under adverse conditions (e.g. [17, 3]). While we concentrated on the use of engine noise for in–car speech recognition in the past [7]), we have more recently started to incorporate a wider variety of non–stationary noise types that were collected with commercially available PDAs.

Recognizer training comprises the definition of a suitable HMM inventory and the determination of the HMM parameters. For that purpose, the training data is viterbi–aligned against its transcription in order to obtain an allophonic label for each feature vector. Context dependent non cross–word triphone HMMs are obtained from the leaves of a decision network [1] that is constructed by asking binary questions about the phonetic context P_i for each feature vector, $i = -1, \ldots, 1$. These questions are of the form:"Is the phone in position i in the subset S_j?", and the subsets are derived from meaningful phone classifications commonly used in speech analysis. We found small gains from using only clean data for the construction of the HMM inventory. Finally, the data at each leaf of the network is used in a k–means procedure to obtain initial output probabilities whose parameters are then refined by running a few iterations of the forward–backward algorithm.

The k–means procedure follows a simple rule of thumb and equally distributes a fixed number Gaussian mixture components across the HMM states. Usually, in a highly dynamic and heterogeneous environment, an increased total number of Gaussians can significantly improve the recognition accuracy. However, this is infeasible for applications that have to deal with a limited amount of memory, and therefore the determination of an appropriate acoustic model size is of particular importance.

The Bayesian Information Criterion

$$BIC(M) \;=\; \log L(X, M) - 1/2(\#(M) \cdot \log(n)) \tag{1}$$

is a model selection criterion that penalizes the likelihood L(X,M) of a data set X of size n by the number of parameters #(M) in the model [5]. We used BIC based clustering as an alternative to the k–means procedure and found that the method can produce both smaller models and more accurate results; cf. [7].

The so created acoustic model can run with either IBM's large vocabulary continuous speech recognition engine, which employs a fast pre–selection of candidate words and an asynchronous stack search algorithm [11, 8], or with a time–synchronous viterbi–decoder. The latter is the core of IBM's Embedded Speech Engine (ESE), which is designed for the use with a moderate vocabulary size and finite state grammars. The highly portable and scalable ESE can run on any

suitable 32 bit general–purpose CPU; see [2] for an overview on design issues and performance.

3 The Multi-modal Interface

This section gives a brief overview on the multi-modal capabilities of the multi-modal VoiceXML browser. Since there is yet no standard for Web-based multi-modality (a W3C interest group has been founded only recently), developers are trying out various approaches to support both visual and oral input in a coordinated manner. The multi-modal techniques introduced in this section constitute one of the possible approaches. We picked this particular solution because it allows to naturally extend the VoiceXML execution model and thus to take advantage of VoiceXML's built-in dialog management capabilities.

In the design of a multi-modal system, one of the important architectural questions is how (and to what extend) the user actions and state changes are propagated from one modality to another (i.e. *the synchronization model*). The synchronization model implemented in our browser is powerful enough to drive applications such as SMS dictation, yet quite compact, and straightforward to be implemented on a commercially available PDA such as Compaq iPAQ. The synchronization framework is now introduced in some more detail.

The multi-modal capabilities of the multi-modal VoiceXML browser are extensions to the standard VoiceXML input (DTMF and speech recognition) and output (pre-recorded audio or TTS) that support the rendering of HTML pages, and the use of HTML links and forms for user input. We allow VoiceXML code to control loading of pages into the HTML component (page-level synchronization) and the HTML component to send specific user-related updates to the VoiceXML component (sub-page-level synchronization).

Displaying HTML pages is implemented through extensions to the semantics of the VoiceXML <prompt> tag. HTML documents referenced from VoiceXML code are passed to the HTML viewer and treated as a page to be displayed. The page stays displayed until it is rewritten with a new content. Instead of referencing an URL, the designer has an option to construct HTML pages at runtime via ECMA script procedures included as parts of the VoiceXML document. Runtime generation of HTML pages is beneficial for multi-modal applications with highly dynamic visual content, which is the case in, for example, SMS applications.

The HTML pages displayed can contain links and forms that can be used to provide explicit synchronization between HTML and VoiceXML components. For example, links with values are used as synchronization anchors that convey the information on user's clicking to the VoiceXML component. Similarly, the values of HTML form variables are propagated to the VoiceXML component upon the form completion (on submit). Such sub-page-level synchronization is necessary to implement multi-modal applications that support dynamic filling of HTML forms via a GUI and voice.

We refined the above mentioned techniques during the development of several multi-modal case study applications. Experience gathered during the development procedure shows that the current set of multi-modal extensions is sufficient for the efficient authoring of PDA-scale, multi-modal applications.

4 Application

As a demo application we picked up a typical embedded device application, namely SMS writing. Because the application runs in the multi–modal browser and the entire message can be composed by voice, the application may properly be named as "SMS dictation".

One of the still existing constraints for speech recognition on embedded devices is the limited vocabulary size. Because of the restricted amount of memory and fairly low CPU power, we can

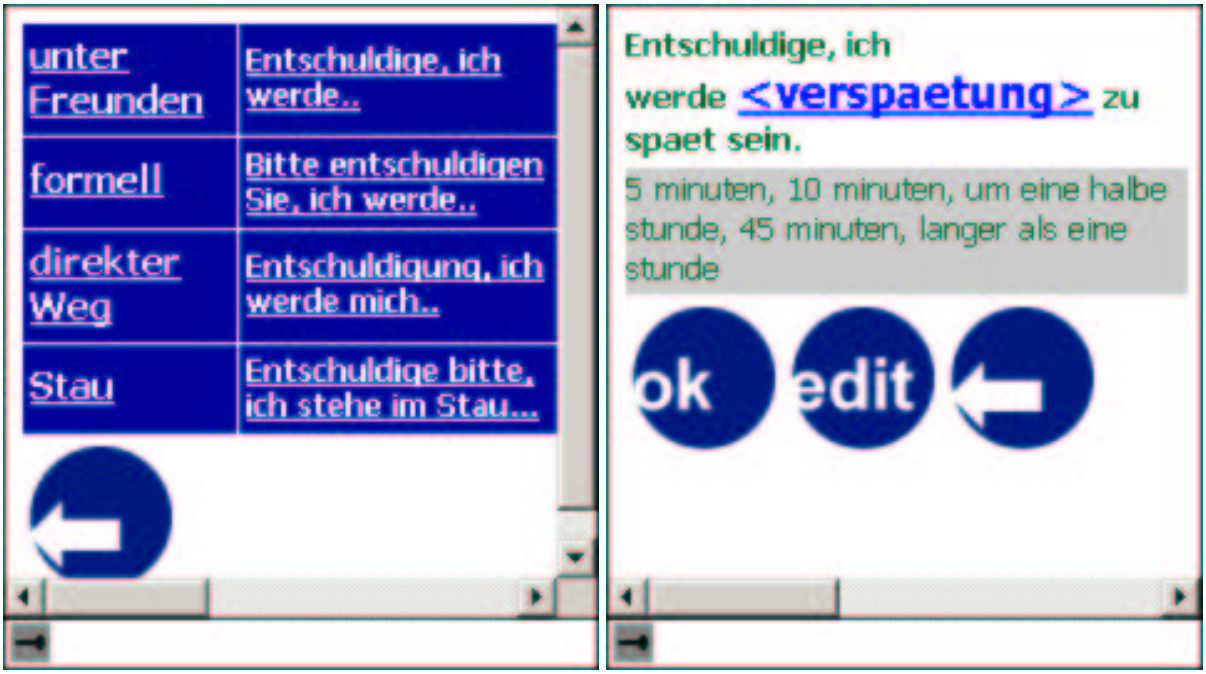

Figure 2 - Dialog example from SMS dictation.

not afford today to build a speech recognition system with free dictation capabilities. Instead one has to create smaller, grammar based applications, which the SMS dictation application really is. For that purpose we defined a set of templates that covers several domains of the SMS messaging. With such templates we can generate thousands of different SMS messages, which can be generated step by step or can be directly dictated.

The entire application is written in VoiceXML, HTML and ECMA script. Available sentences are stored in templates form in XML format and are easy to modify. Therefore, the variety of possible sentences can be easily extended. The following templates belong to a category *verspätung* (which may be used to formally apologize for delaying a meeting) and demonstrate how the dialog is described in XML format.

```
<template keyword="formell">
  Bitte entschuldigen Sie, ich werde
  <fill grammar="delay">verspaetung</fill> zu spaet kommen.
</template>
```

From this simple template grammars for several sentences are generated. These grammars define both natural sentences like *Bitte entschuldigen Sie, ich werde 20 minuten zu spät kommen.*, but also shortened utterances like *verspätung, formell, 20 minuten*, from which the same explicit sentence will be finally constructed. Thus, one of the application features is that the user can arbitrarily change the elaborateness of his voice input at any time.

The same mechanism applies to text or stylus based input. Within each category we have several sentences represented by a key word, which can be dictated or clicked instead of the entire sentence. After picking up the the sentence, the user proceeds to the next step – if necessary – to fill in missing informations. When all information is entered, the final sentence is generated. The user can optionally edit the final sentence and add some additional information. At any time it is possible to go back or start from the beginning, and the multi-modal capabilities are always available. Figure 2 gives dialog examples from the above mentioned category.

After the final SMS message is generated, the dialog is passed to the address book. Searching the address book can be the slowest operation of the entire process, if the address book is large.

Here, voice input gives the greates relief, since the user just needs to say a name instead of scrolling to a large list via stylus or buttons.

In the address book we store names, phone numbers, email addresses and the nationality of a person, so that — dependent on the available connections — the composed SMS can be send either as a regular SMS, as a regular email, or as a email via SMS gateway. Storing the recipient's nationality allows us to translate the created message to the recipient's language if message templates in the corresponding language are available.

The following template for German, Spanish and Italian is from category *treffen*, sub–category *vereinbaren*:

```
<template keyword="formell">
 Ich wuerde gerne <fill grammar="onday">am tag</fill>
 ein Treffen um <fill grammar="time">wann</fill> vereinbaren.
</template>

<template keyword="formal">
 Quiero concertar una cita <fill grammar="onday">tal dia</fill>
 a <fill grammar="time">tal hora</fill>.
</template>

<template keyword="garbato">
 Che ne pensi di incontrarci <fill grammar="onday">giorno</fill>
 alle <fill grammar="time">ora</fill>.
</template>
```

Since we know which templates has been used to generate the final sentence, we can easily translate the sentence to another language by using the equivalent templates and filling the same optional information translated on the grammar level the same way as templates here. At the current stage of development we support monolingual voice input (in several languages) that can be translated to Czech, English, French, German, Italian, Slovak and Spanish.

5 Summary

In this paper we described various aspects of the development of a multi-modal interface for embedded devices. After sketching methods for the training of highly noise robust, low footprint, HMM based acoustic models, we focussed on synchronization aspects in a newly developed multi-modal VoiceXML browser which is capable of processing different input modes, namely voice and typed input, simultaneously. A SMS dictation application was used both to discuss issues in the design of real-life applications as well as to demonstrate the feasibility of the chosen approach. While platform independence and the arbitrary use of an input modality are important features of the browser, we consider the template based SMS translation and the integration into a well established IT-infrastructure as distinct characteristics of the application.

Acknowledgement. The authors wishes to thank the entire Voice Technologies and System Group, IBM Research, Prague, for the design and development of our embedded and multimodal technologies. Special thanks to Vladimír Bergl, Martin Labský, Bořivoj Tydlitát, and Jan Kleindienst for their contributions to this work.

Literatur

[1] L. Bahl, P. de Souza, P. Gopalakrishnan, D. Nahamoo, M. Picheny. Context-dependent Vectoy Quantization for Continous Speech Recognition. In *Proc. of the IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, Minneapolis, 1993.

[2] L. Comerford, D. Frank, P. Gopalakrishnan, R. Gopinath, J. Sedivy. The IBM Personal Speech Assistant. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Salt Lake City, Utah, 2001.

[3] R. Bippus, A. Fischer, V. Stahl. Domain Adaptation for Robust Automatic Speech Recognition in Car Environments. In *Proc. of the 6th Europ. Conf. on Speech Communication and Technology*, volume 5, pages 1943–1946, Budapest, 1999.

[4] D. Chazan, R. Hoory, G. Cohen, M. Zibulski. Speech Reconstruction from Mel Frequency Cepstrum Coefficients. In *Proc. of the IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, pages 1299–1302, Istanbul, Turkey, 2000.

[5] S. Chen, P. Gopalakrishnan. Clustering via the Bayesian Information Criterion with Applications to Speech Recognition. In *Proc. of the IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, pages 645–648, Seattle, 1998.

[6] S. Dharanipragada, B. Rao. MVDR Based Feature Extraction for Robust Speech Recognition. In *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, Salt Lake City, Utah, 2001.

[7] V. Fischer, S. Kunzmann. Bayesian Information Criterion based Multi-style Training and Likelihood Combination for Robust Hands Free Speech Recognition in the Car. In *Proc. of the IEEE Workshop on Handsfree Speech communication*, Kyoto, 2001.

[8] P. Gopalakrishnan, L. Bahl, R. Mercer. A Tree Search Strategy for Large Vocabulary Continuous Speech Recognition. In *Proc. of the IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, pages 572–575, Detroit, 1995.

[9] S. H. Maes, R. Hosn, J. Kleindienst, T. Macek, T.V. Raman, L. Seredi. A DOM-based MVC Multi-modal e-Business. In *Proc. of the IEEE International Conference on Multimedia and Expo (ICME2001)*, Tokyo, Japan, 2001.

[10] Multimodal Requirements for Voice Markup Languages, W3C, *http://www.w3.org/TR/multimodal-reqs*, W3C Working Draft, June 2000.

[11] M. Novak, M. Picheny. Speed Improvement of the Time-Asynchronous Acoustic Fast Match. In *Proc. of the 6th Europ. Conf. on Speech Communication and Technology*, Budapest, 1999.

[12] S. Oviatt. Ten Myths of Multimodal Interaction. *http://www.cse.ogi.edu/CHCC/Papers/sharonPaper/Myths/myths.html*,2000.

[13] S. Oviatt. Taming Recognition Errors with a Multimodal Interface. *http://www.cse.ogi.edu/CHCC/Publications/cacm9-2000/cacm9-2000.htm*, 2000.

[14] G. Ramaswamy, P. Gopalakrishnan. Compression of Acoustic Features for Speech Recognition in Mobile Environments, In *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 977–980, Seattle, 1998.

[15] R. Siemund, H. Höge, S. Kunzmann, K. Marasek. SPEECON – Speech Data for Consumer Devices. In *Proc. of the 2nd Int. Conf. on Language Resources & Evaluation*, pages 883–886 Athens, 2000.

[16] The MIT Galaxy System. *http://www.sls.lcs.mit.edu/GALAXY.html*, *Spoken Language Systems Group*, MIT Laboratory for Computer Science, Cambridge, MA, USA.

[17] A. Varga, H. Steeneken, M. Tomlinson, J. Jones. The NOISEX-92 Study on the Effect of Additive Noise on Automatic Speech Recognition. In *Booklet of the NOISEX-92*, CD-Rom, 1992.

[18] VoiceXML 2.0, W3C, *http://www.w3.org/TR/2001/WD-voicexml20-20011023*, Working Draft, Oct 2001.