HIGH QUALITY SPLIT BAND LPC VOCODER AND ITS FIXED POINT REAL TIME IMPLEMENTATION

S Villette, M Stefanovic, I.Atkinson, AM Kondoz

Centre for Communication Systems Research
University of Surrey, Guildford, Surrey, UK.
a.kondoz@ee.surrey.ac.uk

ABSTRACT

A split band vocoder in which the LP excitation is split into voiced and unvoiced frequencies is presented. In doing this the coder's performance during both mixed voicing and speech containing acoustic noise is greatly improved, producing soft natural sounding speech. In addition a variable rate version which achieves an average rate of 1.4 kb/s is detailed. The issue of fixed point real time implementation of this coder is also presented.

1. INTRODUCTION

Many LPC based vocoders operating at bit rates of 3.0 kbits/sec and below employ simple excitation sources consisting of quasi-periodic impulse trains during voiced speech and white gaussian noise during unvoiced speech. Whilst this may be sufficient for maintaining speech intelligibility, the synthetic speech often sounds robotic and speaker identity is lost. If the speech quality could be improved without significantly increasing the bit rate then many new applications would open up, spawning new products such as voice pagers and internet telephony. Very low bit rate speech coders are also in demand as companions to video compression algorithms for future mobile videophone systems.

The Split-Band LPC Vocoder uses vector quantisation techniques to efficiently encode the LPC parameters in the LSF domain, freeing bits which may then be used to encode additional information about the excitation in order to improve the speech quality.

2. ENCODER

DC rejected, high frequency pre-emphasised speech is processed in 20ms frames. LPC parameters are determined using a 10th order Durbin's algorithm which are then quantised in the LSF domain. The quantised LPC parameters are then used to find the LPC residual required for determination of the excitation harmonic amplitudes. Both the speech signal and the LPC excitation are transformed into the frequency domain using a 512 point FFT, (note that these two real FFTs may be calculated using one complex FFT in order to reduce complexity).

A robust initial pitch analysis is first performed [1] followed by a pitch refinement process and a binary voicing decision for each pair of harmonics in the speech spectrum. Finally the harmonic amplitudes are determined from the excitation spectrum. The LSF, pitch, voicing and excitation parameters are quantised and transmitted to the decoder. The overall bit allocation is presented in Table 1.

3. DECODER

Once the parameters have been decoded, the harmonic excitation amplitudes are modified to reduce noise in the LPC valleys, thereby perceptually improving the coder performance.

The synthetic excitation is formed by adding the unvoiced and voiced generator outputs. The unvoiced generator is performed using FFT filtering, i.e. spectrally shaping a random noise source in the frequency domain according to the voicing and harmonic amplitude information, then transforming back to the time domain. The voiced excitation is generated by summing up each sinusoidal harmonic, scaled by the decoded harmonic amplitude.

Finally the overall spectral shaping is added to the excitation using an LPC synthesis filter whose coefficients are linearly interpolated every 5ms.

Parameter	Version	
	2.5kbits/sec	**2.7kbits/sec**
LSF	28	28
Voicing Freq.	3	3
Pitch	7	7
Energy	6	6
1st 8 Harmonics	6	6
Excitation Shaping	-	4
Total:	50	54

Table 1: Bit Allocation Schemes

4. FIXED RATE CODER PERFORMANCE

Listening tests were performed on the high-level simulation outputs, comparing the Split-Band LPC Vocoder against DoD LPC10e [2], IMBE and DoD 1016 CELP [3]. Individual tests were performed on male and female speech using twenty subjects. The results are presented in Table 2. Although the scores are generally lower than expected due to the unfamiliarity of many of the subjects with low bit rate coders, the results clearly indicate the preference of the Split-Band Vocoder over

LPC10e, IMBE and 1016 CELP at both the 2.5 and 2.7kbits/sec.
Finally the coder was tested on speech containing acoustic noise (vehicle, gaussian, multiple talker). Although the background noise was modified by the coder, the speech retained its intelligibility and talker identity. Very little degradation was observed in the case of multiple talkers.

	Mean Opinion Score		
	Male	**Female**	**Overall**
LPC10 2.4kbs	1.3	1.3	1.3
IMBE 4.15kbs	3.4	3.0	3.2
1016 CELP 4.8kbs	3.2	3.0	3.1
Split Band Vocoder 2.4kbs	3.4	3.4	3.4
Split Band Vocoder 2.7kbs	3.5	3.7	3.6

Table 2: Split Band Vocoder :Listening Test Results

5. VARIABLE QUANTISATION SCHEME

A variable rate version has also been implemented on the structure of the Split-Band LP Vocoder discussed above. The quality of the fixed rate system was used as a benchmark during the design and testing of the variable rate version.
In the new variable rate scheme, the speech frames are classified into one of four categories (silence, unvoiced, mixed voicing and fully voiced). This is indicated to the decoder using 2 bits of header information.
Tests showed that during voiced and mixed speech frames, the pitch could not be encoded using fewer than 7 bits without incurring perceptual distortion. However, during unvoiced and silence frames, encoding the pitch information was unnecessary, which resulted in a saving of 7 bits per frame for quantising these frames.
Subjective tests revealed that during unvoiced frames, the order of the LPC model could be reduced from tenth to sixth order with little effect on speech quality. During transition between sixth and tenth order LP models at speech onsets and offsets, it is necessary to deactivate the LSF interpolation at the decoder. Tests showed that removal of the LSF interpolation during transitions did not cause degradation of the synthetic speech quality.
Encoding of the voicing frequency is only required during mixed voicing frames, since in other three categories, the voicing frequency can be inferred from the header information. During silence, the comfort noise generator is activated at the decoder. The overall variable bit allocation scheme is shown in Table 3.

	Silence	UV	Mixed	Voiced
Header	2	2	2	2
Pitch	0	0	7	7
VUV	0	0	3	0
LSF	0	21	26	26
Energy	0	6	6	6
Env	0	0	6	6
Total	2	29	50	47

Table 3: Bit Allocation for 4 Different Speech Classes

The quantisation rate for these four classes ranges from 0.025 kb/s during silence to 2.5 kb/s for mixed voicing frames. Informal listening test results, presented in Table 4, indicate that no noticable perceptual distortion arises when the variable bit rate quantisation scheme is introduced into the Split-Band LP vocoder, however the average bit rate was reduced to 1.4 kb/s.

	Variable Rate as compared to Fixed Rate
Better	3%
Slightly Better	24%
Same	38%
Slightly Worse	29%
Worse	6%

Table 4: Results of Comparative Listening Tests

In addition to the variable bit rate, a variable frame length scheme was introduced. It gives an option of varying the size of the processed speech frames in the range of 10 ms to 40 ms. It produces a great saving in number of processed frames for both unvoiced regions and steady voiced sections of speech. The speech segmentation is also significantly improved, enhancing the quality, especially in the transition regions.

6. FIXED POINT IMPLEMENTATION

The fixed rate coder presented in this paper has been designed to have complexity requirements within the limits of a single chip DSP. Moreover the quantisation scheme calls for low memory requirements, and hence the DSP's on-chip RAM and ROM should be sufficient, without any need for extra external memory. So this coder opens for a single chip real time DSP implementation. Only the fixed rate version of the coder has been used for this purpose.

6.1. CHOICE OF THE DSP

The initial complexity estimation implied a DSP capable of handling at least 20 Mips. The coder could be implemented using either a 16 bits fixed point DSP or a 32 bits floating point device, both being able to provide this computational power.
Fixed point DSP needs less transistors than floating point ones for equal functionality. Hence they require less power than their floating point equivalent, and need a smaller silicon area, which has a direct impact on the final cost. Finally they allow the developer to decide for each variable whether it should be in simple or double 1precision. Each data can then occupy 16 bits, or 32 bits when required, whereas in a floating point 32 bits device each variable uses 32 bits. This shows that fixed point devices can use smaller memory banks, hence reducing cost and power consumption further. This is of utmost importance for handset applications, where available power is very restricted.
We chose to use a Texas Instrument TMS320C541. It comes with 28K words ROM and 5K words RAM on-

chip for about 20$ and is said to be the best solution in terms of power consumption as well [4]. It is able to handle 50 MIPS, which far exceeds our requirements.

6.2. HIGH LEVEL FIXED POINT SIMULATION

Fixed point DSPs are more difficult to program than their floating point counterparts. An example is given by equation (1), where an autocorrelation is computed.

$$R(0) = \sum_{i=0}^{160} p(i) * p(i) \qquad (1)$$

Example of autocorrelation

The data is represented using 16 bits. The square of $p(i)$ needs 31 bits to be represented, in two's complement. The sum of 160 of these products needs 39 bits to be represented, which is small enough for the 40 bits accumulators of the C54. The problem is how to store this result and use it in further calculations. Keeping the 39 bits is not a viable solution, as it would lead to an increase in the number of bits stored after every multiply.

Only a limited number of bits of the result are relevant, as the data $p(i)$ will not be at its maximum for every i, implying that the sum will never need the full dynamic of the 39 bits. Only a careful study can tell what the real range of the sum is, so that bits which are always equal to zero are not stored in memory.

Such a study is done using a high level C language simulation, in which each data is represented using 16, 32 or 40 bits as in the DSP, and dedicated functions are used to simulate the behaviour of the DSP. Each floating point part is replaced by its fixed point equivalent to check for any loss of quality induced by the reduced precision of the fixed point arithmetic.

The simulation also allowed several algorithms to be investigated for square root calculation. The usual way of dealing with square roots is to use a Newton-Raphson method. As there is no dedicated divider in the DSP, a single precision division requires about 20 times more cycles than a multiplication. Hence it is important to limit the use of divisions. Performing the Newton-Raphson method on the inverse of the square root, as shown by equation 2, makes it possible to use only multiplications and additions to get the square root, which greatly reduces the complexity. Only 5 iterations are usually needed.

$$R[i] = R[i-1] * (1.5 - (N/2) * R[i-1]^2) \qquad (2)$$

Efficient square root calculation

N is the normalised operand, R[I] is the root at iteration i.

As square roots are still time demanding, their use should be avoided whenever possible. To compute the magnitude of a vector, it is possible to avoid using square roots by using the technique described in diagram 1.

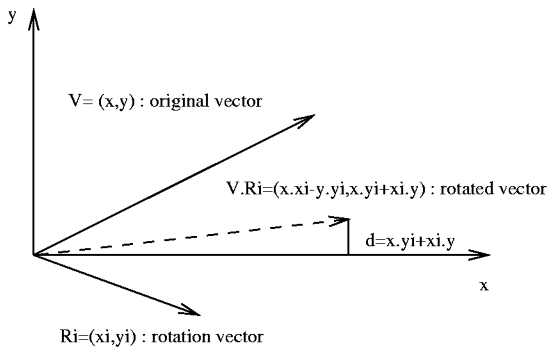
y
V= (x,y) : original vector
V.Ri=(x.xi-y.yi,x.yi+xi.y) : rotated vector
d=x.yi+xi.y
x
Ri=(xi,yi) : rotation vector

Diagram 1: Efficient magnitude calculation

The vector is rotated by multiplying it by vectors of magnitude one, until it is close enough to an axis. Then the bigger coordinate of the rotated vector is taken as its magnitude. A binary search using 16 different rotation vectors between 0 and 45 degrees gives the best angle, hence the magnitude, in only 5 stages each consisting of 2 multiplication and one addition. The accuracy is about 10 bits out of 16, which is usually enough, and the number of cycles is greatly reduced.

6.3. RESULTS OF THE C SIMULATION

The whole algorithm (both encoder and decoder) has been converted to fixed point and compared to the original floating point program. Although the use of double precision has been restricted to a very few areas, the quality is equivalent to the original floating point. It is very difficult to distinguish between the two versions.

An estimation of the complexity has been done by counting precisely the instructions needed in the program, which makes it possible to estimate the final number of cycles on the DSP.

6.4. DSP IMPLEMENTATION

Most of the encoder has been implemented on the TMS320C54, using the TI evaluation board and simulator. The coder is written in assembly language, as C compilers for DSPs do not provide optimised enough code. This has shown that the C fixed point simulation was very accurate, as every memory location in the DSP is exactly what was predicted by the simulation. It implies that the final quality of the coded speech will be exactly what has been predicted by the simulation.

The other issue is the fact that the implementation has to run in real time, with reasonable memory requirements. Only the major part of the encoder has been implemented on the DSP at the moment. But by comparing the effective number of cycles on the DSP with the estimation mentioned earlier, it is possible to have a very good estimation of the final number of cycles required for the whole encoder/decoder. This is shown in table 5.

At 50 MHz with 20 ms frames, 1 million cycles are available on the DSP. It is enough for both the encoder and the decoder, allowing full duplex operation on a single DSP chip. As the code does not yet fully use the parallel instruction capacities of the DSP, a gain of up to

50% can still be obtained upon these figures. This will give spare computational power left for other applications, which can be useful for use in handsets for example.

Function	nb of cycles on the C54
Encoder and Decoder in Full Duplex Operation	1,000,000
Encoder Only	700,000 (*)
Decoder Only	290,000 (*)
Input High Pass Filter	10,175
Pitch Detector	250,000
Pitch Refinement	200,000 (*)
Voicing Decision	20,000 (*)
Calculation of Excitation Amplitudes	20,000 (*)
Durbin LP Calculation	19,220
LP to LSF Transformation	50,803
LSF to LP Transformation	6,602
LSF Quantisation	50,000 (*)
LP Inverse Filter	9,780
FFT Transforms	50,360

Table 5: Complexity estimation on the C54
(*) means these are estimations on still uncoded parts

The memory requirements of the coder have been kept as low as possible, to keep costs down. The memory needed can be classified in 4 types:

1) ROM
   - Program itself
   - Tables (quantisers, sine table for FFTs, windows, specific tables to increase the speed of time-critical parts)
2) RAM
   - Data which have to be kept throughout the program
   - Temporary data which are used only in a small part of the program.

The last type will only require the maximum space needed at the same time by one routine, as this space can be overwritten by other routines when required. The estimation for the encoder given in table 6, is based on the parts which have already been encoded. Most of the required memory is for the pitch algorithm, (which makes extensive use of tables to reduce execution time) so the requirements for the decoder should be much lower than for the encoder.

The total of 6631 words of ROM (out of 28K) and 2905 words of RAM (out of 5K) allows the encoder to fit in the DSP without requiring external memory, and leaves enough space for the decoder to fit as well.

As a conclusion, a single chip fixed point DSP without external memory is all that is required to run the coder in real-time, keeping the different costs (both power consumption and price of the hardware) as low as they can possibly be.

	ROM		RAM	
Function	Prog.	Tables	Perm.	Temp.
Encoder Only	3531	3100	2185	800
General definitions	0	0	44	0
Input Buffer	20	0	512	160
Input High Pass Filter	97	0	24	648
Pitch Detector	895	1548	437	800
Pitch Refinement	300(*)	0	532(*)	530(*)
Voicing Decision	300(*)	0	30(*)	10(*)
Calculation of Excitation Amplitudes	200(*)		10(*)	512(*)
Durbin LP Calculation	275	80	23	332
LP to LSF Transformation	800	0(**)	26	190
LSF to LP Transformation	190	0	23	166
LSF Quantisation	200(*)	448	10(*)	10(*)
LP Inverse Filter	44	0	12	401
FFT Transforms	210	1024	512	512

Table 6: Memory requirements on the C54
(*) means non yet implemented parts
(**) means the sine tables of the FFTs are used there, hence there is no need for an extra table

7. CONCLUSION

A new LPC based vocoder was presented which has been shown to produce higher quality at a bit rate of 2.5kbits/sec than both the IMBE coder operating 4.15kbits/sec and the DoD 1016 CELP coder operating at 4.8kbits/sec. By splitting the speech into voiced and unvoiced bands, both mixed voicing speech and speech containing acoustic noise could be reliably coded without introducing excessive “buzziness” into the synthetic speech. A variable rate version of the coder has also been simulated which achieved an average rate of 1.4 kb/s with no reduction in speech quality.

Real-time implementation of the fixed rate version has also been worked on. Although we have not yet completed this task our estimates show that the encoder/decoder combination can run on a single TMS320C54 with no additional RAM or ROM. This makes the coder very attractive for very low bit rate low power applications.

REFERENCES

1 I. Atkinson, S.Yeldener, A.Kondoz, “High Quality Split-Band LPC Vocoder Operating at Low Bit Rates” ICCASP 97 Proceedings, Volume 2, pp 1559

2 T. Tremain, “The Government Standard Linear Predictive Coding Algorithm (LPC-10)”, Speech Technology, Vol. 1(2), pp 40-49, April 1982.

3. J. P. Campbell, T. Tremain, V. C. Welsh, “The DoD 4.8kbps Standard (Proposed Federal Standard 1016)”, Speech Technology, Vol. 1(2), pp 58-60, April 1990.

4 “TMS320C54x User’s Guide”, pp 4-2