Multimedia SMS Reading in Mobile Phone

R. Talafová [1,2], G. Rozinaj [1]
[1)]Slovak University of Technology, Bratislava
Faculty of Electrical Engineering and Information Technology
Department of Telecommunications
Ilkovicova 3, 81219 Bratislava, Slovakia
[2] Siemens-PSE
Dúbravská cesta 4, 845 37 Bratislava
Phone: +421-2-68279414 Fax: +421-2-68279

Abstract – *This paper is devoted to the speech synthesis and development of a speech synthesiser for a mobile cell phone. The presented results are a part of the more complex project for multimedia reading of sms on the mobile phone. After receiving the sms a talking head based on a photowill appear on the screen and animate the reading while the speech will be synthesized in paralel. This work further analyses an implementation of the speech synthesiser – this means loading the database, synthesis, creating the annotation file and creating the output sound signal. Final syntheses speech utterance is played together with the face animation of the talking human face.*

Keywords – *Speech synthesis, diphones, mobile phone*

1. INTRODUCTION

The purpose of this paper has been to create an application for reading short messages in cell phones. The application is a combination of the speech synthesis and face animation. Speech part is represented by speech synthesizer and animation is represented by 3D model of human face. After the SMS has been recieved, application starts to synthetise text and the output acustic signal played together with animation.

This document focuses on the speech synthesis part of this project.

2. SPEECH SYNTHESIS AND SYNTHESIZERS

The idea that a machine could generate speech has been in minds of people for some time already, but the realization of such machines has only really been practical only within the last 50 years. The rise of a concatenative synthesis began in the 70s, and has largely become practical as large-scale electronic storage has become cheap and robust. Before 1980, research in speech synthesis was limited to the large laboratories that could afford to invest the time and money for hardware. By the mid-80s, more labs and universities started to join in as the cost of the hardware dropped. By the late eighties, purely software synthesizers became feasible, however the speech quality was still decidedly inhuman. And although we are now at the stage were talking computers are with us, there is still a great deal of work to be done.

Speech synthesis means creating human-like speech using a machine, which is known as speech synthesizer.

There are several types of these synthesizers, but each is made to do the same: to reproduce the given text in the clearest and most understandable manner.

There are four basic approaches:

- Synthesis using units
- Formant synthesis
- Articulation synthesis
- HMM synthesis

On Fig. 1 we can see a block diagram of a general synthesizer. Of course this diagram is simplified to our needs and some elements (such as feedback found in some learning synthesizers, etc.) are omitted.

However, virtually every synthesizer consists of following parts:

- Entry text input, analysis
- Preprocessing
- Synthesis
- Post processing
- The synthesized speech

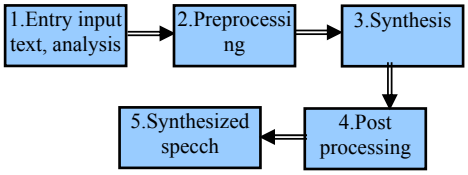

Fig. 1. Block diagram of general synthesizer

In our implementation of speech synthesizer for mobile phone we have decided to use diphone synthesis and naturally diphone database. The major

advantage of this solution is in its size. Slovak language can be sufficiently covered by only 1550 diphones and this makes the size of the solution very reasonable (especially compared to other approaches).

3. DIPHONE SYNTHESIS

A diphone, together with phoneme, is one of major speech elements. It consists of two following phonemes. The boundaries of the diphone are in the middle of these sounds. This means, that a difone length is not double, as one might suspect, but approximately the same as length of one phoneme. The advantage of using difones and not phonemes is that they better represent the change between sounds, because their boundaries are in the middle of sounds where the characteristic time curve is stable.

In the theory, the number of difones is the square of number of phonemes (all combinations of two phonemes is a square). However, the real number is lower, because the particular language does not use, or does not utilize all of them. We can get the real number of diphones by closely studying the language [1].

3.1. Creating database

Before starting the diphone synthesis, the diphone database has to be created. This database consists of real speech recordings which are broken into small parts – diphones. There are two options how to create and record this database. Either to choose words, which will cover all difones from a dictionary, or use some other approach. These words need not to have a meaning, the aim is to have the smallest possible set of recordings.

Of course, the better the recording quality, the better the speech output. Therefore it is advisable to use studio quality recording. Usually, the recording can not be done at once, because the narrator would get tired and the recording quality would deteriorate. Therefore it is important to ensure the same conditions (sound reflections, time of the day, hardware, etc.) during the sessions.

The choice of narrator is also very important. This cannot be just anyone. The best people for this job usually come from TV or radio environment – someone who earns living by speaking.

As already mentioned, the database is not recorded at once, because the narrator would get tired and the recording quality would deteriorate.

The recording has to be further processed to get the final database. It has to be replayed to check for any errors. The next step is a process called labeling. This means that the diphone boundaries have to be set and marked down, otherwise the machine would not know where to find them.

There are several labeling methods. The easiest and slowest one is manual labeling. From automatic or semi automatic methods of labeling we mention labeling based on DTW and acoustic model labeling. The labeling should also include equalizing the recording samples. We do the amplitude equalizing because it is not desirable to have the volume change in the middle of a word – that is what would happen if two difones have different sound level. They have to be attached at the same amplitude level.

Amplitude equalizing is not the only one. There is also phase equalizing, which is even more important. All the boundaries have to be in approximately same phase. If the phase would be opposite, we could be experiencing lot of noise and various acoustic clicks [1].

The results of these steps are an acoustic database – in our case in a WAV file. To the database belongs also an index file. It is actually a list of diphones with its boundaries in WAV file.

3.2. Implementation of diphone synthesizer

The design of the synthesizer is on figure below (Fig. 2) where the principle of the speech synthesis has been described in very simple form how the synthesizer works.

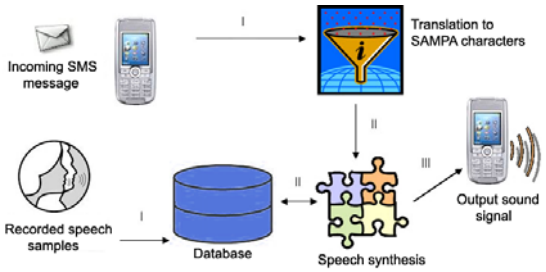

Fig. 2. Synthesizer design

The input text has to be synthesized into speech. But at first it has to be broken down into so called Sampa alphabet. SAMPA (Speech Assessment Methods Phonetic Alphabet) is a phonetic translation which uses only printable ASCII characters. In the first step all characters are retyped to SAMPA. In the second step the result from the first step is retyped according to all rules for pronunciation for Slovak language. These rules were taken from [3]. For example word košeľa will be written as „ko(S)E(L)a“.

Before we get to the synthesis, the database has to be loaded. In our case, this is done right after the start of the application. As the database is in WAV format, the program has to find the data part and save it in the memory. It also loads the index file with the list of phonemes and diphones and its boundaries. In the tables (Table 1 and Table 2) are examples of the part of index file with phonemes and also diphones.

Table 1. Example of index file for phonemes

SAMPA phoneme	Offset	Begin	End	Middle
k	355254	70159	71890	71001
r	246067	49501	51097	50305
a~	277423	51098	55381	53259
(L)	10751	1429	2539	1995
O	202295	33532	34505	34041
U_^	231356	40543	42386	41450

Explanation to the table 1:

- Sampa phoneme – name of phoneme from index file. File contains all phonemes in the same order like they are stored in the database.
- Offset – value represents start of a group of diphones that have the same first phoneme.
- Begin – represents the first sample in database for this phoneme.
- End - represents the last sample in database for this phoneme.
- Middle - represents the middle sample in database for this phoneme.

Table 2. Example of index file for diphones

SAMPA diphone	Begin	End
(J) I	*0*	*1793*
(S) E	*0*	*1812*
I_^a r	*2354*	*4680*
O _	*2326*	*4334*
O x	*4335*	*5771*
U_^ t	*3561*	*4505*

Explanation to the Table 2:

- Sampa diphone - name of diphone from index file. File contains all diphones in the same order like they are stored in database.
- Begin – represents start of diphone in the group of diphones starting with the same phoneme (it's like local number). It means if we want to know the number where the diphone begins in the database (global), we have to add this value and the value of offset from the Table 1. For example the phoneme U_^ starts on a sample number 40543 (Table 1). But if we want to know where the group of diphones is starting with U_^, we have to go to sample number 231356 (offset). Than according second letter in the diphone are samples found. For diphone U_^ t we get values like that: beginning add offset from Table 1 and begin from Table 2(231356+3561 = 234917).
- End - represents where the diphone in the database ends in the group of diphones starting with the same phoneme (it is like a local number). The global value for the end we get in the same way like for the beginning, it means the adding offset from Table 1 and end from Table 2(231356+4505 = 235861).

The algorithm of the synthesis is very simple, because the text is in SAMPA and the database is loaded. For each diphone we check if it is in the database. If it is so, samples are taken from the database and stored. If diphone is not in database, it is replaced with the two phonemes from which diphone is composed. For each phoneme we do not take all samples, just half of them. It is because like we said before, the duration of diphone should be the same like one phoneme. For example, for diphone En (we suppose it is not in the database) samples for E are taken from "middle" to "end", samples for n are taken from "begin" to "middle". We do this for the whole text which should be synthesized. When the program finishes going through the whole text, and audio output is created and written in a WAV file.
The WAV file consists of two basic parts, a header and a data segment. The header consists of so called RIFF descriptor, which contains general file information and a part called FMT. This part information about format of the data part. The format of a WAV file can be seen on Fig. 3 [4].

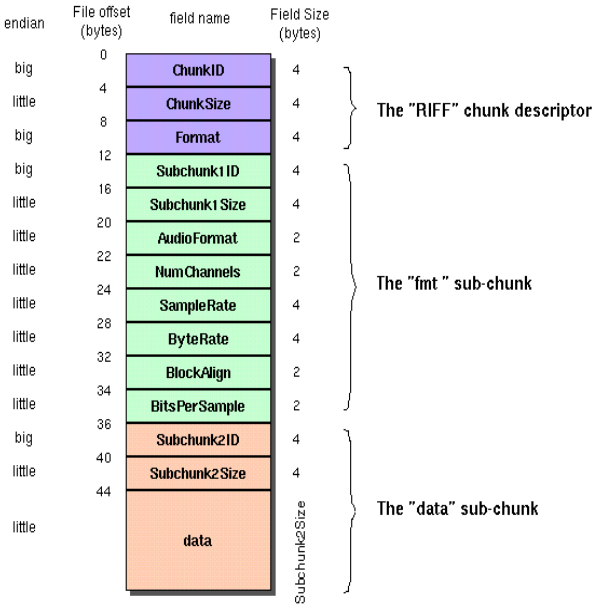

Fig. 3. Wave format

Chunk ID – Contains ASCII characters of RIFF

ChunkSize – Length of file except first two fields (size – 8 bytes)
Format – Contains ASCII characters of WAVE
Subchunk1ID – Contains ASCII characters of fmt
Subchunk1Size – 16 for PCM
Audio Format – 1 for PCM, linear quantization, other values for non linear compression
NumChannels – 1 mono, 2 stereo
Sample rate – sample frequency (Hz) - 16 000
ByteRate - SampleRate*NumChannels*BitsPerSample/8 - 32000
BlockAlign – 2
BitsPerSample – Number of bits per sample - 16
Subchunk2ID – Contains ASCII characters for data
Subchunk2Size – size of sub chunk, Samples*NumChannels*BitsPerSample/8
Data – Values of stored data from -32 768 to 32 767
For our output WAVE file we choused values which are underlined.

After creating output WAVE file, this can be played on mobile phone.

3.3. Synchronization with face animation

The synthesized text has to be synchronized with the speaking face.
We use an annotation file (*.ano) which is a simple text file containing phonemes with its time marks (boundaries). The values are determined individually for each phoneme or phoneme from the index file. From this file we take numbers of samples for the beginning and the end and using the sampling frequency determine the start time and the length of the maximum.
In the table (Table 3) is a sample of this file for word „kráľov“.

Table 3. Example of anotation file

SAMPA	Internal phoneme	Begin [s]	Rise time [s]	Maxim length [s]	Weight
k	k	0.4560	0.0298	0.0298	0.5000
r	r	0.5156	0.0647	0.0647	0.5000
a:	a	0.6451	0.0194	0.0389	0.5000
L	l	0.7034	0.0192	0.0192	0.5000
O	o	0.7417	0.0157	0.0157	0.5000
U_^	u	0.7731	0.0480	0.0480	0.5000

Explanation to the table 3:

- Internal phoneme – represents character which application uses for selection of viseme (shape of mount for face animation). It’s made by conversion from SAMPA using mapping table.
- Begin - time when the viseme starts
- Rise time – time period, when the mouth or the phoneme are open to maximum
- Maximum length: a duration of phoneme’s maximum. Visually, this is the time, when the mouth does not moe (or the movement is not significant)
- Weight: from <0,1>. The higher the weight the more visible the viseme [2].

4. CONCLUSION

In this paper we have presented a Slovak language speech synthesis application for a mobile phone. The application is combined with a speaking face animation and presents a possible future development for future mobile computing.

The type of the created application is Java Midlet, very similar to an applet class, but assigned for cell phones. The application is installed from JAR file. Jar file is an archive and contains compiled classes.

The application can be very easiely modified for needs of a mobile phone. This will be necessary for older types of mobile phones, which have limited memory for applications - in this case the database is loaded like external file, and it is not a part of the JAR file. However, this approach can also be very practical, if there is some change in the database, because it is not necessary to change JAR file.

In the future, with the expansion of the mobile phone memory, the database can be extended. Including more diphones to the database and also some triphones will result in the quality improvement of the output acoustic signal, the speech will be more fluent and human like.

ACKNOWLEDGMENT

This work was supported by Slovak Grant Agency VEGA under grants No. 1/3110/06 and AV 123/06.

REFERENCES

[1] http://festvox.org/bsv/
[2] Vajaš M., Rozinaj G.: Facial Animation During Speech Performance, IASTED International Conference on Communications, Internet and Information Technology (CIIT 2004) November 22-24, 2004, St. Thomas, USA
[3] Cerňak, M.; Rozinaj, G.: Forward Masking Phenomenon in Concatenative Speech Synthesis, 4th EURASIP Conference focused on Video / Image Processing and Multimedia Communications EC-VIP MC’03, July 2-5, 2003, Zagreb, pp.: 691-694
[4] http://ccrma.stanford.edu/courses/422/projects/WaveFormat/