A real-time spoken dialogue demonstrator

Harald Ljøen *Finn Tore Johansen* *Ingunn Amdal* *Stein Kulseth* *Per Olav Heggtveit*

Telenor Research and Development, Instituttveien 23, 2007 Kjeller

E-mail: (Harald.Ljoen Finn.Johansen Ingunn.Amdal Stein.Kulseth Per.Heggtveit)@fou.telenor.no

ABSTRACT

The TABU project at Telenor Research and Development is working with speech-based user interfaces for advanced telephone services in Norwegian. For the purpose of demonstrating the concept, a real-time demonstrator was set up in Oslo for the Telenor Expo exhibition held in March 1995. The demonstrator was a multimodal dialog system with input from speech only and output through speech and window-based computer console. This paper presents the demonstrator and the technology used in it: the hardware, the speech synthesiser, the speech recogniser and the implementation of the dialogue.

1. INTRODUCTION

The TABU project at Telenor Research is working with speech-based user interfaces (**TA**le**B**aserte br**U**kergrensesnitt) for advanced telephone services. The activities in this project were chosen to be presented at Telenor Expo held in Oslo as examples of research activities in the company.

Telenor Expo was an exhibition arranged in March 1995 simultaneously in five major cities in Norway: Oslo, Stavanger, Bergen, Trondheim and Tromsø. The purpose of the exhibition was to mark the transition from a government agency to a state owned joint-stock company and to present the company, its new name and logo and the company profile to the public.

We had been working with speech synthesis [1] and speech recognition [2] for some time, but had not yet started the planned activities in spoken dialogue design. Nevertheless, we chose to present a spoken dialogue system to demonstrate the existing technology in an application-orientated context.

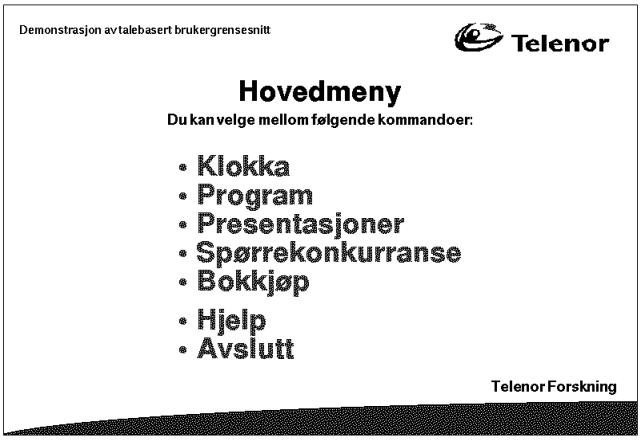

Figure 1. Visual output for the main menu

2. THE DIALOG DEMONSTRATOR

Ease of demonstration was a key issue as our audience at Telenor Expo would have limited time at their disposal and had to get an immediate overview of the user interface. Also, we needed to make our demonstration interesting not only for the person talking with the system, but for bystanders as well. We therefore chose to present a system based on a hand-held microphone and a loudspeaker, assisted by the window-based computer console to visualise the operation.

While screen output is not an option when designing real telephone applications, we found that it added value to our demonstration, and somehow it simulated a familiarity with the interface that one could expect in a real service.

During the operation of the system, one window on the computer screen maintained an updated display of the choices (keywords) at any point in the menu hierarchy. The main menu is shown in figure 1. In another window one could follow the operation of the recogniser through messages like "I'm listening", "I'm thinking", "I recognised keyword-so-and-so", etc. This proved to be an effective demonstration of the underlying speech recognition techniques.

The entry menu of the dialogue system had only

one choice, namely

- **Hallo, Arne** Causes the system (called Arne) to greet you welcome to the demonstration, and to enter the main menu.

The main menu had the above entry plus the following keywords:

- **Klokka (the time)** Causes Arne to tell you the time in an "oral" format, e.g. "seventeen past three" instead of "fifteen-seventeen".
- **Program** Arne will read from the next two hours of the official program at Telenor Expo.
- **Presentasjoner (presentations)** Arne will enter a submenu where you can tell him to read a short presentation of either one of the TABU project or Telenor Research or Telenor Expo.
- **Spørrekonkurranse (quiz)** Arne will present a short quiz, where you will be prompted to answer a few questions. Along with each question Arne will provide three alternative answers, one of which is correct. When you are finished, Arne will tell you your accumulated score.
- **Bokkjøp (book ordering)** Arne enters a book ordering submenu where you can order books written by the "four great" Norwegian authors: Henrik Ibsen, Bjørnstjerne Bjørnson, Alexander Kielland or Jonas Lie. There are four choices under each author, and you are prompted to confirm each choice you make. As this is a demonstration only, the books are not actually dispatched and mailed to you.

The following keywords were part of the main menu and all submenus:

- **Hjelp (help)** Causes Arne to provide context-sensitive help.
- **Avslutt (quit)** Causes Arne to leave the current submenu and return to the previous menu.

All speech output used synthesised speech, even fixed messages that could have used recorded natural speech. Bearing in mind that this was a demonstration, not a real application, we figured that it would be more interesting to hear Arne speaking than just a recorded human, and that the lower quality of synthesised speech was not a key issue here. Thus we also avoided switching between natural and synthesised speech, which would have produced a somewhat annoying effect.

3. THE SPEECH SYNTHESISER

The text-to-speech synthesiser was the latest version of "Arne", developed by Telenor Research [1] with assistance from SINTEF DELAB. This synthesiser consists of two main parts, the text-to-phoneme generator and the phoneme-to-speech generator.

3.1. The text-to-phoneme generator

The text-to-phoneme generator takes all kinds of text as input, and transforms it, sentence by sentence, into the corresponding phoneme string and parameters describing the duration of each phoneme and variation in pitch.

The first step is the text normalisation. Numbers, abbreviations, dates, special characters etc. are transformed into their standardised text representation.

The pronunciation of known words are found in a pronunciation lexicon of about 60.000 words, whereas the pronunciation of unknown words are determined from a set of about 700 phonological rules.

A phrase parser is used to resolve word pronunciation ambiguities. In our lexicon there are about 1000 graphemic forms (written words) with ambigous pronunciation. The problem of resolving ambiguities is far from trivial, because of its dependency on the semantic context of the word. The phrase parser can only exploit local syntactic structure, and therefore a wrong pronunciation is occasionally selected. To improve the pronunciaton, disambiguation is now a subject for further study.

Wordclass information and syntactic structure is applied to decide what words to accent and deaccent. This is used as input to the intonation model and the duration model which determine the exact pitch-movement across the sentence, and how the duration of the sentence is distributed on each of the phonemes.

Phoneme codes together with corresponding values of the pitch and duration are passed on to the phoneme-to-speech generator.

3.2. The phoneme-to-speech generator

The speech generator is based on the PSOLA algorithm [3] for changing pitch and duration in recorded natural speech. A sound inventory of 1200 units is made by recording isolated nonsense words, and semi-automatically segmenting them into the sound units (diphones) we use for concatenation. A diphone unit is defined to start in

the middle of one phoneme and end in the middle of the next.

The nonsense words are constructed in a systematic way to cover all the diphones needed [6]. It is easier to control the phonetic context of each sound when you use nonsense words and not real words. The drawback is that the words could be more difficult to read for a non professional and that stressed reading could produce more unnatural sounding diphones.

The phoneme codes received from the text-to-phoneme module specifies what diphones to concatenate, and the pitch and duration values of each phoneme gives the PSOLA algorithm the necessary information to modify the speech according to the model.

4. THE SPEECH RECOGNISER

The speech recogniser was built using HTK [4] and the TABU.0 Norwegian telephone speech database [2]. TABU.0 contains among other things a control word part and a continuous speech part, both of which were used to train the recogniser.

The total vocabulary consisted of 39 words, but maximum 8 words were allowed at any specific point in the dialogue.

4.1. Phoneme based recogniser

Continuous speech can be used to train speaker and context independent phoneme models. The phoneme recognition accuracy is about 50% for this type of phoneme models. By restricting the recogniser to detect words, word recognition accuracy can be brought up to 95% or more, depending on the vocabulary size and the phonetic dissimilarity between the vocabulary words. This approach has the nice property that acoustic models can be trained once and for all, and vocabulary words are thereafter specified by giving their phonetic transcription. Such a recogniser is very flexible with respect to the employed vocabulary. The vocabulary can even be dynamically defined at run-time if an on line text-to-phoneme conversion is available.

The phoneme models for this dialogue were trained on the part of TABU.0 from the south-eastern part of Norway as the demonstration was to be held in Oslo. In choosing the number of parameters there is a tradeoff between accuracy and recognition speed. The final choice was 12 LPC cepstrum coefficients with deltas and delta energy. The HMM-models contained 3 states per phoneme and 4 continuous density Gaussian mixtures.

4.2. Word based recogniser

An alternative to forming words from phoneme sequences is to use word models directly, i.e. the recogniser is trained on utterances of each vocabulary word. This results in a recogniser with higher accuracy due to the fact that coarticulation effects between phonemes in the words are taken into consideration. The drawback, however, is that the vocabulary has to be decided before collecting the training database. Database collection is a time and labour consuming task which cannot be undertaken for small demos such as the one described in this paper. However, in TABU.0 there is a control word section containing 18 control words. Among these were the following four that we used in our dialogue: "ja" (yes), "nei" (no), "hjelp" (help) and "avslutt" (quit). We trained word models for these keywords and used them in addition to the corresponding concatenated phoneme models.

4.3. Garbage modelling

Untrained users tend to speak to the machine in unforeseen ways. They may use words that are not in the vocabulary, and they often use legal vocabulary words in a context of out-of-vocabulary words. Apart from this, they may cough and make other kinds of noise that can be a challenge to the recogniser. A finite vocabulary speech recogniser will try to match the utterance to one of the vocabulary words, and if there is no out-of-vocabulary handling the system may be perceived as useless. If, for example, the user makes a comment to his friend, and this makes the dialogue step deeply into the spoken dialogue menu system, the user will probably regard the system as user unfriendly.

It is therefore necessary to detect and reject out-of-vocabulary speech. For use in this demonstration we trained one generic phoneme model from all the available phonemes in the database. We then allowed any number of this generic phoneme in sequence to form a "GARBAGE" word model. The recogniser then compared the likelihoods of the "GARBAGE" word with the likelihoods of the ordinary vocabulary words and chose the word with the highest likelihood. This simple scheme caught surprisingly many of the out-of-vocabulary utterances.

Anticipating the acoustic environment at the exhibition, we also trained a simple noise model from recordings of cantina noise, lab noise, coughs, breathing etc. Together with a silence model trained on TABU.0 this model helped sorting out

most of the noise.

5. THE IMPLEMENTATION

5.1. Hardware

The hardware platform was an ordinary single-CPU SPARC 20 UNIX workstation with a 60MHz SuperSparc CPU module installed. The internal audio input and output ports were used. In addition to the UNIX machine with its console (screen, keyboard and mouse), we used a microphone with a microphone amplifier for speech input, and a loudspeaker with built-in power amplifier for speech output. The audio port was set to 16 kHz sampling frequency and 16 bits resolution.

5.2. Software

The software had three main components: a speech synthesiser, a speech recogniser and a dialogue definition script.

5.2.1. Speech synthesiser

The speech synthesiser was our latest version of "Arne" [1]. At the time this was a UNIX-based system programmed in C++. The text-to-phoneme generator and the phoneme-to-speech generator were implemented as two different UNIX processes, where data was piped from one to the other. UNIX pipes were also applied to input text, and to output speech to a play program. The diphones were sampled and output at 16 kHz with 16 bit resolution.

5.2.2. Speech recogniser

The speech recogniser consisted of three programs connected in a UNIX pipe:

- the PCM filter and utterance detection
- the feature extractor
- the Viterbi decoder

The speech input from the microphone was sampled at 16 kHz with 16 bits resolution. Our models were, however, trained on 8 kHz A-law PCM companded data, so the input was filtered according to the ITU G.712 recommendation and decimated to 8 kHz using routines from the ITU UGST software tools [5].

The utterance detector and the PCM filter were implemented as a C-program. We employed utterance detection for two reasons:

- By segmenting the incoming speech in utterances, we reduced the computational load on the recogniser
- The utterance detector reduced the delay by forcing the recogniser backtrack and output the recognised keywords earlier

The utterance detector was based on level estimates of incoming speech and background noise. The begin and end thresholds were adapted to the changing conditons in the exhibition hall.

The feature extractor was HCodeRT from a special real-time version of the Hidden Markov model Toolkit (HTK) [4] kindly provided by Entropics Research Labs. The ouput from the utterance detector was formatted to be readable by HCodeRT.

The speech recogniser itself, the Viterbi decoder, was HViteRT from the above mentioned special real-time version of HTK.

After some lab-tests we found it necessary to include a special r-model trained on speech from the western part of Norway. We also discovered some words in the vocabulary that were more easily confused, for instance "program" and "klokka", but unfortunately to late to change the keywords because the poster with the main menu already was printed.

We used the control word part of TABU.0 with 18 word vocabulary for tests to find a suitable parameter configuration. The test consisted of a total of 926 utterances from the south eastern part of Norway. Our phoneme based real-time recogniser delivered 92% correct recognition on this task.

5.2.3. Dialogue

The dialogue was implemented by UNIX C-shell scripts, one for the main menu and one for each of the three submenus in the system. The scripts were responsible both for interpreting the recognised user input (or lack of input) and taking appropriate action, and for controlling the system resources: recognition pipe, speech synthesiser, audio interface and the screen output.

In general, at each point in the dialogue the scripts would:

- Prompt the user for input
- Display the vocabulary on the screen
- Configure the recogniser with the active vocabulary, and set the timeout value
- Await a recognised keyword or timeout, then:
  - Stop the prompt, if not finished (enabling talk-over)
  - Match the recognised word with one in the current active vocabulary

- Respond by speaking the requested information, or by confirming the user's choice
- Move to the next part of the dialogue

During the recognition phase the dialogue script would wait while the recogniser output was piped to a PERL filter script that would filter out recognised silence, noise and out-of-vocabulary words. Except for timeouts, this filter did not stop executing until a valid keyword was found, which was then passed to the dialogue script.

In most cases, when the response consisted of a fixed message, the speech was synthesised in advance and stored in a file to be played. However, in a few cases, e.g. telling the time, the message texts were generated on the fly and piped to the synthesiser for real-time speech synthesis.

The on-screen menus (cf figure 1) were stored as postscript images and displayed using the X application xv. The recogniser messages were simply ascii text written to a large-font terminal window (xterm).

5.3. Real-time performance

The recognition delay was about 1 - 3 seconds. This delay was caused by the recogniser and PCM-filtering. We chose to have fairly complex models to get a reasonable recognition accuracy even though this would cause increased delay. The PCM-filter introduced some delay, but helped the accuracy.

Occasionally the delay could be as much as 5 - 6 seconds. These exceptional long delays were caused by one or more of the following reasons:

- The utterance detector failed to detect end of utterance
- In cases of short prompts or talk-over, the initialisation of the recogniser might not have finished
- Long utterances or several utterances would cause longer delay as the recognition delay would accumulate

When active, the recogniser used as much CPU as possible. The initialisation of the recogniser with a new vocabulary would take 4 - 5 seconds with all available CPU at disposal. The utterance detection and PCM filtering used aproximately 10% of the CPU during recognition.

The speech synthesis ran in real-time using up to 25% of the CPU. The lexicon, diphonelibrary and the other programs needed for synthesis took about 20 MByte memory in this version which were not optimalised regarding memory consumption. The speech synthesis in this demo was only used for short messages and operated at no point simultaneously with the recognition.

The graphic output displayed by the program xv used only small parts of CPU and not in conflict with any other parts of the dialogue.

6. CONCLUSION

We have demonstrated a real-time Norwegian spoken dialogue system implemented on standard hardware using standard software tools, our own copyrighted software and our own telephone speech database TABU.0.

The public were invited to speak to Arne themselves and the majority of reactions were positive. Many were really impressed, while a few were unfortunate enough to come by just in time to experience one of our daily system failures. A few others experienced that Arne did not recognise them properly. The vast majority of the visitors, however, were able to operate the system with reasonable accuracy.

REFERENCES

[1] J E Natvig, P O Heggtveit. *En sanntids demonstrator for norsk tekst-til-talesyntese.* Technical report TF R 15/93 (in Norwegian), Telenor Research, Kjeller, 1993.

[2] H Ljøen, I Amdal, F T Johansen. *Norwegian Speech Recognition for Telephone Applications* Proc. NORSIG-94, Ålesund, 1994.

[3] F J Charpentier, M G Stella. *Diphone synthesis using an overlap-add techique for speech waveform concatenation.* Proc. ICASSP-86, Tokyo, 1986.

[4] *HTK - Hidden Markov Model Toolkit V1.5.* Cambridge University Engineering Department and Entropic Research Labs., Sept. 1993.

[5] *Software tools for speech and audio coding standards.* ITU recommendation G.191

[6] Georg Ottesen. *Lydgenerering i norsk tekst-til-tale system.* Technical report STF40 F89172 (in Norwegian), SINTEF DELAB, Trondheim, 1989.