EURASIP Journal on Applied Signal Processing 2004:8, 1107–1112

A Fast LSF Search Algorithm Based on Interframe Correlation in G.723.1

Sameer A. Kibey
Digital Signal Processing and Multimedia Group, Tata Elxsi Ltd., Whitefield Road, Hoody, Bangalore 560048, India
Email: sameer@tataelxsi.co.in

Jaydeep P. Kulkarni
Centre for Electronics Design and Technology, Indian Institute of Science, Bangalore 560012, India
Email: kjaydeep@cedt.iisc.ernet.in

Piyush D. Sarode
Honeywell Technology Solutions Labs Pvt. Ltd., Bangalore 560076, India
Email: piyush.sarode@honeywell.com

Received 16 December 2002; Revised 15 October 2003; Recommended for Publication by Ulrich Heute

We explain a time complexity reduction algorithm that improves the line spectral frequencies (LSF) search procedure on the unit circle for low bit rate speech codecs. The algorithm is based on strong interframe correlation exhibited by LSFs. The fixed point C code of ITU-T Recommendation G.723.1, which uses the "real root algorithm" was modified and the results were verified on ARM-7TDMI general purpose RISC processor. The algorithm works for all test vectors provided by International Telecommunications Union-Telecommunication (ITU-T) as well as real speech. The average time reduction in the search computation was found to be approximately 20%.

Keywords and phrases: line spectral frequencies, linear predictive coding, unit circle, interframe correlation, G.723.1.

1. INTRODUCTION

The underlying assumption in most speech processing schemes including speech coding is the short-time stationarity of the speech signal [1]. Based on this assumption, the input speech is divided into frames of size 20–30 ms (typically) and each frame is processed to give a set of parameters which are defined by the *source-filter model* of speech production [2]. The encoding of these parameters requires lesser bits than the conventional waveform coders [2].

In this model, the combined effects of the glottis, the vocal tract, and the radiation of the lips are represented by a time-varying digital filter. The driving input (or the excitation) to the filter is modeled as either an impulse train (for voiced speech) or random noise (for unvoiced speech). In order to obtain the speech parameters, the principle of linear prediction is employed [1, 2]. By minimizing the mean squared error between the actual speech samples and the linearly predicted ones over a finite interval, a unique set of predictor coefficients can be determined. The transfer function of the time-varying filter is of the form

$$H(z) = \frac{G}{1 + \sum_{k=1}^{p} \alpha_k z^{-k}}. \tag{1}$$

Here G is the gain parameter, p is the order (typically 10) of the predictor, and α_k are the coefficients of this filter. The recursive Levinson-Durbin algorithm is generally used to obtain the optimum estimates of α_k coefficients in the least mean squared error sense [1, 2]. These coefficients contain the formant information and hence are very important parameters.

However, for the purpose of quantization, the predictor coefficients α_k, also known as linear predictive coding (LPC) parameters, are converted into a set of numbers called as line spectral frequencies (LSFs), originally proposed by Itakura [3] as an alternative representation of the LPC coefficients. To obtain the corresponding LSFs, the LPC coefficients have to be mapped on to the unit circle in the z-domain.

Different methods for the LPC to LSF conversion have been discussed in the literature [4, 5, 6, 7, 8]. The method proposed by Soong and Juang [4] estimates LSF frequencies by transforming the characteristic polynomials into sum of cosine functions. This method, however, requires large evaluation of trigonometric functions. Kabal and Ramachandran [5] used Chebyshev polynomials to develop a similar but more efficient transformation. Their method was improved by Wu and Chen [7] using a new decimation-in-degree

algorithm. Rothweiler [9] further suggested computational complexity reductions in the method given by [7]. Also, a new method was proposed by Grassi et al. [6], which computes distinct intervals, each containing only one odd and one even-indexed LSF, thus avoiding the zero crossing search. Another approach to compute LSFs based on split Levinson algorithm has been discussed by Saoudi and Boucher [8].

The ITU-T Recommendation G.723.1, however, uses the real root algorithm to compute the LSFs [2, 10]. In this paper, we explain an algorithm for faster conversion from LPC parameters to LSFs in the real root algorithm framework. It is based on the interframe correlation property of LSF parameters.

The rest of the paper is organized as follows. In Section 2, a brief review of LSFs is given and the conventional real root algorithm for LSF search is explained. The next section describes the search procedure used in ITU-T Recommendation G.723.1, which is to be optimized using the proposed algorithm. In Section 4, the algorithm for faster LSF search is explained in detail. The performance evaluation for the algorithm is provided in Section 5. Finally, the concluding remarks are made in Section 6.

2. LINE SPECTRUM FREQUENCIES

A brief review of LSFs and some of the important properties are provided in this section.

The filter $H(z)$ is stable if it exhibits the minimum-phase property, that is, if all the roots of (1) are within the unit circle. If α_k are quantized directly, small changes in any of the coefficients can produce roots outside the unit circle and result in the instability of the reconstruction filter in the receiver [2]. Hence LPC coefficients are converted to LSFs, which are then quantized. A change in one LSF changes the response only in the vicinity of that frequency. In addition, they can be quantized according to auditory perception, that is, low frequencies can be more finely quantized than high frequencies, since they have a larger effect on the quality of the synthesized speech.

From the previous section, the transfer function of the all-pole digital filter for speech synthesis is given by

$$H(z) = \frac{G}{A_p(z)}, \tag{2}$$

where

$$A_p(z) = 1 + \sum_{k=1}^{P} \alpha_k z^{-k}. \tag{3}$$

To derive the LSFs, $A_p(z)$ is used to compose two transfer functions $P_{p+1}(z)$ and $Q_{p+1}(z)$, called the "sum" and "difference" polynomials, respectively,

$$\begin{aligned} P_{p+1} &= A_p(z) + z^{-(p+1)} A_p(z^{-1}), \\ Q_{p+1} &= A_p(z) - z^{-(p+1)} A_p(z^{-1}). \end{aligned} \tag{4}$$

It follows that

$$A_p(z) = \frac{P_{p+1}(z) + Q_{p+1}(z)}{2}. \tag{5}$$

Both these polynomials are of order $(p + 1)$. However, for an even value of p, the polynomials contain trivial zeros at $z = -1$ (corresponding to sum polynomial) and at $z = 1$ (corresponding to difference polynomial). These roots can be ignored and are removed as follows:

$$\begin{aligned} P'(z) &= \frac{P_{p+1}(z)}{(1+z)} = a_0 z^p + a_1 z^{p-1} + \cdots + a_p, \\ Q'(z) &= \frac{Q_{p+1}(z)}{(1-z)} = b_0 z^p + b_1 z^{p-1} + \cdots + b_p. \end{aligned} \tag{6}$$

The roots of $P'(z)$ and $Q'(z)$ lie on the unit circle and are known as LSFs.

The properties of LSFs are as follows [2, 4].

(1) All LSFs lie on the unit circle in the Z plane.
(2) The roots of $P'(z)$ and $Q'(z)$ alternate with each other on the unit circle.
(3) Minimum phase property of $A(z)$ can be easily preserved if the first two properties remain intact after quantization.

2.1. Real root method to find LSFs [2, 10]

This section describes how ITU-T Recommendation G.723.1 converts the LPC parameters to the LSFs [10].

From (4), it is clear that $P_{p+1}(z)$ is a symmetric polynomial and $Q_{p+1}(z)$ is an antisymmetric polynomial. The polynomials $P'(z)$ and $Q'(z)$, derived from $P_{p+1}(z)$ and $Q_{p+1}(z)$ are *symmetrical* [6] and so the following symmetry property holds true for an even value of p:

$$a_n = a_{(p-n)}, \quad 0 \le n \le \frac{p}{2}. \tag{7}$$

Hence, the order of (6) can be reduced to $p/2$ [2]. This is indicated in the following equations:

$$\begin{aligned} P'(z) &= a_0 z^p + a_1 z^{p-1} + \cdots + a_1 z^1 + a_0 \\ &= z^{p/2}\left[a_0\left(z^{p/2} + z^{-p/2}\right) + a_1\left(z^{(p/2-1)} + z^{-(p/2-1)}\right)\right. \\ &\qquad \left. + \cdots + a_{p/2}\right]. \end{aligned} \tag{8}$$

Similarly,

$$\begin{aligned} Q'(z) &= b_0 z^p + b_1 z^{p-1} + \cdots + b_1 z^1 + b_0 \\ &= z^{p/2}\left[b_0\left(z^{p/2} + z^{-p/2}\right) + b_1\left(z^{(p/2-1)} + z^{-(p/2-1)}\right)\right. \\ &\qquad \left. + \cdots + b_{p/2}\right]. \end{aligned} \tag{9}$$

As all the roots are on the unit circle, we can evaluate these two equations on the unit circle directly.

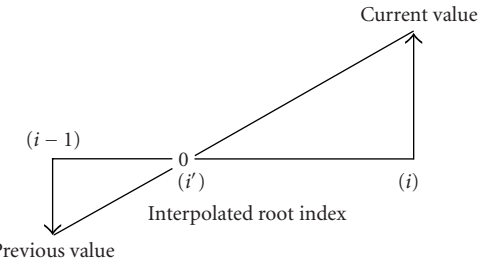

Figure 1: First-order interpolation to find LSF root.

Putting $z = e^{j\omega}$ then $z^1 + z^{-1} = 2\cos(\omega)$, we have

$$\begin{aligned} P'(e^{jw}) &= 2e^{jp\omega/2}\left[a_0\cos\left(\frac{p}{2}\omega\right) + a_1\cos\left(\frac{p-2}{2}\omega\right) + \cdots + \frac{1}{2}a_{p/2}\right], \\ Q'(e^{jw}) &= 2e^{jp\omega/2}\left[b_0\cos\left(\frac{p}{2}\omega\right) + b_1\cos\left(\frac{p-2}{2}\omega\right) + \cdots + \frac{1}{2}b_{p/2}\right]. \end{aligned} \tag{10}$$

These two equations have to be solved to give the LSFs.

3. SEARCH PROCEDURE USED IN G.723.1 [10]

In G.723.1, input speech is divided into frames of 240 samples each (30 milliseconds at sampling frequency of 8 kHz). Each frame is further subdivided into 4 subframes, each of 60 samples. The LPC analysis is then performed on a subframe basis [10]. Since the predictor order is 10, these 10 LPC coefficients are to be transformed into the corresponding 10 LSFs. This transformation is done once per frame, for the last subframe only. The LSFs of the remaining 3 subframes are obtained by performing linear interpolation between the LSF vectors of current and the previous frame.

The transform algorithm first generates sum and difference polynomials from the LPC coefficients. The unit circle is then divided into 512 equal intervals, each of length $\pi/256$ (which corresponds to intervals of approximately 16 Hz at 8 kHz sampling frequency). The sum and difference polynomials are evaluated along the unit circle from 0 to π to search for the roots, that is, the LSFs.

Intervals where a sign change occurs are linearly interpolated to find the zeros of the polynomials. If the sign change occurs between interval number i and $i-1$, a first-order interpolation is performed as follows [10],

$$i' = \left(i - 1 + \frac{\text{Abs\_Prev\_Value}}{\text{Abs\_ Prev\_Value} + \text{Abs\_Curr\_Value}}\right), \tag{11}$$

where i' is the interpolated root *index*, Abs_Prev_Value is the absolute magnitude of the result of polynomial evaluation at interval number $i-1$, and Abs_Curr_Value is the absolute magnitude of the result of polynomial evaluation at interval number i. Figure 1 indicates the location of root index (i') obtained by linear interpolation.

It should be noted that the true LSF value can be obtained as follows

$$\text{True\_LSF\_value} = i' \times \frac{\pi}{256}. \tag{12}$$

While checking for sign change, that is, zero crossings, the interlacing property of LSFs is used. Since the zeros of $P'(z)$ and $Q'(z)$ alternate, only one of them needs to be evaluated at any given step. For the same reason, once a root for a polynomial has been located, the search for the next root is performed by evaluating the other polynomial, starting from the current root. In this way, the region from 0 to π is searched sequentially and the 10 LSFs are located one by one.

4. FASTER SEARCH ALGORITHM

The study of LSF vectors indicates that there is a strong correlation between the LSFs of successive frames and that the change from one LSF vector to another is not too abrupt in general, as observed by Kondoz [2]. Thus, using the previous values as the starting estimates to locate the roots, the number of computations required for each root can be reduced considerably.

Figure 2 shows the distribution plots of the difference between LSF values for successive frames. (Note that the LSF value here means the *interval number* in which the root was located.) A sample speech file containing different male and female voices of total length 7.5 minutes, that is, about 15000 frames, is considered for this experiment. For each frame, the difference between the current LSF value and the previous frame's LSF value is computed. This is done for all the 10 LSFs and the plots in Figure 2 are generated.

From these plots, it can be seen that the average difference is highly concentrated between −10 to +10. Hence, instead of using previous frame's LSF as a starting point directly, we can use a range of values centered around the previous root as the initial search interval. However, if the range is too large, a higher-order root may be falsely detected. To prevent this during the narrowed search, the optimum range of the search interval was chosen as −3 to +3 of the previous root.

If the current root happens to be in this narrowed search interval, then a zero crossing occurs and hence a sign change is detected. Thus, the root is said to be located in that interval. The algorithm then starts searching for the next LSF by evaluating the other polynomial in the appropriate $[i-3, i+3]$ interval.

However, if the root is not present in the initial search interval, no sign change is encountered. In this case, the root is found using the normal G.723.1 procedure. The search now begins from the location of the previous LSF in the current frame and continues till the root is found. The narrowed initial search interval is, however, skipped in this second step as it has already been searched in the first step.

4.1. Explanation for choice of search interval

If the initial search interval is too large, then in some cases a higher-order LSF would be wrongly detected as the current root, since it is also a root of the same polynomial. Also, if

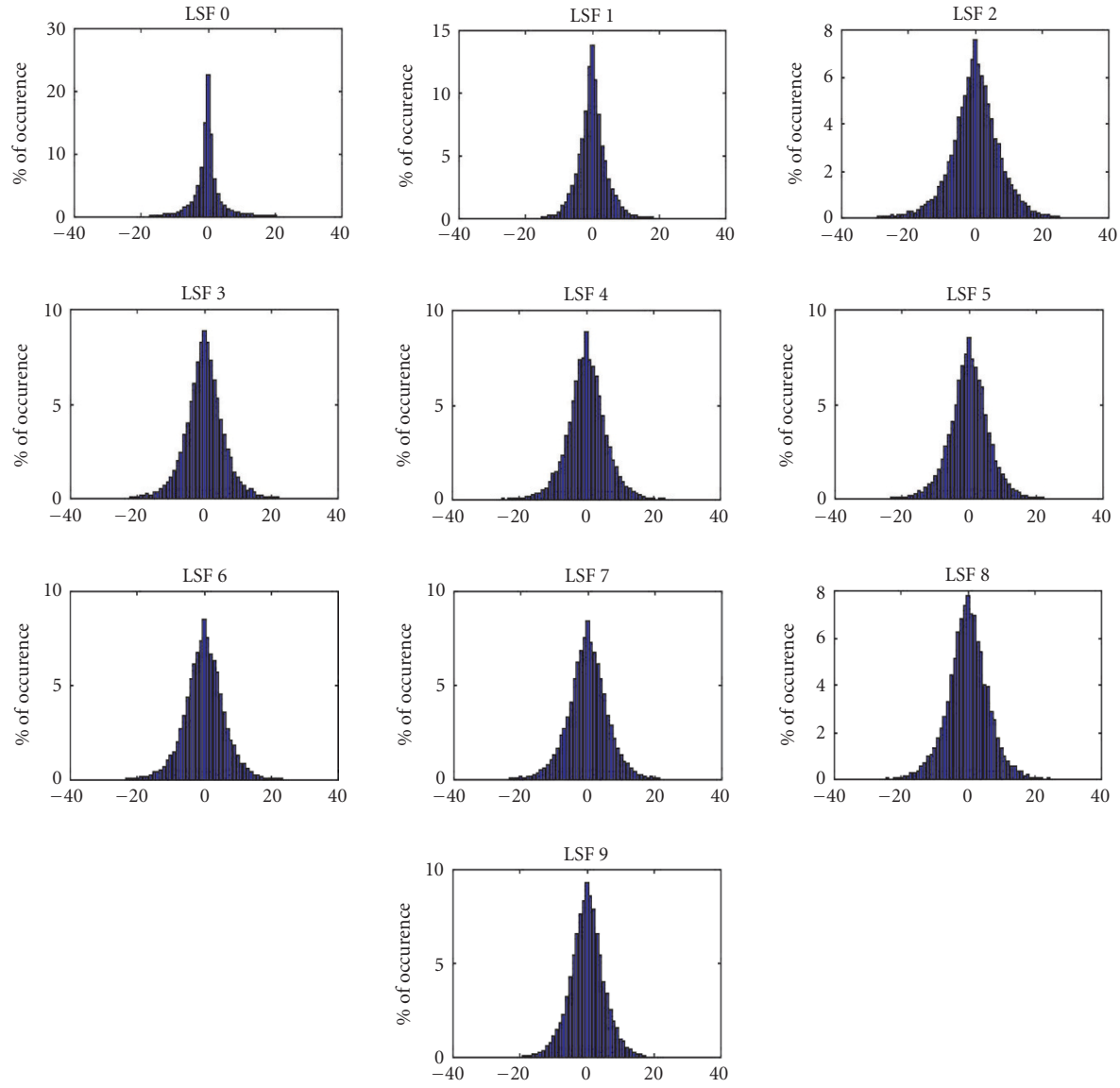

Figure 2: Distribution plots for 10 LSFs (x-axis is the difference between current and previous frame's LSF interval number).

the search region is too small, the search would be unsuccessful most of the times. Thus, an optimum value of the search range needs to be chosen.

As mentioned earlier, this value is found to be from +3 to −3 of the previous frame's root. Separation between adjacent i's is 16 Hz (see Section 3), which implies an interval of about $16 \times 3 \approx 50$ Hz on either side of the center value. Since theoretically the minimum separation between adjacent LSFs is typically 40 Hz [2], the difference between alternate roots (about 80 Hz) exceeds the search range. This prevents the incorrect detection of a higher-order root.

4.2. Corrective measure

Though the possibility of a higher-order root occurring in the range [+3, −3] is very small, it cannot be completely ignored. In that case, the algorithm would fail and the result would not be G.723.1-compliant. Hence, a corrective measure must be adopted. This can be done as follows.

We say the LSF 8 is being searched for the current frame. Also assume that previous frame's LSF 8 was found in the interval number 70. The proposed algorithm then first searches the LSF in the intervals 67 to 73. Further, as an example of the above-mentioned case, assume that the LSF 8 is for current frame is actually located at interval 60 and the next higher-order root, that is, LSF 10 for this frame happens to be at interval 72. This would then wrongly be detected as LSF 8. Next, when the algorithm tries to search LSF 9, it would start from interval 72 onwards and would not find any zero crossing, because interval 72 happened to contain the last root.

This implies that if a higher-order root is incorrectly detected, the search algorithm leads to *less than 10 LSFs* at the end of the complete search. Once this happens, *all*

Table 1: Reduction in count. "Count" represents the total number of times the polynomials $P'(z)$ and $Q'(z)$ are evaluated.

Filename	Original count	Count after modification	Percentage reduction
SAMPLE_SPEECH.PCM	3481856	2363495	47.31
OVERC63.TIN	4524	3065	32.25
OVERC53H.TIN	4764	3258	31.61
INEQC53.TIN	14490	5254	63.74
TAMEC63H.TIN	23542	5920	74.85
PATHC53.TIN	240530	134549	44.06
PATHC63H.TIN	238513	139040	41.71

Table 2: Reduction in "clock cycles" and indication of the percentage reduction in terms of clock cycles in the LSF search due to the algorithm.

Filename	Original clock cycles/frame	New clock cycles/frame	Reduction in clock cycles	Percentage reduction
SAMPLE_SPEECH.PCM	93010618	75440849	17569769	18.89
OVERC63.TIN	123317	106274	17043	13.82
OVERC53H.TIN	122738	106520	16218	13.21
INEQC53.TIN	127447	98131	29316	23.003
TAMEC63.TIN	130739	79571	50988	39.02

the 10 LSFs should be searched again using the normal G.723.1 method. By this preventive measure, the algorithm would never violate the G.723.1 recommendation. However, it should be noted that due to the corrective measure, the peak MIPS would get approximately doubled, since the LSF search for all 10 roots has to be done twice. But at the same time, the possibility of this case occurring is very small, hence the average MIPS is not adversely affected.

5. RESULTS

As mentioned before, the fixed point C code of G.723.1 was modified as per this algorithm and the results were verified on ARM-7TDMI general purpose RISC processor.

Table 1 shows the reductions for the prerecorded sample speech of duration 7.5 minutes, that is, about 15000 frames (SAMPLE_SPEECH.PCM, 16 bit PCM, 8 kHz, mono, signed) and also various G.723.1 test vectors given by ITU-T. The test vectors being synthesized sounds of short duration (and not real speech), are used only for testing the bit exactness of the algorithm. The results for SAMPLE_SPEECH.PCM are more meaningful for practical applications.

6. CONCLUSION

For real speech signals, the proposed algorithm can be expected to give an approximate improvement of 20% over the G.723.1 real root search algorithm. The algorithm has been tested for all the test vectors provided by ITU-T, so it is bit-exact compliant with G.723.1.

However, the percentage reduction in computations is implementation dependent. The C code that we ported on the ARM-7TDMI gives an average percentage reduction of about 20%, as indicated in Table 2. This is lesser than the percentage reduction in "count" shown by Table 1. This is because the algorithm involves many *if-else* checks. Such decision-making instructions lead to pipeline flushing and therefore tend to slow down the process.

It should be noted that the algorithm reduces only the *average* MIPS. The *peak* MIPS increases as mentioned in Section 4.2. Though the algorithm has been implemented in context of ITU-T Recommendation G.723.1, it is applicable to any other low bit rate codec provided it uses similar LSF search procedure.

ACKNOWLEDGMENT

The authors would like to thank Mr. Shivaram Gavankar, Mr. Mahesh Shukla, and Mr. Ravi Chaugule from Cirrus Logic Software Pvt. Ltd., India, for their guidance and support.

REFERENCES

[1] L. Rabiner and R. Schafer, *Digital Processing of Speech Signals*, Prentice-Hall, Eaglewood Cliffs, NJ, USA, 1978.

[2] A. M. Kondoz, *Digital Speech: Coding for Low Bit Rate Communication Systems*, John Wiley & Sons, New York, NY, USA, 1994.

[3] F. Itakura, "Line spectrum representation of linear predictive coefficients of speech signals," *Journal of the Acoustical Society of America*, vol. 57, no. 1, pp. s35, 1975.

[4] F. K. Soong and B. H. Juang, "Line spectrum pairs (LSP)

and speech data compression," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP '84)*, vol. 9, pp. 1.10.1–1.10.4, San Diego, Calif, USA, March 1984.

[5] P. Kabal and R. Ramachandran, "The computation of line spectral frequencies using Chebyshev polynomials," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 34, no. 6, pp. 1419–1426, 1986.

[6] S. Grassi, A. Dufaux, M. Ansorge, and F. Pellandini, "Efficient algorithm to compute LSP parameters from 10th-order LPC coefficients," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (ICASSP '97)*, vol. 3, pp. 1707–1710, Munich, Germany, April 1997.

[7] C. H. Wu and J.-H. Chen, "A novel two-level method for the computation of the LSP frequencies using a decimation-in-degree algorithm," *IEEE Trans. Speech and Audio Processing*, vol. 5, no. 2, pp. 106–115, 1997.

[8] S. Saoudi and J. Boucher, "A new efficient algorithm to compute LSP parameters for speech coding," *Signal Processing (Elsevier)*, vol. 28, no. 2, pp. 201–212, 1992.

[9] J. Rothweiler, "On polynomial reduction in the computation of LSP frequencies," *IEEE Trans. Speech and Audio Processing*, vol. 7, no. 5, pp. 592–594, 1999.

[10] ITU-T Recommendation G.723.1, "Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s," 1996.

Sameer A. Kibey received his B.S. degree with honours in electronics and telecommunication engineering from the Government College of Engineering, University of Pune, Pune, India in 2002. Since then, he has been with the Digital Signal Processing (DSP) & Multimedia Group at Tata Elxsi Ltd., Bangalore, India. His interests include algorithm development and optimizations for speech, audio, image, and video coding.

Jaydeep P. Kulkarni received his B.S. degree in electronics and telecommunication engineering from the Government College of Engineering, University of Pune, Pune, India in 2002 with honours. He is currently pursuing his M.Tech degree in electronics design and technology at the Centre for Electronics Design and Technology (CEDT), Indian Institute of Science, Bangalore. His research interests include transistor design methodologies in sub-100-nm regime, analog and RF CMOS design, VLSI for signal processing, and data compression techniques.

Piyush D. Sarode received his Diploma in electronics and telecommunications from Government Polytechnic, Nagpur, India in 1999 and his B.S. degree in electronics and telecommunication engineering from the Government College of Engineering, University of Pune, Pune, India in 2002 with honours. Since then he has been working in the field of real-time operating systems development at Honeywell Technology Solutions Labs, Bangalore, India. His interests include real-time operating systems, embedded systems, and algorithm development.