SILK Speech Codec

draft-vos-silk-00.txt

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on January 7, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.
Abstract

This document describes SILK, a speech codec for real-time, packet-based voice communications. Targeting a diverse range of operating environments, SILK provides scalability in several dimensions. Four different sampling frequencies are supported for encoding the audio input signal. Adaptation to network characteristics is provided through control of bitrate, packet rate, packet loss resilience and use of discontinuous transmission (DTX). And several different complexity levels let SILK take advantage of available processing power without relying on it. Each of these properties can be adjusted during operation of the codec on a frame-by-frame basis.

Table of Contents

1. Introduction .................................................. 3
2. Technical Requirements for Internet Wideband Audio Codec .... 4
   2.1. Bitrate .................................................... 4
   2.2. Sampling Rate ............................................. 4
   2.3. Complexity ............................................... 4
   2.4. Packet Loss Resilience .................................. 4
   2.5. Delay ...................................................... 4
   2.6. DTX ........................................................ 4
3. Outline of the Codec .......................................... 6
   3.1. Encoder .................................................... 6
      3.1.1. Control Parameters ................................... 6
      3.1.2. Voice Activity Detection ............................. 9
      3.1.3. High-Pass Filter ..................................... 9
      3.1.4. Pitch Analysis ....................................... 10
      3.1.5. Noise Shaping Analysis ............................... 11
      3.1.6. Prefilter ............................................. 15
      3.1.7. Prediction Analysis .................................. 15
      3.1.8. LSF Quantization ..................................... 16
      3.1.9. LTP Quantization ..................................... 19
      3.1.10. Noise Shaping Quantizer .............................. 20
      3.1.11. Range Encoder ....................................... 20
   3.2. Decoder .................................................... 21
      3.2.1. Range Decoder ......................................... 22
      3.2.2. Decode Parameters .................................... 22
      3.2.3. Generate Excitation .................................. 22
      3.2.4. LTP Synthesis ......................................... 22
      3.2.5. LPC Synthesis ......................................... 23
4. Reference Implementation ....................................... 24
5. Security Considerations ....................................... 25
6. Informative References ........................................ 26
Authors’ Addresses ............................................... 27
1. Introduction

A central component in voice communications is the speech codec, which compresses the audio signal for efficient transmission over a network. A good speech codec achieves high coding efficiency, meaning that it delivers high audio quality at a given bitrate. However, for a good user experience in a broad range of environments, a speech codec should also be able adapt its operating point to the characteristics and limitations of network, hardware and audio signal. SILK is a novel speech codec for real-time voice communications designed and developed by Skype [skype-website] to offer this kind of scalability. This document describes the technical details of SILK.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.
2. Technical Requirements for Internet Wideband Audio Codec

The Internet Wideband Audio Codec MUST be optimized towards real-time communications over the Internet, and MUST have the flexibility to adjust to the environment it operates in. Below is a list of concrete requirements for the codec.

2.1. Bitrate

The codec MUST provide a quality/bitrate trade-off that is competitive with other state-of-the-art codecs. It MUST be capable of running at bitrates below 10 kbps. At low bitrates it MUST deliver good quality for clean, noisy or hands-free speech in any language. At high bitrates the quality MUST be excellent for any audio signal, including music. The bitrate MUST be adjustable in real-time.

2.2. Sampling Rate

The codec MUST support multiple sampling rates, ranging from narrowband (8 kHz) to super wideband (24 kHz or more). Switching between sampling rates MUST be carried out in real-time.

2.3. Complexity

The codec MUST be capable of running at below 50 MHz of a x86 core in wideband mode (16 kHz sampling rate). The codec SHOULD have a complexity that is adjustable in real-time, where a higher complexity setting improves the quality/bitrate trade-off.

2.4. Packet Loss Resilience

The codec MUST be capable of running with little error propagation, meaning that the decoded signal after one or more packet losses is close to the decoded signal without packet losses after no more than two additional packets. The codec MUST have a packet loss resilience that is adjustable in real-time, where a lower packet loss resilience setting improves the quality/bitrate trade-off.

2.5. Delay

The codec MUST be capable of running with an algorithmic delay of no more than 30 milliseconds.

2.6. DTX

The codec SHOULD be capable of using Discontinuous Transmission (DTX) where packets are sent at a reduced rate when the input signal
contains only background noise.
3. Outline of the Codec

The SILK codec consist of an encoder and a decoder as described in Section 3.1 and Section 3.2, respectively.

3.1. Encoder

We start the description of the encoder by listing the parameters that controls the operating point of the encoder. Afterwards, we describe the encoder components in detail.

3.1.1. Control Parameters

The encoder with control parameters specifying the operating point is depicted in Figure 1. All control parameters can be changed during regular operation of the codec, when inputting a frame of audio data, without interrupting the audio stream from encoder to decoder. The codec control parameters are described in Section 3.1.1.1 to Section 3.1.1.5.

![Block diagram illustrating the control parameters that specifies the operating point of the SILK encoder.](image)

3.1.1.1. Sampling Rate

SILK can switch in real-time between audio sampling rates of 8, 12, 16 and 24 kHz. A higher sampling rate improves audio quality by preserving a larger part of the input signal frequency range, at the cost of increased CPU load and bitrate.
3.1.1.2. Bitrate

The bitrate can be set between 6 and 40 kbps. A higher bitrate improves audio quality by lowering the amount of quantization noise in the decoded signal. The required bitrate for a given level of quantization noise is approximately linear with the sampling rate. Good quality is achieved at around 1 bit/sample, and at 1.5 bits/sample the quality becomes transparent for most material.

3.1.1.3. Packet Rate

SILK encodes frames of 20 milliseconds at a time and can combine 1, 2, 3, 4 or 5 of these frames in one payload, thus creating one packet every 20, 40, 60, 80 or 100 milliseconds. Because of the overhead from IP/UDP/RTP headers, sending fewer packets per second reduces the bitrate, but increases latency and sensitivity to packet losses as losing one packet constitutes a loss of a bigger chunk of audio signal.

3.1.1.4. Packet Loss Resilience

Speech codecs often exploit inter-frame correlations to reduce the bitrate at a cost in error propagation: after losing one packet several packets need to be received before the decoder is able to accurately reconstruct the speech signal. The extent to which SILK exploits inter-frame dependencies can be adjusted on the fly to choose a trade-off between bitrate and amount of error propagation.

3.1.1.5. Complexity

SILK has several optional optimizations that can be enabled to reduce the CPU load by a few times, at the cost of increasing the bitrate by a few percent. The most important algorithmic parts controlled by the three complexity settings (that is, high, medium, and low) are:

- The filter order of the whitening filter and the downsampling quality in the pitch analysis.
- The filter order of the short-term noise shaping filter used in the prefilter and noise shaping quantizer.
- The accuracy in the prediction analysis, the use of simulated output, and adjustment of the number of survivors that are carried over between stages in the multi-stage LSF vector quantization.
- The number of states in delayed decision quantization of the residual signal.

In the following, we focus on the core encoder and describe its components. For simplicity, we will refer to the core encoder simply as the encoder in the remainder of this document. An overview of the encoder is given in Figure 2.
1: Input speech signal
2: High passed input signal
3: Voice activity estimate
4: Pitch lags (per 5 ms) and voicing decision (per 20 ms)
5: Noise shaping quantization coefficients
   - Short term synthesis and analysis noise shaping coefficients (per 5 ms)
   - Long term synthesis and analysis noise shaping coefficients (per 5 ms and for voiced speech only)
   - Noise shape tilt (per 5 ms)
   - Quantizer gain/step size (per 5 ms)
6: Input signal filtered with analysis noise shaping filters
7: Simulated output signal
8: Short and long term prediction coefficients
   LTP (per 5 ms) and LPC (per 20 ms)
9: LSF quantization indices
10: LSF coefficients
3.1.2. Voice Activity Detection

The input signal is processed by a VAD (Voice Activity Detector) to produce a measure of voice activity, and also spectral tilt and signal-to-noise estimates, for each frame. The VAD uses a sequence of half-band filterbanks to split the signal in four subbands: 0 – Fs/16, Fs/16 – Fs/8, Fs/8 – Fs/4, and Fs/4 – Fs/2, where Fs is the sampling frequency (8, 12, 16 or 24 kHz). The lowest subband, from 0 – Fs/16 is high-pass filtered with a first-order MA (Moving Average) filter (with transfer function $H(z) = 1-z^(-1)$) to reduce the energy at the lowest frequencies. For each frame, the signal energy per subband is computed. In each subband, a noise level estimator tracks the background noise level and an SNR (Signal-to-Noise Ratio) value is computed as the logarithm of the ratio of energy to noise level. Using these intermediate variables, the following parameters are calculated for use in other SILK modules:

- Average SNR. The average of the subband SNR values.
- Smoothed subband SNRs. Temporally smoothed subband SNR values.
- Speech activity level. Based on the average SNR and a weighted average of the subband energies.
- Spectral tilt. A weighted average of the subband SNRs, with positive weights for the low subbands and negative weights for the high subbands.

3.1.3. High-Pass Filter

The input signal is filtered by a high-pass filter to remove the lowest part of the spectrum that contains little speech energy and may contain background noise. This is a second order ARMA (Auto Regressive Moving Average) filter with a cut-off frequency around 70 Hz.

In the future, a music detector may also be used to lower the cut-off frequency when the input signal is detected to be music rather than speech.
3.1.4. Pitch Analysis

The high-passed input signal is processed by the open loop pitch estimator shown in Figure 3.

![Diagram of pitch estimator]

1: Input signal
2: Lag candidates from stage 1
3: Lag candidates from stage 2
4: Correlation threshold
5: Voiced/unvoiced flag
6: Pitch correlation
7: Pitch lags

Block diagram of the pitch estimator.

Figure 3

The pitch analysis finds a binary voiced/unvoiced classification, and, for frames classified as voiced, four pitch lags per frame - one for each 5 ms subframe - and a pitch correlation indicating the periodicity of the signal. The input is first whitened using a Linear Prediction (LP) whitening filter, where the coefficients are computed through standard Linear Prediction Coding (LPC) analysis. The order of the whitening filter is 16 for best results, but is reduced to 12 for medium complexity and 8 for low complexity modes. The whitened signal is analyzed to find pitch lags for which the time...
correlation is high. The analysis consists of three stages for reducing the complexity:
  o In the first stage, the whitened signal is downsampled 4 times and the current frame is correlated to a signal delayed by a range of lags, starting from a shortest lag corresponding to 500 Hz, to a longest lag corresponding to 56 Hz.
  o The second stage operates on a two times downsampled signal and measures time correlations only near the lags corresponding to those that had sufficiently high correlations in the first stage. The resulting correlations are adjusted for a small bias towards short lags to avoid ending up with a multiple of the true pitch lag. The highest adjusted correlation is compared to a threshold depending on:
    * Whether the previous frame was classified as voiced
    * The speech activity level
    * The spectral tilt.
  If the threshold is exceeded, the current frame is classified as voiced and the lag with the highest adjusted correlation is stored for a final pitch analysis of the highest precision in the third stage.
  o The last stage operates directly on the whitened input signal to compute time correlations for each of the four subframes independently in a narrow range around the lag with highest correlation from the second stage.

3.1.5. Noise Shaping Analysis

The noise shaping analysis finds gains and filter coefficients used in the prefilter and noise shaping quantizer. These parameters are chosen such that they will fulfill several requirements:
  o Balancing quantization noise and bitrate. The quantization gains determine the step size between reconstruction levels of the excitation signal. Therefore, increasing the quantization gain amplifies quantization noise, but also reduces the bitrate by lowering the entropy of the quantization indices.
  o Spectral shaping of the quantization noise; the noise shaping quantizer is capable of reducing quantization noise in some parts of the spectrum at the cost of increased noise in other parts, without substantially changing the bitrate. By shaping the noise such that it follows the signal spectrum, it becomes less audible. In practice, best results are obtained by making the shape of the noise spectrum slightly flatter than the signal spectrum.
  o Deemphasizing spectral valleys; by using different coefficients in the analysis and synthesis part of the prefilter and noise shaping quantizer, the levels of the spectral valleys can be decreased relative to the levels of the spectral peaks such as speech formants and harmonics. This reduces the entropy of the signal, which is the difference between the coded signal and the
quantization noise, thus lowering the bitrate.

- Matching the levels of the decoded speech formants to the levels of the original speech formants; an adjustment gain and a first order tilt coefficient are computed to compensate for the effect of the noise shaping quantization on the level and spectral tilt.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{silk DIAG.png}
\caption{Noise shaping and spectral de-emphasis illustration.}
\end{figure}

Figure 4 shows an example of an input signal spectrum (1). After de-emphasis and level matching, the spectrum has deeper valleys (2). The quantization noise spectrum (3) more or less follows the input signal spectrum, having slightly less pronounced peaks. The entropy, which provides a lower bound on the bitrate for encoding the excitation signal, is proportional to the area between the deemphasized spectrum (2) and the quantization noise spectrum (3). Without de-emphasis, the entropy is proportional to the area between input spectrum (1) and quantization noise (3) - clearly higher.

The transformation from input signal to deemphasized signal can be described as a filtering operation with a filter

\[ W_{\text{ana}}(z) \]
\[
H(z) = G \times (1 - c_{\text{tilt}} \times z^{-1}) \times \frac{\text{Wsyn}(z)}{	ext{Wana}(z)},
\]

having an adjustment gain G, a first order tilt adjustment filter with tilt coefficient \(c_{\text{tilt}}\), and where

\[
\text{Wana}(z) = \sum_{k=1}^{d} (1 - (a_{\text{ana}}(k) \times z^{-k}) \times (1 - z^{-L}) \times b_{\text{ana}}(k) \times z^{-k}),
\]

is the analysis part of the de-emphasis filter, consisting of the short-term shaping filter with coefficients \(a_{\text{ana}}(k)\), and the long-term shaping filter with coefficients \(b_{\text{ana}}(k)\) and pitch lag \(L\). The parameter \(d\) determines the number of long-term shaping filter taps.

Similarly, but without the tilt adjustment, the synthesis part can be written as

\[
\text{Wsyn}(z) = \sum_{k=1}^{d} (1 - (a_{\text{syn}}(k) \times z^{-k}) \times (1 - z^{-L}) \times b_{\text{syn}}(k) \times z^{-k}).
\]

All noise shaping parameters are computed and applied per subframe of 5 milliseconds. First, an LPC analysis is performed on a windowed signal block of 16 milliseconds. The signal block has a look-ahead of 5 milliseconds relative to the current subframe, and the window is an asymmetric sine window. The LPC analysis is done with the autocorrelation method, with an order of 16 for best quality or 12 in low complexity operation. The quantization gain is found as the square-root of the residual energy from the LPC analysis, multiplied by a value inversely proportional to the coding quality control parameter and the pitch correlation.

Next we find the two sets of short-term noise shaping coefficients \(a_{\text{ana}}(k)\) and \(a_{\text{syn}}(k)\), by applying different amounts of bandwidth expansion to the coefficients found in the LPC analysis. This bandwidth expansion moves the roots of the LPC polynomial towards the origin, using the formulas
\[ a_{\text{ana}}(k) = a(k) \times g_{\text{ana}}^k, \text{ and} \]
\[ a_{\text{syn}}(k) = a(k) \times g_{\text{syn}}^k, \]

where \( a(k) \) is the \( k \)'th LPC coefficient and the bandwidth expansion factors \( g_{\text{ana}} \) and \( g_{\text{syn}} \) are calculated as

\[ g_{\text{ana}} = 0.94 - 0.02 \times C, \text{ and} \]
\[ g_{\text{syn}} = 0.94 + 0.02 \times C, \]

where \( C \) is the coding quality control parameter between 0 and 1. Applying more bandwidth expansion to the analysis part than to the synthesis part gives the desired de-emphasis of spectral valleys in between formants.

The long-term shaping is applied only during voiced frames. It uses three filter taps, described by

\[ b_{\text{ana}} = F_{\text{ana}} \times [0.25, 0.5, 0.25], \text{ and} \]
\[ b_{\text{syn}} = F_{\text{syn}} \times [0.25, 0.5, 0.25]. \]

For unvoiced frames these coefficients are set to 0. The multiplication factors \( F_{\text{ana}} \) and \( F_{\text{syn}} \) are chosen between 0 and 1, depending on the coding quality control parameter, as well as the calculated pitch correlation and smoothed subband SNR of the lowest subband. By having \( F_{\text{ana}} \) less than \( F_{\text{syn}} \), the pitch harmonics are emphasized relative to the valleys in between the harmonics.

The tilt coefficient \( c_{\text{tilt}} \) is for unvoiced frames chosen as

\[ c_{\text{tilt}} = 0.4, \text{ and as} \]
\[ c_{\text{tilt}} = 0.04 + 0.06 \times C \]

for voiced frames, where \( C \) again is the coding quality control parameter and is between 0 and 1.

The adjustment gain \( G \) serves to correct any level mismatch between original and decoded signal that might arise from the noise shaping and de-emphasis. This gain is computed as the ratio of the prediction gain of the short-term analysis and synthesis filter coefficients. The prediction gain of an LPC synthesis filter is the square-root of the output energy when the filter is excited by a
unit-energy impulse on the input. An efficient way to compute the prediction gain is by first computing the reflection coefficients from the LPC coefficients through the step-down algorithm, and extracting the prediction gain from the reflection coefficients as

\[
\text{predGain} = \left( \prod_{k=1}^{K} 1 - (r_k)^2 \right)^{-0.5},
\]

where \( r_k \) is the \( k \)'th reflection coefficient.

Initial values for the quantization gains are computed as the square-root of the residual energy of the LPC analysis, adjusted by the coding quality control parameter. These quantization gains are later adjusted based on the results of the prediction analysis.

3.1.6. Prefilter

In the prefilter the input signal is filtered using the spectral valley de-emphasis filter coefficients from the noise shaping analysis, see Section 3.1.5. The filter output is called the simulated output signal and is passed on to the prediction analysis.

Also, by applying only the noise shaping analysis filter to the input signal, it provides the input to the noise shaping quantizer.

3.1.7. Prediction Analysis

The prediction analysis is performed in one of two ways depending on how the pitch estimator classified the frame. The processing for voiced and unvoiced speech are described in Section 3.1.7.1 and Section 3.1.7.2, respectively. Inputs to this function include the pre-whitened signal from the pitch estimator, see Section 3.1.4.

3.1.7.1. Voiced Speech

For a frame of voiced speech the pitch pulses will remain dominant in the pre-whitened input signal. Further whitening is desirable as it leads to higher quality at the same available bit-rate. To achieve this, a Long-Term Prediction (LTP) analysis is carried out to estimate the coefficients of a fifth order LTP filter for each of four sub-frames. The LTP coefficients are used to find an LTP residual signal with the simulated output signal as input to obtain better modelling of the output signal. This LTP residual signal is the input to an LPC analysis where the LPCs are estimated using the...
covariance method, such that the residual energy is minimized. The estimated LPCs are converted to a Line Spectral Frequency (LSF) vector, and quantized as described in Section 3.1.8. After quantization, the quantized LSF vector is converted to LPC coefficients and hence by using these quantized coefficients the encoder remains fully synchronized with the decoder. The LTP coefficients are quantized using a method described in Section 3.1.9. The quantized LPC and LTP coefficients are now used to filter the simulated output signal and measure a residual energy for each of the four subframes.

3.1.7.2. Unvoiced Speech

For a speech signal that has been classified as unvoiced there is no need for LTP filtering as it has already been determined that the pre-whitened input signal is not periodic enough within the allowed pitch period range for an LTP analysis to be worth-while the cost in terms of complexity and rate. Therefore, the pre-whitened input signal is discarded and instead the simulated output is used for LPC analysis using the covariance method. The resulting LPC coefficients are converted to an LSF vector, quantized as described in the following section and transformed back to obtain quantized LPC coefficients. The quantized LPC coefficients are used to filter the simulated output signal and measure a residual energy for each of the four subframes.

3.1.8. LSF Quantization

The purpose of quantization is to significantly lower the bit rate at the cost of some introduced distortion. A higher rate should always lead to lower distortion, and lowering the rate will generally lead to higher distortion. A commonly used but generally sub-optimal approach is to use a quantization method with a constant rate where only the error is minimized when quantizing.

3.1.8.1. Rate-Distortion Optimization

Instead, we minimize an objective function that consists of a weighted sum of rate and distortion, and use a codebook with an associated non-uniform rate table. Thus, we take into account that the probability mass function for selecting the codebook entries are by no means guaranteed to be uniform in our scenario. The advantage of this approach is that it ensures that rarely used codebook vector centroids, which are modelling statistical outliers in the training set can be quantized with a low error but with a relatively high cost in terms of a high rate. At the same time this approach also provides the advantage that frequently used centroids are modelled with low error and a relatively low rate. This approach will lead to
equal or lower distortion than the fixed rate codebook at any given average rate, if the data is similar to the data used for training the codebook.

3.1.8.2. Error Mapping

Instead of minimizing the error in the LSF domain, we map the errors to spectral distortion by applying a weight to the error of each element in the error vector. These weight vectors are calculated for each input vector as a linear approximation of the true mapping function, which is accurate for small errors. Consequently, we solve the following minimization problem, i.e.,

$$\text{LSF}_q = \arg\min \{ (\text{LSF} - c)' * W * (\text{LSF} - c) + \mu * \text{rate} \},$$

where $\text{LSF}_q$ is the quantized vector, $\text{LSF}$ is the input vector to be quantized, and $c$ is the quantized LSF vector candidate taken from the set $C$ of all possible outcomes of the codebook.

3.1.8.3. Multi-Stage Vector Codebook

We arrange the codebook in a multiple stage structure to achieve a quantizer that is both memory efficient and highly scalable in terms of computational complexity, see e.g. [sinervo-norsig]. In the first stage the input is the LSF vector to be quantized, and in any other stage $s > 1$, the input is the quantization error from the previous stage, see Figure 5.

**Multi-Stage LSF Vector Codebook Structure.**
By storing total of $M$ codebook vectors, i.e.,

\[
S
\]

\[
M = \sum_{s=1}^{S} M_s,
\]

where $M_s$ is the number of vectors in stage $s$, we obtain a total of

\[
T = \sum_{s=1}^{S} M_s
\]

possible combinations for generating the quantized vector. It is for example possible to represent $2^{36}$ unique vectors using only 216 vectors in memory, as done in SILK for voiced speech at all sample frequencies above 8 kHz.

### 3.1.8.4. Survivor Based Codebook Search

This number of possible combinations is far too high for a full search to be carried out for each frame so for all stages but the last, i.e., $s$ smaller than $S$, only the best $\min(L, M_s)$ centroids are carried over to stage $s+1$. In each stage the objective function, i.e., the weighted sum of accumulated bit-rate and distortion, is evaluated for each codebook vector entry and the results are sorted. Only the best paths and the corresponding quantization errors are considered in the next stage. In the last stage $S$ the single best path through the multistage codebook is determined. By varying the maximum number of survivors from each stage to the next $L$, the complexity can be adjusted in real-time at the cost of a potential decrease in the objective function for the resulting quantized vector. This approach scales all the way between the two extremes, $L=1$ being a greedy search, and the desirable but infeasible full search, $L=T/MS$. In fact, a performance almost as good as what can be achieved with the infeasible full search can be obtained at a substantially lower complexity by using this approach, see e.g. [leblanc-tsap].
3.1.8.5. LSF Stabilization

If the input is stable, finding the best candidate will usually result in the quantized vector also being stable, but due to the multi-stage approach it could in theory happen that the best quantization candidate is unstable and because of this there is a need to explicitly ensure that the quantized vectors are stable. Therefore we apply a LSF stabilization method which ensures that the LSF parameters are within valid range, increasingly sorted, and have minimum distances between each other and the border values that have been pre-determined as the 0.01 percentile distance values from a large training set.

3.1.8.6. Off-Line Codebook Training

The vectors and rate tables for the multi-stage codebook are trained by minimizing the average of the objective function for LSF vectors from a large training set.

3.1.9. LTP Quantization

For voiced frames, the prediction analysis described in Section 3.1.7.1 resulted in four sets (one set per subframe) of five LTP coefficients, plus four weighting matrices. The LTP coefficients for each subframe are quantized using entropy constrained vector quantization. A total of three vector codebooks are available for quantization, with different rate-distortion trade-offs. The three codebooks have 10, 20 and 40 vectors and average rates of about 3, 4, and 5 bits per vector, respectively. Consequently, the first codebook has larger average quantization distortion at a lower rate, whereas the last codebook has smaller average quantization distortion at a higher rate. Given the weighting matrix \( W_{ltp} \) and LTP vector \( b \), the weighted rate-distortion measure for a codebook vector \( cb_i \) with rate \( r_i \) is give by

\[
RD = u \times (b - cb_i)' * W_{ltp} * (b - cb_i) + r_i,
\]

where \( u \) is a fixed, heuristically-determined parameter balancing the distortion and rate. Which codebook gives the best performance for a given LTP vector depends on the weighting matrix for that LTP vector. For example, for a low valued \( W_{ltp} \), it is advantageous to use the codebook with 10 vectors as it has a lower average rate. For a large \( W_{ltp} \), on the other hand, it is often better to use the codebook with 40 vectors, as it is more likely to contain the best codebook vector. The weighting matrix \( W_{ltp} \) depends mostly on two aspects of the input signal. The first is the periodicity of the signal; the more
periodic the larger W_ltp. The second is the change in signal energy in the current subframe, relative to the signal one pitch lag earlier. A decaying energy leads to a larger W_ltp than an increasing energy. Both aspects do not fluctuate very fast which causes the W_ltp matrices for different subframes of one frame often to be similar. As a result, one of the three codebooks typically gives good performance for all subframes. Therefore the codebook search for the subframe LTP vectors is constrained to only allow codebook vectors to be chosen from the same codebook, resulting in a rate reduction.

To find the best codebook, each of the three vector codebooks is used to quantize all subframe LTP vectors and produce a combined weighted rate-distortion measure for each vector codebook and the vector codebook with the lowest combined rate-distortion over all subframes is chosen. The quantized LTP vectors are used in the noise shaping quantizer, and the index of the codebook plus the four indices for the four subframe codebook vectors are passed on to the range encoder.

3.1.10. Noise Shaping Quantizer

The noise shaping quantizer independently shapes the signal and coding noise spectra to obtain a perceptually higher quality at the same bitrate.

The prefilter output signal is multiplied with a compensation gain G computed in the noise shaping analysis. Then the output of a synthesis shaping filter is added, and the output of a prediction filter is subtracted to create a residual signal. The residual signal is multiplied by the inverse quantized quantization gain from the noise shaping analysis, and input to a scalar quantizer. The quantization indices of the scalar quantizer represent a signal of pulses that is input to the pyramid range encoder. The scalar quantizer also outputs a quantization signal, which is multiplied by the quantized quantization gain from the noise shaping analysis to create an excitation signal. The output of the prediction filter is added to the excitation signal to form the quantized output signal y(n). The quantized output signal y(n) is input to the synthesis shaping and prediction filters.

3.1.11. Range Encoder

Range encoding is a well known method for entropy coding in which a bitstream sequence is continually updated with every new symbol, based on the probability for that symbol. It is similar to arithmetic coding but rather than being restricted to generating binary output symbols, it can generate symbols in any chosen number
In SILK all side information is range encoded. Each quantized parameter has its own cumulative density function based on histograms for the quantization indices obtained by running a training database.

### 3.1.11.1. Bitstream Encoding Details

TBD.

### 3.2. Decoder

At the receiving end, the received packets are by the range decoder split into a number of frames contained in the packet. Each of which contains the necessary information to reconstruct a 20 ms frame of the output signal. An overview of the decoder is given in Figure 6.

![Decoder block diagram.](image)

1: Range encoded bitstream
2: Coded parameters
3: Pulses and gains
4: Pitch lags and LTP coefficients
5: LPC coefficients
6: Decoded signal

**Figure 6**
3.2.1. Range Decoder

The range decoder decodes the encoded parameters from the received bitstream. Output from this function includes the pulses and gains for the excitation signal generation, as well as LTP and LSF codebook indices, which are needed for decoding LTP and LPC coefficients needed for LTP and LPC synthesis filtering the excitation signal, respectively.

3.2.2. Decode Parameters

Pulses and gains are decoded from the range decoded bitstream in the following way... (TBD)

When a voiced frame is decoded and LTP codebook selection and indices are received, LTP coefficients are decoded using the selected codebook by choosing the vector that corresponds to the given codebook index. This is done for each of the four subframes. The LPC coefficients are decoded from the LSF codebook by first adding the chosen vectors, one vector from each stage of the codebook. The resulting LSF vector is stabilized using the same method as was used in the encoder, see Section 3.1.8.5. The LSF coefficients are then converted to LPC coefficients, and passed on to the LPC synthesis filter.

3.2.3. Generate Excitation

The pulses signal is multiplied with the quantization gain to create the excitation signal.

3.2.4. LTP Synthesis

For voiced speech, the excitation signal \( e(n) \) is input to an LTP synthesis filter that will recreate the long term correlation that was removed in the LTP analysis filter and generate an LPC excitation signal \( e_{\text{LPC}}(n) \), according to

\[
e_{\text{LPC}}(n) = e(n) + \sum_{i=-d}^{d} e(n - L - i) \cdot b_i,
\]

using the pitch lag \( L \), and the decoded LTP coefficients \( b_i \). For unvoiced speech, the output signal is a copy of the input signal, i.e., \( e_{\text{LPC}}(n) = e(n) \).
3.2.5. LPC Synthesis

In a similar manner, the short-term correlation that was removed in the LPC analysis filter is recreated in the LPC synthesis filter. The LPC excitation signal $e_{\text{LPC}}(n)$ is filtered using the LTP coefficients $a_i$, according to

$$
d_{\text{LPC}} \quad y(n) = e_{\text{LPC}}(n) + \sum_{i=1}^{d_{\text{LPC}}} e_{\text{LPC}}(n - i) * a_i,
$$

where $d_{\text{LPC}}$ is the LPC synthesis filter order, and $y(n)$ is the decoded signal.
4. Reference Implementation

To Be Defined.
5. Security Considerations

To Be Defined.
6. Informative References

[leblanc-tsap]

[sinervo-norsig]

[skype-website]
Authors’ Addresses

Koen Vos  
Skype Technologies S.A.  
Stadsgaarden 6  
Stockholm 11645  
SE  
Phone: +46 855 921 989  
Email: koen.vos@skype.net

Soeren Skak Jensen  
Skype Technologies S.A.  
Stadsgaarden 6  
Stockholm 11645  
SE  
Phone: +46 855 921 989  
Email: soren.skak.jensen@skype.net

Karsten Vandborg Soerensen  
Skype Technologies S.A.  
Stadsgaarden 6  
Stockholm 11645  
SE  
Phone: +46 855 921 989  
Email: karsten.vandborg.soerensen@skype.net