Quadravox Text-to-Speech Synthesis

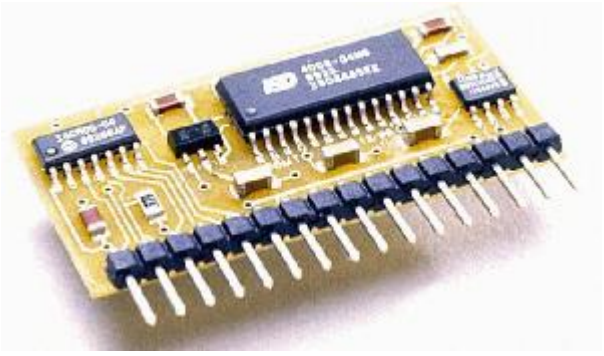
Quadravox, Inc.
1701 N. Greenville Ave, Suite #608
Richardson Tx 75081
Ph:(972) 669 4002
Fax: (972) 437 6382
(www.quadravox.com)

Introduction

The Quadravox Text-to-Speech synthesis system aims at replacing the hard to find GI SP0256 allophone synthesizer, and while we were at it, we tried to make it (more) intelligible (and we threw in a 1W amplifier, in the QV306M1-TTS version).

Our system consists of 2 parts: software and hardware (no surprises so far).

The software (diphones.exe) is a Windows based program that can create an utterance in English from a text input. The result is placed in a wav file that can be copied and used anywhere wav files can be used. The software can be used in "stand alone" form, to create and listen to synthetic wav files, or ,in conjunction with the hardware , to prepare data for a micro-controller application (for instance, a "Basic Stamp™" application).



The hardware is our newest sound module, and comes in two "flavors": QV306M1-TTS (see picture, available December 2001) or QV306M4-TTS (available now) and QV356 (dedicated to robotics applications, available December 2001). The modules present a 3 wire RS232 interface to the world, and contain 2 main chips: a PIC micro-controller, and an ISD chipcorder™. The PIC chip interprets the commands received on the RS232 interface, and causes the ISD chip to play a sequence of pre-recorded sounds (or "diphones", see below).

Speech Synthesis

The hardware we chose for this application all but dictated the method we picked for speech synthesis. It also constrained the overall quality. Since the ISD chipcorders™ can only play prerecorded segments, without any modifications, we chose a very simple variant of the so called “diphone synthesis “ method.

60 Second Speech Synthesis Tutorial

This tutorial would not pass scrutiny by a purist, but I am trying to cut to the chase here. Speech is made up of elementary sounds called “phonemes”. (I can hear the linguists screaming). For example, English has about 3 dozen phonemes (depending on dialect). The list we use in given in Appendix 1. The first stage of a text-to-speech system is a text-to-phonetics program, which transforms natural spelling (and abbreviations, acronyms, etc...) into a sequence of phonemes.
For example, using our phonetic code, the word “hello” has to be transformed into the sequence of phonemes:
_# _H EH _L OW _# (_# is the “silent phoneme”)

Perceptual experiments, starting more than 60 years ago have shown that it is a bad idea to try to “concatenate” isolated phonemes to create speech. What matters most in speech perception is the TRANSITION between phonemes, not so much the phonemes themselves. Consequently, there are two choices left: either to GENERATE the transitions (in a certain parameter domain), or to STORE them. This is , of course, a simplification. A half-way solution consists in using several context dependent variants (“allophones”) for each phoneme. This was the solution adopted by the GI SP0256 chip.
Generating the transitions was in favor until relatively recently, because it required less memory space. However, it also requires a lot of rules, which in turn require a lot of in-depth study of speech sounds. Now that memory is almost free (...), the methods that store the transitions have almost eclipsed all the others.
In the “diphone” method, the basic (and false) hypothesis is that a phoneme can only influence its immediate neighbors. This is not strictly true, but is a good first approximation, as the results will prove. Consequently, we just have to store the transitions between the “stable middle” of a phoneme and the “stable middle” of the next phoneme. The storage requirements increase as the square of the number of phonemes, but this is no longer a problem. Synthesis is then performed by concatenating diphones, in a “domino fashion”: the junction between diphones happens in a supposedly stable portion of the speech sounds, so that there is not too much of a discontinuity at the border. The transitions themselves are “perfect”, since they were actually recorded.
For example, for the word “hello” we will concatenate the following diphones:

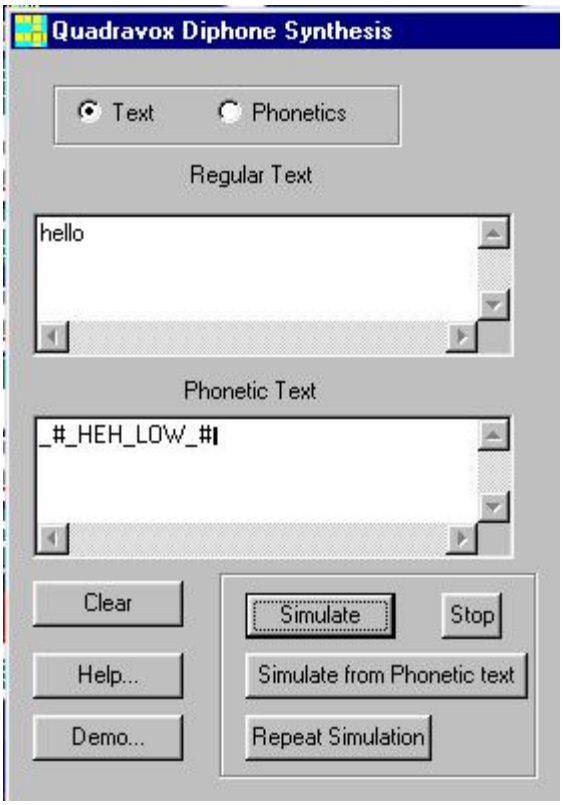
_#:_H , _H:EH, EH:_L ,_L:OW and **OW:_#**

Nowadays, the diphone waveforms are usually stored directly, without compression. The state-of-the-art text-to-speech systems :
(check out: http: //www.research.att.com/ ~mjm/cgi-bin/ttsdemo) , perform some very sophisticated signal processing to modify the prosody (pitch and duration) of the phonetic elements (not necessarily diphones): this way, they can create a natural sounding utterance, where words are stressed on the correct syllable, and questions sound like questions.

We do not have such luxury, since our ISD recorded waveforms cannot be changed in any way with the hardware we have on the module. We still had to take some precautions to avoid waveform discontinuities between diphones, which could cause very audible clicks. Due to all these constraints, our speech quality, though very intelligible in most cases, is also very mechanical and is not well suited for long sentences.

Quadravox Diphone Synthesis

Our software synthesizer (diphones.exe, downloadable from our web site at www.quadravox.com) comprises several stages. (For a detailed description, see the help file after you download and install the software)



The first stage translates regular spelling into phonemes. We borrowed and modified for our purposes a "public domain" program written by Tom Jennings (1999). His purpose was to create input data for the GI SP0256 chip. This program is far from perfect, but it is a good start, since the user also has access to the input to the second stage.
The results of the first stage (string of phonemes) are put in a "phonetic window" on the screen. The user can directly modify the phonetic transcription if it proves unsatisfactory, upon listening to the synthesized utterance. The second stage is the concatenation of diphones, based on the phonetic transcription in the phonetic window. Clicking on "Simulate" automatically calls stage 1 followed by staged 2 and produces the synthetic output. Clicking on "Simulate from Phonetic text" calls only stage 2 before producing the synthetic output.

Every time an utterance is synthesized, the associated string of phonemes is appended to an ASCII log file. This way, the user can save the data that will be necessary to control an

eventual micro-controller application using the Quadravox hardware (instead of the PC sound card) for synthesis.

The size of our software is about 5 Mbytes. (most of it is occupied by the diphone dictionary). We have exactly 1,200 diphones in our diphone dictionary (though we recorded about 1,600). This is due to the addressing constraints of the ISD chip we use for synthesis. Consequently, certain transitions are not present in our dictionary: we have to fake them by inserting a "phantom" silence in between the two phonemes for which we don't have a diphone. We of course tried to keep the most common transitions in our dictionary, so that the extra silence does not happen often.
The diphone dictionary file (dicodiph.sig), which comes with our software, is also recorded on the ISD chip on our module. The PIC chip on our module simply makes the ISD chip play back a sequence of short segments (diphones) in rapid sequence. This is the exact equivalent of our software simulation, which concatenates those same segments and stores them in a wav file.

Products, prices, etc...

Our software is freely available for download from our Web site.

For prices and availability of the various modules, please go to our Web site at
www.quadravox.com

Appendix 1

Our list of English phonemes, and their 2 letter codes (the letters that correspond to the phoneme are indicated in **bold**):

Vowels:

IY	b**ee**t	IH	b**i**t	EY	b**ai**t	EH	b**e**t	AE	**a**t	UY	b**oo**t
UH	b**oo**k	OW	b**oa**t	AO	c**au**se	AA	c**o**t	AX	**a**head	AH	b**u**t
AY	b**i**te	OY	b**oy**	AW	h**ow**	ER	ch**ur**ch	OR	c**ore**	AR	p**ar**k

Consonants:

_P	**p**et	_T	**t**op	_K	**k**eep	_B	**b**et	_D	**d**ot	_G	**g**o
_M	**m**y	_N	**n**ot	NG	si**ng**	_F	**f**it	_V	**v**ote	TH	**th**in
DH	**th**en	_S	**s**it	_Z	**z**ip	SH	**sh**in	ZH	rou**ge**	CH	ou**ch**
_J	**j**ar	_H	**h**ot	_R	**r**ed	_L	**l**ip	_W	**w**att	_Y	**y**et

Miscellaneous:

_#	silence