

6.2.2 低频信号编码概述

6.2.2.1 ACELP 和 TAC 双核编码

$F_s/2$ 频率采样的低频信号($0 \sim F_s/4$ 频带)使用基于 ACELP 和 TAC 切换的双核编码器。ACELP 属于时域预测的编码技术,适合语音信号和瞬态信号编码。TAC 属于变换域的编码技术,更适合典型的音乐信号和稳态信号编码,本标准采用了其中一种称为 TVC 的变换域编码技术。

6.2.2.2 ACELP 和 TVC 的时间图

ACELP 和 TVC 双核编码的输入是按 $F_s/2$ 频率采样的单声道信号,以连续 256 个采样点组成一帧进行处理。每帧可采用两种模式编码,采用哪一种取决于信号特征,见图 28 所示。在 ACELP 模式中,采用 ACELP 核编码。在 TVC 模式中,采用 TVC 核编码,由于 TVC 是变换编码技术,需要加下一帧的前 32 个样本用于帧重叠。

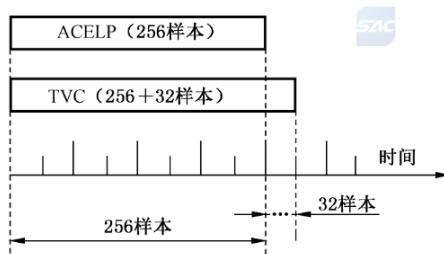


图 28 帧类型的时间图

6.2.2.3 ACELP 和 TVC 的闭环模式选择

音频帧首先使用多种模式分别编码,然后选择最好的模式。选择的标准是加权信号 $x_w(n)$ 和合成加权信号 $\hat{x}_w(n)$ 间的分段信噪比均值。子帧的分段信噪比见式(2):

$$segSNR_i = 20 \log_{10} \left\{ \frac{\sum_{n=0}^{N-1} x_w^2(n)}{\sum_{n=0}^{N-1} (x_w(n) - \hat{x}_w(n))^2} \right\} \quad \dots \dots \dots \quad (2)$$

式中:

N ——子帧长度(64 个样点)。

每帧分段信噪比均值计算见式(3):

$$\overline{segSNR} = \frac{1}{N_{SF}} \sum_{i=0}^{N_{SF}-1} segSNR_i \quad \dots \dots \dots \quad (3)$$

式中:

N_{SF} ——音频帧中子帧的数目,本标准规定 N_{SF} 的值是 4。

6.2.3 ACELP 编码

6.2.3.1 预加重

输入到 ACELP 核编码的信号,通过一阶的预加重滤波器 $H_{emph}(z)$,见 6.2.3.3。

6.2.3.2 LP 分析和量化

6.2.3.2.1 线性预测分析

线性预测分析是用 16 阶线性预测器作短时分析,采用莱文逊—杜宾(Levinson—Durbin)算法进行线性预测系数求解,对每帧分析一次得到一组线性预测系数。线性预测系数在编码前要先转化为 ISF 系数,然后再进行量化。LP 合成滤波器的传递函数见式(4):

式中：

\hat{a}_i ——量化后的线性预测系数, $m=16$ 是预测阶数。

LP 分析首先用 384 个样本的非对称窗加权预加重后的信号 $s(n)$, 计算自相关系数, 用莱文逊—杜宾算法求 LP 系数, 然后转换为 ISP 系数并在 ISP 域插值, 最后转到 ISF 域量化。

384 个样本的 LP 分析帧结构如图 29 所示, 其中 256 个样本来自第 n 帧, 64 个样本来自第 $n-1$ 帧, 64 个样本来自第 $n+1$ 帧。第 n 帧分析窗与第 $n-1$ 帧分析窗有 128 个样本的重叠。因此 LP 分析需要前瞻 64 个样本。

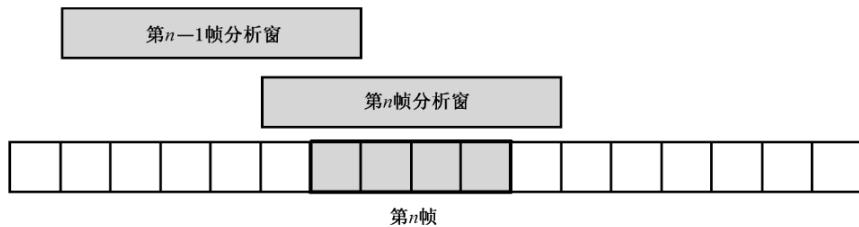


图 29 线性预测分析帧结构图

6.2.3.2.2 加窗和自相关函数的计算

分析窗采用重心在第四个子帧的非对称窗,该窗由两部分组成,第一部分是半个海明(Hamming)窗,第二部分是 $1/4$ 余弦窗,见式(5):

式中：

$$L_1=256, L_2=128.$$

设加窗后信号为 $s'(n)$ ，则有式(6)：

式中：

$w(n)$ ——加窗函数；

$s(n)$ ——预加重后的信号。

$s'(n)$ 对应的自相关函数见式(7)：

然后用滞后窗 $w_{lag}(i)$ 乘自相关函数使其具有 60 Hz 的带宽扩展, 滞后窗 $w_{lag}(i)$ 的表达式见式(8):

式中：

f_0 ——扩展的带宽, $f_0 = 60$ Hz;

f_s ——采样频率, $f_s = 12.8$ kHz。

另外 $r(0)$ 乘以白噪声校正因子 1.000 1。最后得到修正后的自相关函数 $r'(k)$ 如式(9)：

6.2.3.2.3 用莱文逊—杜宾算法求解 LP 系数

用修正后的自相关函数 $r'(k)$ 求解线性预测系数 a_k , $k = 1, \dots, 16$, 即求解下述方程组, 见式(10):

该方程组可用莱文逊—杜宾算法求解, 算法计算步骤如式(11):

```

 $E(0) = r'(0);$ 
 $for(i=1; i <= 16; ++i)$ 
 $\{$ 
 $k_i = - \left[ r'(i) + \sum_{j=1}^{i-1} a_j^{i-1} r'(i-j) \right] / E(i-1);$ 
 $a_i^{(i)} = k_i;$ 
 $for(j=1; j <= i-1; ++j)$ 
 $\{$ 
 $a_j^{(i)} = a_j^{(i-1)} + k_i a_{i-j}^{(i-1)};$ 
 $\}$ 
 $E(i) = (1 - k_i^2) E(i-1);$ 
 $\}$ 

```

最后得到线性预测系数 $a_j = a_j^{(16)}$, $j = 1, \dots, 16$ 。

线性预测系数转化成 ISP 系数,以便于量化和内插。

6.2.3.2.4 LP 系数转换为 ISP 系数

对于 16 阶线性预测器来说, ISP 系数就是下面多项式式(12)和式(13)的根:

多项式 $F'_1(z)$ 和 $F'_2(z)$ 分别为对称和反对称多项式。它们的根在单位圆上，而且相互交替出现。其中 $F'_2(z)$ 有两个根分别为 $z=1(\omega=0)$ 和 $z=-1(\omega=\pi)$ 。为了消除这两个根，定义两个新的多项式记作 $F_1(z)$ 和 $F_2(z)$ ，见式(14)和式(15)：

多项式 $F_1(z)$ 和 $F_2(z)$ 分别有 8 个和 7 个共轭复根在单位圆上 ($e^{\pm j\omega_i}$)，由式(16)和式(17)表示：

式中：

q_i ——ISP 系数, $q_i = \cos(\omega_i)$, $i = 0, \dots, 15$;

ω_i ——ISF 系数, ISF 系数满足顺序特性, 即 $0 < \omega_0 < \omega_1 < \dots < \omega_{14} < \pi$;

a_{16} ——最后一阶线性预测系数, $a_{15} = a_{16}$ 。

由下面递推关系得到 $F_1(z)$ 和 $F_2(z)$ 多项式的系数 $f_1(i)$ 和 $f_2(i)$, 见式(18)：

$$\begin{aligned}
 & \text{for } i = 0; i <= 7; ++i \\
 & \quad \{ \\
 & \quad \quad f_1(i) = a_i + a_{m-i}; \\
 & \quad \quad f_2(i) = a_i - a_{m-i} + f_2(i-2); \\
 & \quad \} \\
 & \quad f_1(8) = 2a_8;
 \end{aligned} \tag{18}$$

式中：

$m = 16$ ——预测器阶数；

$f_2(-2) = f_2(-1) = 0$ 。

ISP 系数的求解过程描述如下：

首先将 0 到 π 初分成 100 点, 找到符号变化的区间。然后使用切比雪夫多项式法在每个区间求根, 符号变换的区间进一步细分为 4 个子区间。使用这个方法得到的根是 ISP 系数。

多项式 $F_1(z)$ 和 $F_2(z)$ 在 $z = e^{j\omega}$ 处可表示为式(19)和式(20)：

$$F_1(\omega) = 2e^{-j8\omega}C_1(x) \text{ 和 } F_2(\omega) = 2e^{-j7\omega}C_2(x) \tag{19}$$

$$C_1(x) = \sum_{i=0}^7 (f_1(i)T_{8-i}(x) + f_1(8)/2) \text{ 和 } C_2(x) = \sum_{i=0}^6 (f_2(i)T_{8-i}(x) + f_2(7)/2) \dots \tag{20}$$

式中：

$T_m = \cos(m\omega)$ ——切比雪夫多项式第 m 个根。

多项式 $C(x)$ 在 $x = \cos(\omega)$ 处的递归计算如式(21)：

$$\begin{aligned}
 & \text{for } (k = n_f - 1; k <= 1; --k) \\
 & \quad \{ \\
 & \quad \quad b_k = 2xb_{k+1} - b_{k+2} + f(n_f - k); \\
 & \quad \} \\
 & \quad C(x) = xb_1 - b_2 + f(n_f)/2
 \end{aligned} \tag{21}$$

对于多项式 $C_1(x)$, $n_f = 8$; 对于多项式 $C_2(x)$, $n_f = 7$; 迭代初始值 $b_{nf} = f(0)$ 和 $b_{nf+1} = 0$ 。

6.2.3.2.5 ISP 系数转换为 LP 系数

一旦 ISP 系数被量化和内插, 将被再次转回到 LP 系数。具体转换过程如下：

根据已量化和内插的 ISP 系数, 用式(16)、式(17)求 $F_1(z)$ 和 $F_2(z)$ 的系数, 用 $q_i = \cos(\omega_i)$, $i = 0, \dots, m-1$ (其中 $m=16$) 迭代计算系数 $f_1(i)$, 见式(22)：

$$\begin{aligned}
 & \text{for } (i = 2; i <= m/2; ++i) \\
 & \quad \{ \\
 & \quad \quad f_1(i) = -2q_{2i-2}f_1(i-1) + 2f_1(i-2); \\
 & \quad \quad \text{for } (j = i-1; j <= 2; --j) \\
 & \quad \quad \quad f_1(j) = f_1(j) - 2q_{2i-2}f_1(j-1) + f_1(j-2); \\
 & \quad \quad f_1(1) = f_1(1) - 2q_{2i-2}; \\
 & \quad \}
 \end{aligned} \tag{22}$$

式中：

初始值 $f_1(0) = 1$, $f_1(1) = -2q_0$ 。

同理,计算 $f_2(i)$,只是需要用 q_{2i-1} 代替式(22)中的 q_{2i-2} ,用 $m/2-1$ 代替 $m/2$,初始值变为 $f_2(0)=1$, $f_2(1)=-2q_1$ 。

求出 $f_1(i)$, $i=0,\dots,m/2$ 和 $f_2(i)$, $i=0,\dots,m/2-1$ 即得到 $F_1(z)$ 和 $F_2(z)$ 的系数, $F_2(z)$ 再乘以 $1-z^{-2}$ 就得到 $F'_2(z)$ 。那么 $F'_1(z)$ 和 $F'_2(z)$ 多项式的系数 $f'_1(i)$ 和 $f'_2(i)$ 如式(23):

$$\begin{aligned} f'_2(i) &= f_2(i) - f_2(i-2), \quad i=2,\dots,m/2-1 \\ f'_1(i) &= f_1(i), \quad i=0,\dots,m/2 \end{aligned} \quad \text{.....(23)}$$

$F'_1(z)$ 和 $F'_2(z)$ 分别乘以 $1+q_{m-1}$ 和 $1-q_{m-1}$,多项式系数最终变为式(24):

$$\begin{aligned} f'_2(i) &= (1-q_{m-1})f'_2(i), \quad i=0,\dots,m/2-1 \\ f'_1(i) &= (1+q_{m-1})f'_1(i), \quad i=0,\dots,m/2 \end{aligned} \quad \text{.....(24)}$$

由于 $F'_1(z)$ 和 $F'_2(z)$ 分别是对称和反对称多项式,根据关系式 $A(z)=(F'_1(z)+F'_2(z))/2$,最后得到 LP 系数,见式(25):

$$a_i = \begin{cases} 0.5f'_1(i) + 0.5f'_2(i), & i=1,\dots,m/2-1 \\ 0.5f'_1(i) - 0.5f'_2(i), & i=m/2+1,\dots,m-1 \\ 0.5f'_1(m/2), & i=m/2 \\ q_{m-1}, & i=m \end{cases} \quad \text{.....(25)}$$

6.2.3.2.6 ISP 系数的量化

ISP 系数在量化之前先要转换为频域的 ISF 系数。ISF 系数表达式见式(26):

$$f_i = \begin{cases} \frac{f_s}{2\pi} \arccos(q_i), & i=0,\dots,14 \\ \frac{f_s}{4\pi} \arccos(q_i), & i=15 \end{cases} \quad \text{.....(26)}$$

式中:

$f_i \in [0,6.4]$ kHz —— ISF 系数;

$f_s = 12.8$ kHz —— 采样频率。

ISF 矢量表示为 $f^t = [f_0, f_1, \dots, f_{15}]$, t 表示矢量的转置。

用一阶 MA 预测法,先求出当前帧的 ISF 预测残差矢量,然后量化 ISF 预测残差矢量。

定义 $z(n)$ 为去均值后的当前帧 ISF 矢量。预测残差矢量为 $r(n)$,其表达式见式(27)和式(28):

$$z(n) = isf_n - mean_isf \quad \text{.....(27)}$$

$$r(n) = z(n) - p(n) \quad \text{.....(28)}$$

式中:

isf_n —— 第 n 帧的 ISF 矢量;



$mean_isf$ —— ISF 矢量均值;

$p(n)$ —— 第 n 帧的预测 ISF 矢量,由一阶 MA 预测法得到,见式(29):

$$p(n) = \frac{1}{3}\hat{r}(n-1) \quad \text{.....(29)}$$

式中:

$\hat{r}(n-1)$ —— 前一帧量化后的 ISF 残差矢量。

为了利用 ISF 系数的帧内相关性,将 16 个 ISF 残差系数(矢量 VQ1)按照其索引号的奇偶顺序分为两组,如下所示:

分组 1: $res_isf_0, res_isf_2, res_isf_4, res_isf_6, res_isf_8, res_isf_{10}, res_isf_{12}, res_isf_{14}$

分组 2: $res_isf_1, res_isf_3, res_isf_5, res_isf_7, res_isf_9, res_isf_{11}, res_isf_{13}, res_isf_{15}$

将 $res_isf_0, res_isf_2, res_isf_4$ 三个系数组成子矢量 VQ2, $res_isf_6, res_isf_8, res_isf_{10}$ 三个系数组成子矢量 VQ3, 分别对 VQ2 和 VQ3 进行矢量量化, VQ2 矢量量化需要 10 比特, VQ3 矢量量化需要 9 比特, 见式(30):

VQ2 = {res_isf₀, res_isf₂, res_isf₄} 和 VQ3 = {res_isf₆, res_isf₈, res_isf₁₀}(30)
 量化采用的误差准则为均方量化误差准则,见式(31):

量化得到对应的最佳量化矢量为 $VQ'2$ 和 $VQ'3$, 见式(32):

VQ'2 = {res_isf₀^q, res_isf₂^q, res_isf₄^q} 和 VQ'3 = {res_isf₆^q, res_isf₈^q, res_isf₁₀^q}(32)

利用 VQ'2 的 $res_isf_0^q$ 和 $res_isf_2^q$ 对 VQ1 的 res_isf_1 系数进行预测，并计算相应的残差 $res_isf'_1$ ，见式(33)：

$$res_isf_1^{predict} = \theta + \alpha_0 \times res_isf_0^q + \beta_2 \times res_isf_2^q \quad(33)$$

式中：

θ ——ISF 系数均值;

α_0 , β_2 ——预测系数。

将矢量 VQ1 中的 $res_isf_{12}, res_isf_{14}$ 和 $res_isf'_{11}$ 组成子矢量 VQ4，并对矢量 VQ4 进行矢量量化，得到对应的最佳量化矢量 $VQ'4$ ，VQ4 矢量量化需要 9 比特，见式(34)：

$$\begin{aligned} \text{VQ4} &= \{res_isf_{12}, res_isf_{14}, res_isf'_{1}\} \\ \text{VQ}'4 &= \{res_isf'^{q}_{12}, res_isf'^{q}_{14}, res_isf'^{q}_{1}\} \end{aligned} \quad \dots \dots \dots \quad (34)$$

至此 16 维矢量 VQ1 中已经有下列 ISF 残差系数进行了矢量量化：

量化前:res_isf₀,res_isf₂,res_isf₄,res_isf₆,res_isf₈,res_isf₁₀,res_isf₁₂,res_isf₁₄,res_isf₁₆

量化后:res_isf^q₁,res_isf^q₂,res_isf^q₃,res_isf^q₄,res_isf^q₅,res_isf^q₆,res_isf^q₇,res_isf^q₈,res_isf^q₉,res_isf^q₁₀,res_isf^q₁₁,res_isf^q₁₂,res_isf^q₁₃,res_isf^q₁₄,res_isf^q₁₅

剩余的 7 个未量化 ISF 残差系数组成子矢量 VQ5, 利用上述量化的 ISF 残差系数对 VQ5 进行预测, 并计算相应的残差子矢量 VQ6, 见式(35):

$$res_isf_i^{predict} = \theta_i + \alpha_{i-1} \times res_isf_{i-1}^q + \beta_{i+1} \times res_isf_{i+1}^q, i=3,5,7,9,11,13 \quad(35)$$

武中

res_isf_i^{predict} —— 第 *i* 个 ISF 残差系数的预测值;

$res_isf'_i$ ——第 i 个 ISF 残差系数与其预测值的残差；

θ_i ——ISF 系数均值；

α_{i-1} 和 β_{i+1} —— 预测系数。

将 VQ6 分为如下两个矢量, 见式(36):

$$\begin{aligned} \text{VQ7} &= \{res_isf'_3, res_isf'_5, res_isf'_7\} \\ \text{VQ8} &= \{res_isf'_9, res_isf'_11, res_isf'_13, res_isf'_15\} \end{aligned} \quad \dots \dots \dots \quad (36)$$

对 VQ7 矢量和 VQ8 矢量分别进行矢量量化, VQ7 需要 9 比特量化, VQ8 需要 9 比特量化。

对 16 维 ISF 残差系数进行矢量量化总共需要的比特数:VQ2 需要 10 比特,VQ3 需要 9 比特,VQ4 需要 9 比特,VQ7 需要 9 比特,VQ8 需要 9 比特。总共需要 46 比特。

6.2.3.2.7 ISP 系数的插值

定义 $q^{(n)}$ 是第 n 帧 LP 分析得到的 ISP 矢量, $q^{(n-1)}$ 是第 $n-1$ 帧 LP 分析得到的 ISP 矢量。每个子

帧的 ISP 矢量 $q_i^{(n)}$ 插值见式(37)：

$$\begin{aligned} q_1^{(n)} &= 0.55q^{(n-1)} + 0.45q^{(n)} \\ q_2^{(n)} &= 0.2q^{(n-1)} + 0.8q^{(n)} \\ q_3^{(n)} &= 0.04q^{(n-1)} + 0.96q^{(n)} \\ q_4^{(n)} &= q^{(n)} \end{aligned} \quad \dots \dots \dots \quad (37)$$

得到每个子帧的 ISP 系数后，再将 ISP 系数转换为 LP 系数得到每个子帧的 LP 滤波器。

上面的插值公式既用于量化前的 ISP 系数，也用于量化后的 ISP 系数。

6.2.3.3 感知加权

对信号进行感知加权滤波处理,感知加权处理后的输出信号用于后续处理环节。

感知加权滤波器形式见式(38):

$$W(z) = \frac{A(z/\gamma_1)}{H_{\text{emph}}(z)} \quad \dots \dots \dots \quad (38)$$

式中：

$$\gamma_1 = 0.92;$$

$H_{\text{emph}}(z)$ ——预加重滤波器。

当信号的高频能量小于低频能量时,进行高频预加重滤波,用来提升信号的高频部分,预加重滤波器见式(39):

当信号的低频能量小于高频能量时,进行低频预加重滤波,用来提升信号的低频部分,预加重滤波器见式(40):

预加重滤波器系数 μ_1 取 0.68, μ_2 取 0.18。在高频预加重和低频预加重模式切换时,为了避免出现切换噪声,需要进行过渡平滑。具体为将预加重滤波器的系数在一定范围内渐次平滑过渡到另一模式。此外,编码器端需要编码 1 比特预加重模式信息,以告诉解码器感知加权滤波器中预加重采用的模式。规定此标志位为 0 时,表示高频预加重,为 1 时,表示低频预加重。

解码时使用逆感知加权滤波器,逆感知加权滤波器见式(41):

$$\frac{1}{W(z)} = \frac{H_{\text{emph}}(z)}{A(z/\gamma_1)} \quad \dots \dots \dots \quad (41)$$

6.2.3.4 ACELP 激励编码

6.2.3.4.1 开环基音搜索

6.2.3.4.1.1 自相关函数序列的计算

开环基音搜索每两个子帧估计一次基音周期。进行开环基音搜索是为了估计出一个比较准确的基音周期，从而降低闭环基音周期搜索的复杂度。

开环基音周期搜索基于感知加权后的信号进行分析。加权信号 $s_w(n)$ 在进行基音周期搜索之前，先使用四阶 FIR 滤波器 $H_{decim2}(z)$ 进行滤波，然后再进行 2 倍下采样处理，得到信号 $s_{w\downarrow}(n)$ 进行开环基音周期搜索。

归一化的自相关函数,计算如式(42):

$$corr' = \frac{\sum_{n=0}^{63} s_{wd}(n)s_{wd}(n - delay)}{\sqrt{\sum_{n=0}^{63} s_{wd}^2(n) \sum_{n=0}^{63} s_{wd}^2(n - delay)}} \quad \dots \dots \dots \quad (42)$$

式中：

$s_{wd}(n)$ —— 感知加权域中降采样信号；

$delay$ —— 基音周期候选值，搜索范围同内部采样频率 F_s 相关，当 $F_s = 25.6$ kHz 时，范围为 19~115；

$corr'$ —— 对应于该候选值的自相关函数。

为了降低复杂度，计算 $corr = sign(corr') \times corr' \times corr'$ 代替式(42)中自相关函数，其中 $sign(corr')$ 代表 $corr'$ 的符号。

对于每个基音周期候选值，计算自相关函数值，然后从中选取最多六个满足以下关系的基音周期候选值：

- 该候选值对应的自相关函数值大于前一个候选值对应的自相关函数值，并且大于后一个候选值对应的自相关函数值；
- 对于满足条件 a) 的基音周期候选值，选取其对应自相关函数最大的最多六个值（若出现自相关函数值相同，则候选值较小的优先）按其对应的自相关函数值从大到小排列保存，并保存其对应的基音周期候选值序列。

由以上两个条件确定有序自相关函数序列 $maxcorr[6]$ 以及对应基音周期候选值序列 $peakpos[6]$ 。

6.2.3.4.1.2 基音周期全局参考确定

引入基音周期全局参考 $global_pitch$ 进行辅助判断，使基音周期具有平滑性。基音周期全局参考确定方法如图 30 所示，具体描述如下：

利用 6.2.3.4.1.1 确定的基音周期候选值序列 $peakpos[6]$ 以及自相关函数序列 $maxcorr[6]$ 。

首先选择与前帧的基音周期全局参考接近的基音周期候选值，对其相应的自相关函数值乘以 1.2 进行加权。重新排列基音周期候选值序列 $peakpos[6]$ 以及自相关函数序列 $maxcorr[6]$ 。

然后对 $peakpos[6]$ 以及 $maxcorr[6]$ 消除基音周期加倍。倍周期的消除采用的是固定加权方法，其目的是找出一个最佳基音周期候选值，算法为：

- 设定最佳的基音周期候选值为自相关函数最大值对应的基音周期候选值。考察基音周期候选序列，对每一个基音周期候选值，选择一个自相关函数值的缩放因子。根据基音周期候选值的大小来选择缩放因子，当基音周期候选值大于阈值 25 时，选择缩放因子为 1.2；否则，选择缩放因子为 1.11；
- 比较该基音周期候选值对应的自相关函数值与自相关函数值序列中的最大值和缩放因子的比值，若同时满足：
 - 当前考虑的基音周期候选值小于当前最佳的基音周期；
 - 当前基音周期候选值对应的自相关函数值大于自相关函数值序列中的最大值和缩放因子的比值。

则设定基音周期最佳候选值为当前的基音周期候选值。如此循环，直至基音周期候选值序列中的每一个基音周期候选值计算完成；

- 判断自相关函数序列中的最大值对应的基音周期候选值是否为当前最佳基音周期候选值的加倍。若是，保持当前的基音周期最佳候选值；否则，设定自相关函数序列中最大值对应的基音周期候选值作为最佳基音周期候选值。

得到了最佳基音周期候选值后，要进行可靠的基音周期全局参考确定。确定基音周期全局参考的算法为：

- 满足以下四个条件之一，即可确定可靠的基音周期参考：

- 基音周期候选值序列中，自相关函数最大值并不是最佳基音周期候选对应的自相关函数

- 值的加倍,并且最佳的基音周期候选值同当前的基音周期全局参考(延续前一帧)的差值绝对值小于 8;
- 2) 自相关函数序列中的最大值与其他值的比值均大于 1.7;
 - 3) 基音周期候选值序列中存在基音周期候选值为最佳基音周期候选值的加倍;
 - 4) 当前的基音周期全局参考(延续前一帧)是当前最佳基音周期候选值的加倍,并且自相关函数的最大值要大于阈值 0.36。
- b) 如果当前帧能确定可靠的基音周期参考,则为新的基音周期全局参考;否则,当前帧要延续前一帧的基音周期全局参考。如果满足以下三个条件之一,则强制基音周期全局参考为 0:
- 1) 自相关函数最大值小于 0.15;
 - 2) 保持基音周期全局参考的帧数超过 2 帧;
 - 3) 弱自相关函数的帧数超过 1 帧。

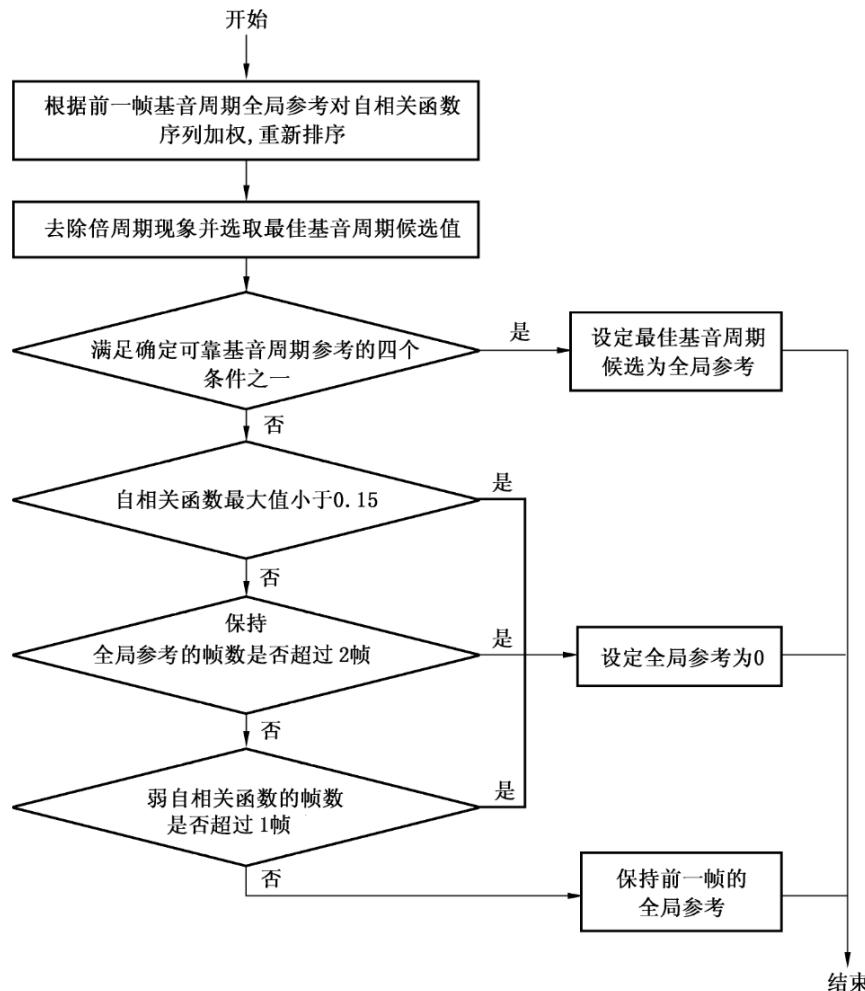


图 30 基音周期全局参考确定流程

6.2.3.4.1.3 基音周期最终确定

在确定基音周期时候,利用自相关函数序列以及其对应的基音周期候选序列,将分三种情况确定最终的基音周期,具体方法如图 31 所示。

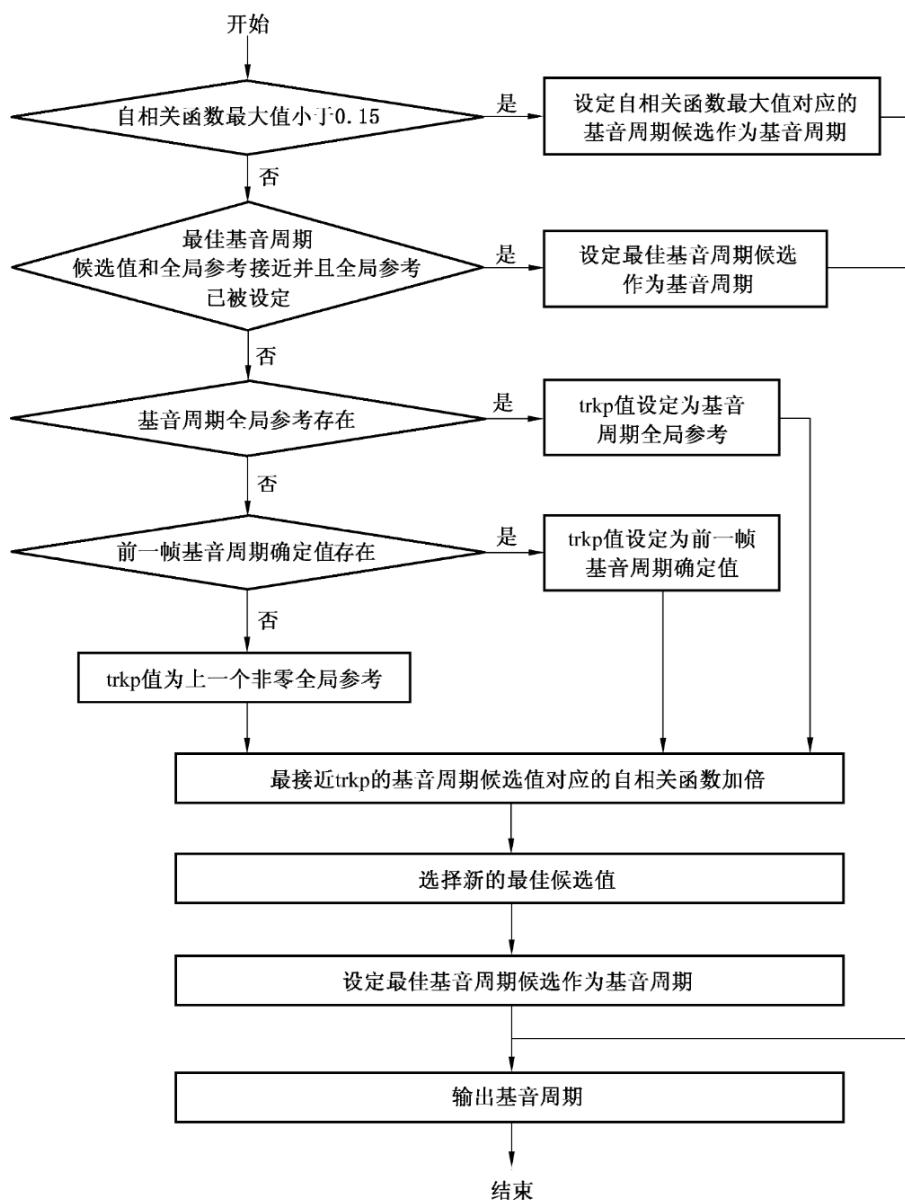


图 31 基音周期确定流程

- 基音周期最佳候选值与基音周期全局参考的差值的绝对值小于 5，并且基音周期全局参考大于 12。此时直接输出最佳基音周期值作为基音周期；
- 最大自相关函数值小于阈值 0.15，语音段信号的相关程度比较小，不易判断出明显的基音周期。当前的基音周期搜索没有实际意义，只是为闭环基音搜索提供一个最大程度去除长时相关性的参考。因而直接输出自相关函数最大值对应的基音周期搜索候选值作为基音周期；
- 无法明显判断基音周期。

引入一个基音周期确定参考值(trkp)，该值用来对最后的基音周期的确定起到参考的作用，该值确定的步骤如下：

- 若基音周期全局参考非零，trkp 值设定为基音周期的全局参考；
- 否则，若前一帧基音周期确定值非零，则 trkp 值设定为前一帧基音周期确定值；
- 否则，trkp 值设定为上一个不为 0 的基音周期全局参考。若上一个不为 0 的基音周期全局参考保持超过 3 帧，则 trkp 值强制为 0。

利用上述的条件确定的 $trkp$ 值,对整个基音周期候选序列进行搜索。找到基音周期候选值最接近 $trkp$ 的一个值,将其对应的自相关函数加倍,并重新排序自相关函数值。最后将自相关函数最大的基音周期候选值作为基音周期输出。

6.2.3.4.2 脉冲响应计算

脉冲响应是指感知加权合成滤波器(见式(43))的脉冲响应 $h(n)$ 。脉冲响应每个子帧计算一次,将滤波器 $A(z/\gamma_1)$ 的系数用零扩展后,通过滤波器 $1/\hat{A}(z)$ 和滤波器 $1/H_{\text{emph}}(z)$ 得到脉冲响应 $h(n)$ 。

$$H(z)W(z) = \frac{A(z/\gamma_1)}{\hat{A}(z)H_{\text{emph}}(z)} \quad \dots \dots \dots \quad (43)$$

6.2.3.4.3 目标信号计算

目标信号 $x(n)$ 定义为加权信号 $s_w(n)$ 与感知加权合成滤波器 $H(z)W(z)$ 零输入响应 \hat{s}_0 的差, 见式(44):

6.2.3.4.4 自适应码书

6.2.3.4.4.1 闭环基音周期搜索

闭环基音搜索的准则是使原始信号和重建信号之间加权均方误差最小,即使式(45)中的 $R(k)$ 最大。

$$R(k) = \frac{\sum_{n=0}^{63} x(n)y_k(n)}{\sqrt{\sum_{n=0}^{63} y_k^2(n)}} \quad \dots \dots \dots \quad (45)$$

式中：

$x(n)$ —— 目标信号；

$y_k(n)$ —— 延时 k 的滤波激励(即过去激励与 $h(n)$ 的卷积, $y_k(n) = h(n) \otimes exc(n - k)$)。

搜索范围限制在开环基音搜索值附近,对于第一和第三子帧,使用相应开环基音值 T_{op} 附近的值,而对第二和第四子帧,使用前一子帧分数延时 T_1 的整数部分 $\text{floor}(T_1)$ 附近的值。

先计算搜索范围的第一个延时 t_{\min} 的卷积 $y_k(n)$, 对其他整数延时 $k = t_{\min} + 1, \dots, t_{\max}$ 用式(46)递归计算:

式中：

$exc(k)$, $k = -(231 + 17), \dots, 63$ ——激励缓冲器的值;

$$y_{k-1}(-1) = 0.$$

在搜索阶段 $exc(n), n=0, \dots, 63$ 是未知的,而且只有基音延迟小于 64 才需要,为使搜索简单化,将 LP 残差存入 $exc(n)$ 使式(46)对所有延迟都有效。

确定最佳整数闭环延时后，则在最佳整数闭环延时附近按 $1/4$ 样本分辨率搜索分数基音延时。内插归一化系数 $R(k)$ ，并搜索其最大值得到的分数基音周期。搜索使用的 FIR 插值滤波器为海明窗 $sinc$ 函数，截断在 ± 15 处，滤波器的截止频率(-3 dB)为 5.063 kHz。

确定分数基音延时后,在其对应的整数延时 k 和小数延时 t 处内插过去的激励 $exc(n)$ 来计算自适应码书激励 $v(n)$,见式(47):

式中：

内插滤波器 b_{64} ——海明窗 sinc 函数, 截断在 ± 63 处, 滤波器的截止频率(-3 dB)为 6.016 kHz。

6.2.3.4.4.2 自适应码书激励的滤波

由于宽带信号的周期性不一定会扩展到高频部分,所以为了改善宽带信号下自适应码书的性能,需要对自适应码书激励信号进行滤波。

首先,对式(47)计算得到的自适应码书激励 $v(n)$ 做低通滤波得到其低频部分 $v_{low}(n)$,其计算过程如式(48):

式中：

$$b(-1) = b(1) = 0.26, b(0) = 0.48.$$

然后,计算自适应码书激励 $v(n)$ 的高频部分 $v_{high}(n)$, 见式(49):

式中：

$s(n)$ ——经过预加重的信号；

\hat{a}_i ——量化的线性预测系数。

比较相关 $corr$ 与给定阈值 $\alpha = 0.19$ 的大小，并计算自适应码书增益：

- a) 若 $corr > \alpha$ ，则最终的自适应码书激励信号为 $v(n)$ ；令加权合成信号为 $synth(n)$ ，则有 $synth(n) = h(n) \otimes v(n)$ ，此时增益 g_p 见式(52)，其中 $x(n)$ 为目标信号：

$$g_p = \frac{\sum_{n=0}^{63} x(n) \times synth(n)}{\sum_{n=0}^{63} synth^2(n)} \quad \dots \dots \dots \quad (52)$$

- b) 若 $corr \leqslant \alpha$, 则最终的自适应码书激励信号为 $v_low(n)$; 令加权合成信号为 $synth'(n)$, 则有 $synth'(n) = h(n) \otimes v_low(n)$ 。此时增益大小 g_p 见式(53):

$$g_p = \frac{\sum_{n=0}^{63} x(n) \times synth'(n)}{\sum_{n=0}^{63} (synth'(n))^2} \quad \dots \dots \dots \quad (53)$$

于 60 Hz, 来保证 LP 滤波器的稳定。如果检测到 LP 滤波器可能处于不稳定状态, 则限制 g_p 不超过 0.95, 以防止滤波器发散。

在编码码流中使用 1 比特来标识自适应码书激励信号是否使用低通滤波。

6.2.3.4.5.4 代数码书

6.2.3.4.5.4.1 代数码书结构

代数码书结构采用的是正负交错脉冲设计。每个子帧的 64 个样本位置被分为 4 个轨道, 每个轨道 16 个位置。每个轨道的脉冲个数由对应的码率所决定, 具体码书结构见表 142 所示。

表 142 代数码书结构

| 轨道 | 位置 |
|----|---|
| 1 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

6.2.3.4.5.4.2 10.8 kbps 模式

10.8 kbps 模式下, 代数码书矢量有 4 个脉冲, 每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道, 每个轨道包含 1 个脉冲, 见表 143 所示(脉冲 P_i 的序号 i 表示搜索顺序)。

表 143 10.8 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|-------|---|
| 1 | P_0 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P_1 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P_2 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P_3 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

每个轨道中 1 个脉冲的位置需要 4 比特编码, 脉冲符号用 1 比特编码, 4 个脉冲总共需要 $4 \times (4+1)=20$ 比特。

6.2.3.4.5.4.3 12.4 kbps 模式

12.4 kbps 模式下, 代数码书矢量有 6 个脉冲, 每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道, 其中轨道 1 和轨道 2 各包含 2 个脉冲, 其余轨道各包含 1 个脉冲, 见表 144。

表 144 12.4 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|------------|---|
| 1 | P_0, P_1 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P_2, P_3 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P_4 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P_5 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

轨道 1 和轨道 2 中各有 2 个脉冲,各需要 $2 \times 4 + 1 = 9$ 比特。轨道 3 和轨道 4 中各有 1 个脉冲,各需要 $4 + 1 = 5$ 比特,6 个脉冲总共需要 $2 \times 9 + 2 \times 5 = 28$ 比特。

6.2.3.4.5.4 14.0 kbps 模式

14.0 kbps 模式下,代数码书矢量有 8 个脉冲,每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道,每个轨道各包含 2 个脉冲,见表 145。

表 145 14.0 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|-------|---|
| 1 | P0,P1 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P2,P3 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P4,P5 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P6,P7 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

每个轨道中 2 个脉冲共需要 9 比特编码,8 个脉冲总共需要 $4 \times 9 = 36$ 比特。

6.2.3.4.5.5 15.6 kbps 模式

15.6 kbps 模式下,代数码书矢量有 10 个脉冲,每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道,其中轨道 1 和轨道 2 各包含 3 个脉冲,轨道 3 和轨道 4 各包含 2 个脉冲,见表 146。

表 146 15.6 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|----------|---|
| 1 | P0,P1,P2 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P3,P4,P5 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P6,P7 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P8,P9 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

轨道 1 和轨道 2 中各有 3 个脉冲,各需要 13 比特编码,轨道 3 和轨道 4 中各有 2 个脉冲,各需要 9 比特,10 个脉冲总共需要 $2 \times 13 + 2 \times 9 = 44$ 比特。

6.2.3.4.5.6 17.2 kbps 模式

17.2 kbps 模式下,代数码书矢量有 12 个脉冲,每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道,每个轨道各包含 3 个脉冲,见表 147。

表 147 17.2 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|------------|---|
| 1 | P0,P1,P2 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P3,P4,P5 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P6,P7,P8 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P9,P10,P11 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

每个轨道中 3 个脉冲各需要 13 比特编码,12 个脉冲总共需要 $4 \times 13 = 52$ 比特。

6.2.3.4.5.7 19.6 kbps 模式

19.6 kbps 模式下,代数码书矢量有 16 个脉冲,每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道,每个轨道各包含 4 个脉冲,见表 148。

表 148 19.6 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|-----------------|---|
| 1 | P0,P1,P2,P3 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P4,P5,P6,P7 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P8,P9,P10,P11 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P12,P13,P14,P15 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

每个轨道各有 4 个脉冲,各需要 16 比特编码,16 个脉冲总共需要 $4 \times 16 = 64$ 比特。

6.2.3.4.5.8 21.2 kbps 模式

21.2 kbps 模式下,代数码书矢量有 18 个脉冲,每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道,其中轨道 1 和轨道 2 各包含 5 个脉冲,轨道 3 和轨道 4 各包含 4 个脉冲,见表 149。

表 149 21.2 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|-----------------|---|
| 1 | P0,P1,P2,P3,P4 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P5,P6,P7,P8,P9 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P10,P11,P12,P13 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P14,P15,P16,P17 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

轨道 1 和轨道 2 中各有 5 个脉冲,各需要 20 比特编码,轨道 3 和轨道 4 中各有 4 个脉冲,各需要 16 比特,18 个脉冲总共需要 $2 \times 20 + 2 \times 16 = 72$ 比特。

6.2.3.4.5.9 24.4 kbps 模式

24.4 kbps 模式下,代数码书矢量有 24 个脉冲,每个脉冲的幅度为 +1 或 -1。每个子帧的 64 个位置被分为 4 个轨道,每个轨道各包含 6 个脉冲,见表 150。

表 150 24.4 kbps 代数码书脉冲结构

| 轨道 | 脉冲 | 位置 |
|----|-------------------------|---|
| 1 | P0,P1,P2,P3,P4,P5 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2 | P6,P7,P8,P9,P10,P11 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3 | P12,P13,P14,P15,P16,P17 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4 | P18,P19,P20,P21,P22,P23 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

每个轨道各有 6 个脉冲,各需要 22 比特编码,24 个脉冲总共需要 $4 \times 22 = 88$ 比特。

6.2.3.4.6 代数码书搜索前的预滤波

6.2.3.4.7 代数码书搜索

代数码书搜索是按子帧进行,其搜索准则是使加权的输入信号和加权的合成信号之间的均方误差最小。搜索前需要对自适应码书闭环搜索中使用的目标矢量进行更新,具体方法是将原来的目标矢量减去自适应码书的贡献,见式(54):

式中：

$x_0'(n)$ ——更新后的用于代数码书搜索的目标矢量；

$x_0(n)$ ——用于自适应码书闭环搜索的第一级目标矢量；

g_p ——未量化的自适应码书增益；

$y_1(n) = v(n) \otimes h(n)$ ——自适应码矢量与感知加权合成滤波器脉冲响应的卷积。

令 g_c 表示未量化的代数码书增益, $y_2(n) = c(n) \otimes h(n)$ 表示代数码书矢量与感知加权合成滤波器脉冲响应的卷积, 则使用式(55)计算平方误差:

式中：

N ——子帧长度。

最小的 g_c (通过 e 对 g_c 的偏导数为零得到) 见式(56):

将得到的 g_c 代入式(55)得最小平方误差, 见式(57):

选择使 e_{\min} 最小的代数码书激励矢量, 即选择使式(57)右边第二项最大的代数码书激励矢量作为合成信号的最佳激励。

若索引为 k 的代数码书激励矢量为 c_k , 将式(57)右边第二项转换为矩阵形式, 见式(58):

武中

$d \equiv H' x'_0$ ——目标矢量 $x'_0(n)$ 和脉冲响应 $h(n)$ 的互相关:

$\Phi = H^t H$ —— $h(n)$ 的自相关矩阵。

矢量 d 和矩阵 Φ 在码书搜索前可预先计算, 见式(59)和式(60);

因为代数码矢量只有少量的非零脉冲,所以公式(58)中的分子表示如式(61):

式中：

m_i ——第 i 个脉冲的位置；

a_i ——第 i 个脉冲的幅度;

N_p ——脉冲的个数。

式(58)的分母表示如式(62):

由于 $d(n)$ 和 $\phi(i, j)$ 的值在码书搜索前已经计算好, a_i 的值为 1 或 -1, 所以码书搜索时仅进行简单的加减运算, 这样大幅度降低码书搜索的运算量。

为了简化搜索过程,先通过参考信号 $b(n)$ 做脉冲幅度的预判决,即设置某位置上的脉冲幅度等于 $b(n)$ 在这个位置上的符号。参考信号 $b(n)$ 的计算公式见式(63)和式(64):

$$b(n) = \frac{res'_{LTP}(n)}{\sqrt{\sum_{i=0}^{63} res'_{LTP}(i)res'_{LTP}(i)}} + \frac{d(n)}{\sqrt{\sum_{i=0}^{63} d(i)d(i)}}, \quad n=0, \dots, 63 \quad \dots\dots\dots(63)$$

式中：

$w(n)$ —— 谱倾斜滤波器 ($1 - 0.3z^{-1}$) 的脉冲响应;

$res_{LTP}(n)$ ——长时预测后的残差信号,即减去自适应码书贡献的 LP 残差信号。

参考信号 $b(n)$ 可以预测脉冲的位置,从而降低搜索复杂度。每个轨道对应 16 个不同的脉冲位置,每个不同位置对应一个 $b(n)$ 值。在同一轨道内,依据 $b(n)$ 绝对值的大小,从小到大对 $b(n)$ 进行排序,记录下 8 个 $b(n)$ 绝对值最大值对应的脉冲位置。后续进行搜索时,可以直接对这 8 个脉冲位置进行搜索。

为简化搜索过程,在码书搜索前需要用符号信息修正 $d(n)$ 和 $\phi(i, j)$ 。

第1步,计算出符号信息 $s_b(n) = \text{sign} [b(n)]$ 和信号 $d'(n) = d(n)s_b(n)$;

第2步,用符号信息修正 $\phi(i,j)$,即 $\phi'(i,j) = s_b(i)s_b(j)\phi(i,j)$ 。

则式(61)和式(62)可分别转化为式(65)和式(66):

按照从上往下顺序，依次搜索每一个轨道中的所有脉冲，在搜索完一个轨道的所有脉冲后，再开始搜索下一个轨道的脉冲。在搜索时，每次搜索确定同一轨道的两个脉冲，当一个轨道上剩余脉冲个数为一个时，可以与相邻下一轨道的第一个脉冲进行组合搜索。接着从下一轨道剩余的脉冲中确定两个脉冲继续搜索。

为了具体描述搜索方法,下面就以一个轨道两个脉冲的情形进行介绍,脉冲轨道划分见表 151。

表 151 脉冲轨道示意

| 轨道(Tx) | 脉冲 | 位置 |
|--------|-------|---|
| 1(T0) | P0,P1 | 0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60 |
| 2(T1) | P2,P3 | 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61 |
| 3(T2) | P4,P5 | 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62 |
| 4(T3) | P6,P7 | 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63 |

假设脉冲轨道 1 到 4 分别用 T0、T1、T2 和 T3 表示, 在轨道 T0 中, 搜索脉冲 P0 和 P1。脉冲 P0 的位置由脉冲位置参考信号 $b(n)$ 在 T0 中的 8 个最大值所对应的位置中进行搜索确定。脉冲 P1 在轨道 T0 中的 16 个位置进行全搜索, 总的搜索次数为 $8 \times 16 = 128$ (次)。在脉冲 P0 和 P1 搜索完成以后, 判断轨道 T0 中是否还有未搜索脉冲, 本实例中, 轨道 T0 的所有脉冲已经搜索完毕, 接下来开始搜索轨道 T1 的脉冲。最佳脉冲的判断准则是使式(58)取值最大。

在轨道 T1、T2 和 T3 中的搜索方法与轨道 T0 中的方法相同。当四个轨道的所有脉冲搜索完毕之后, 整个搜索过程结束, 输出所有脉冲的最佳位置和符号。整个搜索过程中脉冲搜索顺序为: P0-P1、P2-P3、P4-P5 和 P6-P7, 搜索次数为 $8 \times 16 \times 4 = 512$ (次)。最后, 再依次搜索脉冲起始位置(P0 的位置)在轨道 T1、T2 和 T3 的情形。总搜索次数为: $512 \times 4 = 2\,048$ (次)。

6.2.3.4.8 代数码书矢量的编码

6.2.3.4.8.1 代数码书矢量的编码过程

对每个轨道搜索出的脉冲使用分类组合索引编码, 编码过程如下:

- 对轨道上需要编码的脉冲按照位置进行统计, 获得有脉冲位置的数目 pos_num(设 pos_num 的值为 N)、有脉冲位置在轨道上的分布 P(N) 和各个有脉冲位置上的脉冲数目 SU(N);
- 按照有脉冲位置的数目 pos_num 确定第一索引 I1。第一索引表示了在相同有脉冲位置的数目条件下, 有脉冲位置在轨道上全部可能的分布情况;
- 按照有脉冲位置在轨道上的分布 P(N) 确定第二索引 I2;
- 按照各个有脉冲位置上的脉冲数目 SU(N) 确定第三索引 I3;
- 最后生成编码索引 Index(N), 编码索引包括第一、二、三索引和脉冲符号索引信息。

分类组合索引编码步骤见 6.2.3.4.8.2~6.2.3.4.8.6。图 32 给出了分类组合索引编码的处理流程。

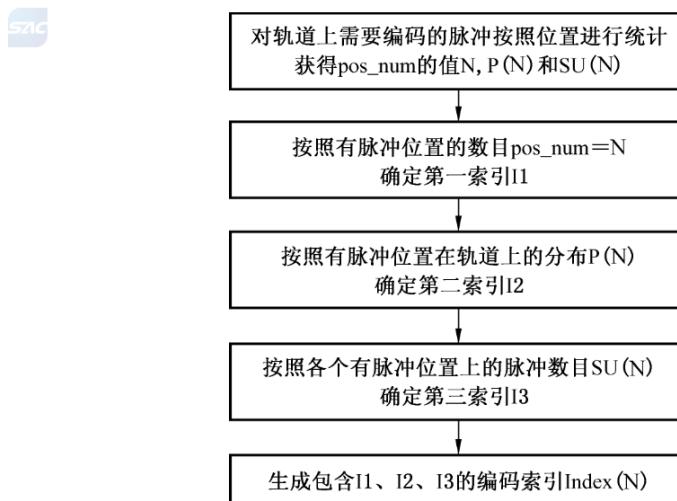


图 32 代数码书分类组合索引编码流程框图

6.2.3.4.8.2 预处理(脉冲统计)

对轨道上需要编码的脉冲按照位置进行统计,获得有脉冲位置的数目,有脉冲位置在轨道上的分布和各个有脉冲位置上的脉冲数目。

pulse_num 表示需要编码的脉冲总数,与码率模式相对应,设 $\text{pulse_num} = \omega$ 。 pos_num 表示有脉冲位置的数目,设 $\text{pos_num} = N$,由于 ω 个脉冲在轨道上的分布可能出现位置重叠,显然有 $N \in [1, \omega]$ 。有脉冲位置矢量 $P(N) = \{p(0), p(1), \dots, p(N-1)\}$ 表示有脉冲位置在轨道上的分布。 $p(n)$ 表示有脉冲位置在轨道上的位置序号, $n \in [0, N-1]$, $p(n) \in [0, M-1]$, $M=16$ 表示轨道上的位置总数,且 $0 \leq p(0) < p(1) < \dots < p(N-1) \leq 15$ 。脉冲数目矢量 $SU(N) = \{su(0), su(1), \dots, su(N-1)\}$,表示各个有脉冲位置上的脉冲数目。 $su(n)$ 表示 $p(n)$ 位置的脉冲数目, $su(0) + su(1) + \dots + su(N-1) = \omega$ 。

对轨道上需要编码的脉冲按照位置进行统计时,还需要获得各个有脉冲位置的脉冲符号信息。脉冲符号矢量 $S(N)=\{s(0), s(1), \dots, s(N-1)\}$ 表示各个有脉冲位置的脉冲符号信息, $s(n)$ 表示 $p(n)$ 位置的脉冲符号。采用 $s(n)=0$ 表示正脉冲, $s(n)=1$ 表示负脉冲的编码方法。

6.2.3.4.8.3 第一索引 I1(按位置数分类)

按照有脉冲位置的数目 pos_num=N 确定第一索引 I1。

每个轨道上共有 16 个位置。编码的脉冲数目为 6 时,有脉冲的位置数分别是 1、2、3、4、5、6,与其对应的第一索引分别为 0x1E0000、0x1D0000、0x1C0000、0x080000、0x100000、0x000000;编码的脉冲数目为 5 时,有脉冲位置的数目分别为 1、2、3、4、5,与其对应的第一索引分别为 0x000000、0x080000、0x100000、0x200000、0x40000;编码的脉冲数目为 4 时,有脉冲位置的数目分别为 1、2、3、4,与其对应的第一索引分别为 0x0000、0x2000、0x4000、0x8000;编码的脉冲数目为 3 时,有脉冲位置的数目分别为 1、2、3,与其对应的第一索引分别为 0x1C00、0x1800、0x0000;编码的脉冲数目为 2 时,有脉冲位置的数目分别为 1、2,与其对应的第一索引分别为 0x1E0、0x000;编码的脉冲数目为 1 时,有脉冲位置的数目为 1,与其对应的第一索引是 0x00。

6.2.3.4.8.4 第二索引 I2(位置编码)

第二索引 I2 指示当前有脉冲位置的分布情况。

将有脉冲位置在轨道上的分布对应为一个 N 维脉冲位置矢量 $P(N)$, 见式(67):

式中：

$P(N)$ 的可能组合样本数为 C_M^N 。这些可能的组合样本按 $p(0)$ 小的矢量排列在前, $p(0)$ 相同时, $p(1)$ 小的矢量排列在前, 后面依次类推顺序排列。这样所有可能的脉冲位置矢量就有了一个大小为 C_M^N 排序表。一个 N 维脉冲位置矢量 $P(N)$ 在排序表的位置序号就是第二索引 I_2 , 计算公式见式(68):

式中：

C_M^N —— I2 全部可能的取值数, I2 的值从 0 开始计数, 且 $I2 \in [0, C_M^N - 1]$ 。

6.2.3.4.8.5 第三索引 I3(按有脉冲位置上脉冲数量分布分类)

SU(N)与 P(N)是同维度的矢量,但受限于 $\text{su}(0)+\text{su}(1)+\dots+\text{su}(N-1)=\omega$,且 ω 的数值通常不大,一般为 1~6,因此 SU(N)的可能组合样本数 Class(N)较小。SU(N)与第三索引 I3 关系在高维度情况下采用查询关系,在低维度情况采用计算关系。在某些极端情况下,例如 $N=1$ 或 $N=\omega$,此时 SU(N)只有一种可能情况,无须由 I3 进行指示,可不编码 I3。索引 I3 的值从 0 开始计数,且 $I3 \in [0, \text{Class}(N)-1]$ 。

6.2.3.4.8.6 编码索引生成

由于 I_2, I_3 一般不能表示为 2 的整数幂, 所以 I_2, I_3 合并成 I_{23} , 合并公式见式(69):

编码索引 $Index(N)$ 还包含各个脉冲符号索引 $s(n)$ 的信息, 长度为 N 的脉冲符号矢量 $S(N)$ 字段被添加到编码索引的最后 N 位。

编码索引 $Index(N)$ 的表示见式(70):

6.2.3.4.8.7 各个脉冲数量下的索引编码的比特分配情况

各种情况下脉冲索引编码的比特详细分配情况见表 152~表 157。

表 152 6 脉冲比特分配表

| 脉冲 位置数 | 编码 起始值 | 比特 | | | | | | | | | | | | | | |
|-----------|-----------|----|----|----|----|----|------|----------|-------------|-------------|------|------|------|------|------|---|
| | | 21 | 20 | 19 | 18 | 17 | 16 | 15 | ... | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 6 | 0x000000 | 0 | 0 | 0 | | | | I2 | | s(0) | s(1) | s(2) | s(3) | s(4) | s(5) | |
| 5 | 0x100000 | 0 | 1 | | | | I3×4 | 368+I2 | | s(0) | s(1) | s(2) | s(3) | s(4) | | |
| 4 | 0x080000 | 0 | 0 | 1 | | | I3×1 | 820 + I2 | | s(0) | s(1) | s(2) | s(3) | | | |
| 3 | 0x1C0000 | 0 | 1 | 1 | 1 | 0 | 0 | | I3×560 + I2 | | s(0) | s(1) | s(2) | | | |
| 2 | 0x1D0000 | 0 | 1 | 1 | 1 | 0 | 1 | | | I3×120 + I2 | | | s(0) | s(1) | | |
| 1 | 0x1E0000 | 0 | 1 | 1 | 1 | 1 | 0 | | | I2 | | | | | s(0) | |

表 153 5 脉冲比特分配表

| 脉冲位置数 | 编码起始值 | 比特 | | | | | | | | | | | | | |
|-------|---------|----|----|----|----|----|----|-------------|----------|----|------|------|------|------|------|
| | | 19 | 18 | 17 | 16 | 15 | 14 | ... | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 5 | 0x40000 | 0 | 1 | | | | | I2 | | | s(0) | s(1) | s(2) | s(3) | s(4) |
| 4 | 0x20000 | 0 | 0 | 1 | | | | I3×1 | 820 + I2 | | s(0) | s(1) | s(2) | s(3) | |
| 3 | 0x10000 | 0 | 0 | 0 | 1 | | | I3×560 + I2 | | | s(0) | s(1) | s(2) | | |
| 2 | 0x08000 | 0 | 0 | 0 | 0 | 1 | | I3×120 + I2 | | | | s(0) | s(1) | | |
| 1 | 0x00000 | 0 | 0 | 0 | 0 | 0 | | | | I2 | | | | | s(0) |

表 154 4 脉冲比特分配表

| 脉冲位置数 | 编码起始值 | 比特 | | | | | | | | | | | | | |
|-------|--------|----|----|----|----|----|---------------|-----|---|---|---|------|------|------|------|
| | | 15 | 14 | 13 | 12 | 11 | 10 | ... | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 4 | 0x8000 | 1 | | | | | I2 | | | | | s(0) | s(1) | s(2) | s(3) |
| 3 | 0x4000 | 0 | 1 | | | | I3 × 560 + I2 | | | | | s(0) | s(1) | s(2) | |
| 2 | 0x2000 | 0 | 0 | 1 | | | I3 × 120 + I2 | | | | | s(0) | s(1) | | |
| 1 | 0x0000 | 0 | 0 | 0 | 0 | | | I2 | | | | | | s(0) | |

表 155 3 脉冲比特分配表

| 脉冲位置数 | 编码起始值 | 比特 | | | | | | | | | | | | | | |
|-------|--------|----|----|----|-------------|---|---|---|---|---|---|---|------|------|------|------|
| | | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| 3 | 0x0000 | 0 | I2 | | | | | | | | | | s(0) | s(1) | s(2) | |
| 2 | 0x1800 | 1 | 1 | 0 | I3×120 + I2 | | | | | | | | | | s(0) | s(1) |
| 1 | 0x1C00 | 1 | 1 | 1 | I2 | | | | | | | | | | s(0) | |

表 156 2 脉冲比特分配表

| 脉冲位置数 | 编码起始值 | 比特 | | | | | | | | |
|-------|-------|----|----|---|---|----|---|---|------|------|
| | | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 2 | 0x000 | 0 | I2 | | | I2 | | | s(0) | s(1) |
| 1 | 0x1E0 | 1 | 1 | 1 | 1 | I2 | | | I2 | |

表 157 单脉冲比特分配表

| 脉冲位置数 | 编码起始值 | 比特 | | | | | |
|-------|-------|----|---|---|---|---|------|
| | | 4 | 3 | 2 | 1 | 0 | |
| 1 | 0x00 | I2 | | | | | s(0) |

6.2.3.4.9 自适应和代数码书增益量化

每个子帧的自适应码书和代数码书增益使用 7 比特进行联合矢量量化, 其中自适应码书增益 g_p 直接量化, 代数码书增益 g_c 通过对校正因子 γ 和每帧的代数码书平均能量 \bar{E}_s 进行量化来实现。每帧的代数码书平均能量使用 2 比特量化。

设 $E_s(n)$ 是第 n 个子帧代数码书激励矢量的能量, 计算公式见式(71):

$$E_s(n) = 10 \log_{10} \left(\frac{1}{N} g_c^2 \sum_{i=0}^{N-1} c^2(i) \right) = 20 \log_{10}(g_c) + E_i \quad \dots \dots \dots (71)$$

式中:

$N=64$ ——子帧的长度;

$c(i)$ ——代数码书激励矢量;

E_i 是按下式计算的能量值, 见式(72):

$$E_i = 10 \log_{10} \left(\frac{1}{N} \sum_{i=0}^{N-1} c^2(i) \right) \quad \dots \dots \dots (72)$$

计算代数码书平均能量 \bar{E}_s 并量化, 然后用于计算代数码书预测增益 g'_c , 见式(73):

$$g'_c = 10^{0.05(\bar{E}_s - E_i)} \quad \dots \dots \dots (73)$$

上式是从公式 $\bar{E}_s = 20 \log_{10}(g'_c) + E_i$ 中推导出的。

代数码书增益 g_c 和预测增益 g'_c 之间的校正因子 γ 由式(74)得到:

$$\gamma = g_c / g'_c \quad \dots \dots \dots (74)$$

每个子帧的自适应码书增益 g_p 和代数码书增益的校正因子 γ 采用 7 比特联合矢量量化, 码书的搜

索准则使原始信号和重建信号之间加权均方误差最小,计算公式见式(75):

$$E = \frac{1}{N} \sum_{n=0}^{N-1} [x_0(n) - \hat{g}_p y_1(n) - \hat{g}_c y_2(n)]^2 \quad \dots \dots \dots \quad (75)$$

式中:

$x_0(n)$ ——自适应码书搜索的目标信号;

$y_1(n)$ ——自适应码书矢量同感知加权合成滤波器脉冲响应的卷积;

$y_2(n)$ ——代数码书矢量同感知加权合成滤波器脉冲响应的卷积;

\hat{g}_p ——量化的自适应码书增益;

\hat{g}_c ——量化的代数码书增益。

代数码书平均能量 \bar{E}_s 的计算和量化过程如下:

首先计算每个子帧 LP 预测残差的能量,见式(76):

$$E_{\text{res}}(n) = 10 \log_{10} \left(\frac{1}{N} \sum_{i=0}^{N-1} r^2(i) \right) \quad \dots \dots \dots \quad (76)$$

然后计算每帧的平均残差能量,见式(77):

$$\bar{E}_{\text{res}} = \frac{1}{4} \sum_{n=0}^3 E_{\text{res}}(n) \quad \dots \dots \dots \quad (77)$$

从残差能量中减掉自适应码书的贡献,得到每帧的代数码书平均能量 \bar{E}_s 。这通过减掉每帧开环基音搜索得到的归一化自相关能量平均值实现,见式(78):

$$\bar{E}_s = \bar{E}_{\text{res}} - 10R \quad \dots \dots \dots \quad (78)$$

式中:

R ——开环基音搜索得到的归一化自相关能量平均值。

平均能量 \bar{E}_s 每帧用 2 比特量化,有 4 个量化级别:18,30,42,54。通过每次对 \bar{E}_s 加上 12(量化索引加 1)迭代,来限制 \bar{E}_s 量化范围 $(E_{\text{max}} - 27) < \bar{E}_s \leqslant 54$ 。其中 E_{max} 是 4 个子帧中 $E_{\text{res}}(n)$ 最大值。

6.2.4 TVC 编码

6.2.4.1 TVC 编码过程

TVC 编码过程见图 33。

变换域矢量编码技术(TVC)编码步骤简述如下:

- a) 音频输入信号通过感知加权滤波器得到目标信号;
- b) 加自适应窗;
- c) 通过 FFT 变换将时域信号变换到频域;
- d) 在变换域中,进行峰值预整形和缩放因子调整,以减少低频噪声;
- e) 对预整形后的信号进行基于变长分裂表矢量量化;
- f) 增益平衡和峰值逆整形;
- g) 逆时频变换,将频域信号变换到时域,得到量化后的时域信号;
- h) 计算和量化全局增益;
- i) 加自适应窗和重叠加,以减少因变换域量化而引起的块效应;
- j) 为下一帧保存重叠信号;
- k) 重建信号通过逆感知加权滤波器和 LP 分析滤波器得到激励信号,以更新 ACELP 的自适应码书,允许 TVC 和 ACELP 模式之间切换。

编码码流需要传输四个参数:噪声因子、缩放因子、频谱的量化值、全局增益。6.2.4.2~6.2.4.12 将详细阐述编码算法工作流程。

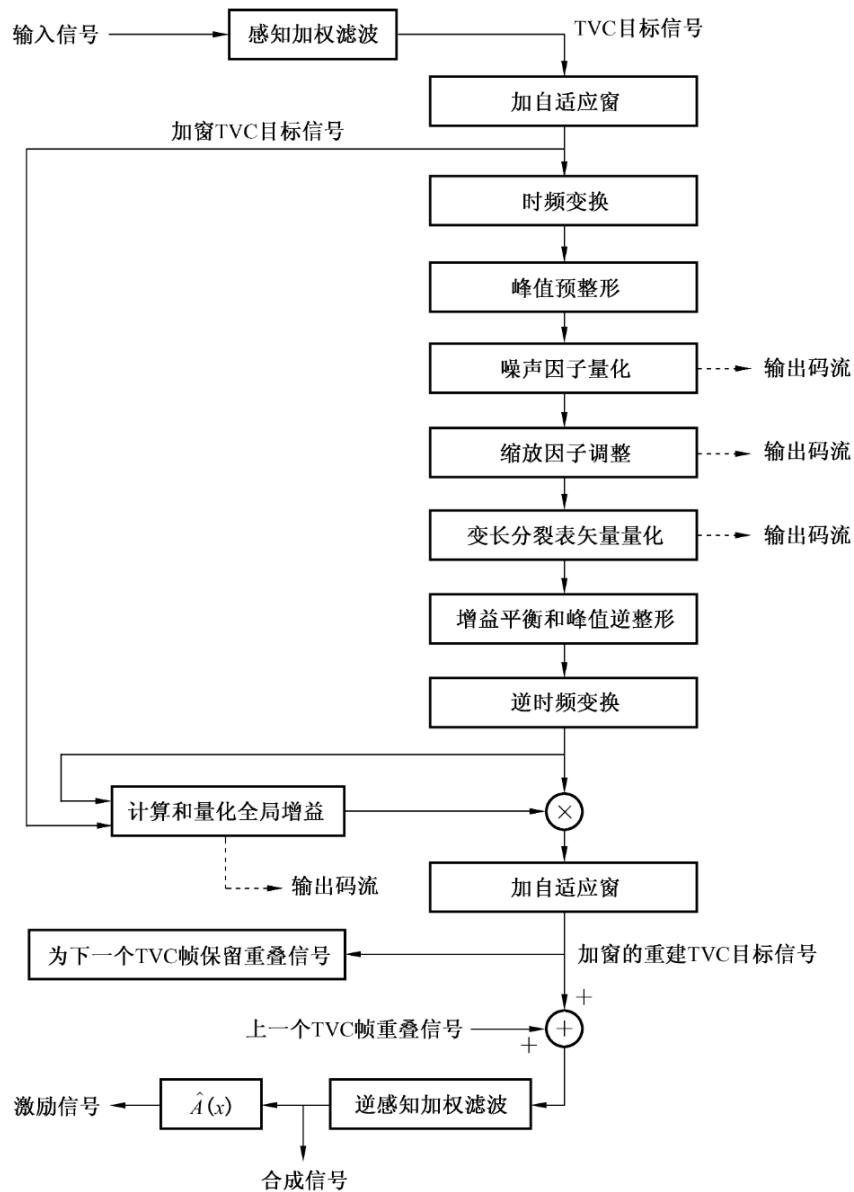


图 33 TVC 编码过程图

6.2.4.2 计算目标信号

对输入的信号进行感知加权滤波得到目标信号,后续计算都针对目标信号进行,感知加权滤波器传输函数见式(79):

$$W(z) = \frac{\hat{A}(z/\gamma_1)}{H_{\text{emph}}(z)} \quad \dots \dots \dots \quad (79)$$

式中：

$$\gamma_1 = 0.92;$$

$\hat{A}(z)$ ——由量化系数构成的 LP 滤波器；

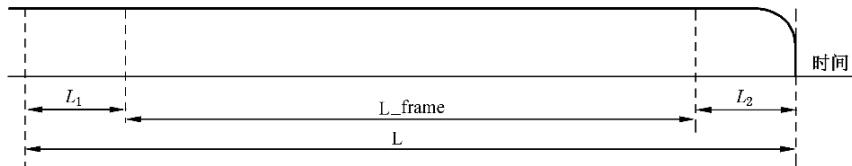
$H_{\text{emph}}(z)$ ——预加重滤波器。

感知加权滤波器的详细描述见 6.2.3.3。

6.2.4.3 加自适应窗

自适应窗的具体窗型同上一帧编码模式以及编码模式切换有关。假设 L_{frame} 表示当前帧输入信号的长度; L 表示当前帧自适应窗的长度; L_1 表示与上一帧重叠的样本长度; L_2 表示与下一帧重叠的样本长度。

上一帧使用 ACELP 编码时,自适应窗如图 34 所示。



说明：

$$L_1 = 16;$$

$$L_2 = 16;$$

```
L_frame=256;
```

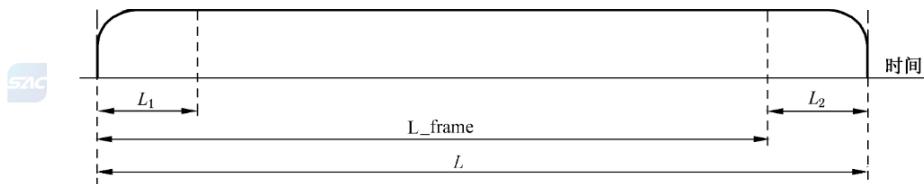
$$L=288.$$

图 34 上一帧 ACELP 编码时自适应窗

图 34 中自适应窗的窗函数见式(80):

因为当前帧与下一帧的重叠部分长度为 L_2 , 所以当下一帧还是 TVC 编码时, 下一帧帧头所加的窗长要和 L_2 长度一致。

上一帧使用 TVC 编码时,自适应窗如图 35 所示。



说明：

$L_1=16$ (上一帧由 ACELP 切换为 TVC 编码)或 32(上一帧没有发生模式切换);

$L_2 = 32$:

```
L_frame=256;
```

$$L = 288.$$

图 35 上一帧 TVC 编码时自适应窗

图 35 中对应自适应窗的窗函数见式(81):

$$w_1(n) = \sin(2\pi n/(4L_1)), \quad n=0, \dots, L_1-1$$

$$w_2(n) = 1, \quad n = 0, \dots, L - L_1 - L_2 - 1 \quad \dots \dots \dots \quad (81)$$

$$w_2(n) \equiv \cos(2\pi n/(4L_2)), \quad n = 0, \dots, L_2 - 1$$

6.2.4.4 时频变换

加窗信号通过 DFT 变换到频域, 见式(82)。

$$X(k) = \sum_{n=0}^{L_{\text{DFT}}-1} x(n) e^{-j\frac{2\pi}{L_{\text{DFT}}} nk} \quad \dots \dots \dots \quad (82)$$

式中：

6.2.4.5 峰值预整形

峰值预整形算法见图 36。

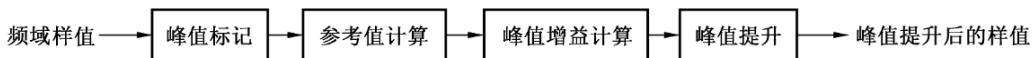


图 36 峰值预整形算法流程

处理步骤如下：

- a) 计算频谱的幅值 $M(n) = \sqrt{\text{Re}^2(n) + \text{Im}^2(n)}$;
 - b) 标记前 $1/4$ 频谱中的峰值集合 $\{p_i\}$, p_i 定义为整形频谱段幅值的局部最大值。若 $M(n) > M(m)$, $\forall m \in [n-j, n+j]$, $m \neq n$, 则 $M(n) \in \{p_i\}$ 表示一个 $2j+1$ 点局部的最大值, 实际中 j 选择为 1;
 - c) 计算参考值 $ref_{\max} = \sqrt{E_{\max}/8}$, E_{\max} 为前 $1/4$ 频谱中划分八维频谱矢量块的最大能量;
 - d) 计算峰值 p_i 的放大因子 $R_i = (ref_{\max}/p_i)^{1/2}$ 。如果 $R_i > R_{i-1}$, 则 $R_i = R_{i-1}$, 以保证放大因子的递减性;
 - e) 在峰值集合 $\{p_i\}$ 中除去 ref_{\max} 相关的峰值点, 保证 ref_{\max} 的不变性。对剩余的峰值点 p_i 进行放大 $p_i = p_i R_i$ 。

该模块通过提升低频峰值，达到减小低频中较小峰值处量化噪声的目的。由于只对少量的频谱点进行放大，对全局增益影响很小。

6.2.4.6 噪声因子量化

预整形的频谱 X 划分成 $K=N/8$ 个八维矢量。定义 B_k 为第 k 个矢量, $k=0,1,\dots,K-1$ 。计算 B_k 的能量 E_k 见式(83)。

根据 E_k 得到消耗比特数的初始估计, 见式(84):

比特消耗估计迭代：

初始条件:设 $\text{fac} = 128$, $\text{offset} = 0$ 和 $\text{nbits_max} = 0.95 \times (b_{\max} - K)$, b_{\max} 表示频谱量化可用比特数。

迭代执行 10 次：

- a) $\text{offset} = \text{offset} + \text{fac};$
 - b) $nbits = \sum_{k=1}^K \max(0, R_k(1) - \text{offset}) ;$
 - c) if($nbits \leq nbits_max$) $\text{offset} = \text{offset} - \text{fac};$
 - d) $\text{fac} = \text{fac} / 2.$

迭代过程完成后,确定了参数 offset 值,再进行下面的迭代:

初始条件: nbits = 0; n = 1

迭代过程:for (k = K / 2;k<=K-1;++k)

{

tmp = R_k(1) - offset;

if (tmp < 5)

1

`nbits = nbits + tmp;`

$$n = n \pm 1$$

1

3

迭代过程完成后, $n_{\text{bits}} \equiv n_{\text{bits}} / n$

噪声因子计算见式(85)。

$$\hat{o}_{\text{noise}} = 10^{\frac{\log_{10}(2)}{10}(\text{nbits}-5)} \quad \dots \dots \dots \quad (85)$$

噪声因子范围在 0.1~0.8 之间, 使用 3 比特量化, 量化索引为 $idx = \text{floor}((8 - 10 \times \delta_{\text{noise}}) + 0.5)$ 。

6.2.4.7 缩放因子控制

选择合适的缩放因子对整个频谱的不同部分进行调整,使各缩放因子控制的频带内量化噪声的分布更合理。TVC 采用两个缩放因子 g_1 和 g_2 分别对频谱样值进行调整,最终对调整后的频谱样值进行量化。缩放因子调整见图 37。

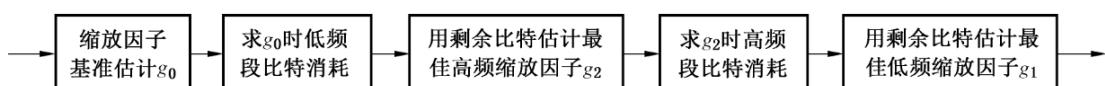


图 37 两个缩放因子的调整算法

假设频谱样值序列为 $X(0,1,\dots,N)$ ，先将整个频带等分为低频带 $X(0,1,\dots,N/2)$ 和高频带 $X(N/2+1,\dots,N)$ ，消耗比特数的估计函数为 $b = \text{cons}(X, N, g)$ ， b_{\max} 表示频谱量化可用比特数。按照下列步骤调整缩放因子：

- a) 选择 $g = g_0$, 使得全频带的比特消耗 $b_0 = \text{cons}(X(0, 1, \dots, N), N, g_0)$ 满足 $b_0 < b_{\max}$ 且最小化 $(b_{\max} - b_0)$, 则 g_0 将作为两个缩放因子调整的基准值;
 - b) 计算低频带的比特消耗 $b_l = \text{cons}(X(0, 1, \dots, N/2), N/2, g_0)$;
 - c) 选择 $g = g_2$, 使得高频带的比特消耗 $b_h = \text{cons}(X(N/2, \dots, N), N/2, g_2)$ 满足 $b_h < b_{\max} - b_l$ 且最小化 $(b_{\max} - b_l - b_h)$;
 - d) 选择 $g = g_1$, 使得低频带的比特消耗 $b_l = \text{cons}(X(0, 1, \dots, N/2), N/2, g_1)$, 满足 $b_l < b_{\max} - b_h$ 且最小化 $(b_{\max} - b_l - b_h)$;
 - e) 用缩放因子 g_1 和 g_2 对序列 X 进行缩放。缩放后的频谱为 $X' = [X(0, 1, \dots, N/2)/g_1, X(N/2 + 1, \dots, N)/g_2]$, 最后将 X' 送入矢量量化器;
 - f) 对 $\frac{g_1}{g_2}$ 进行 7 比特编码传输, 为了保持编码比特数不变, 应相应减少量化可用比特数。

6.2.4.8 变长分裂表矢量量化

6.2.4.8.1 变长分裂表矢量量化过程

TVC 使用格型量化器量化经过缩放的频谱 X' ，频谱划分成八维矢量，使用由 Gosset 格子集(又称 RE_8)构成的矢量码表量化。格的生成矩阵 G 产生一个格中的所有点， $c = kG$, k 是由整数值组成的。

行矢量， c 就是产生的格点。为了生成满足给定码率的矢量码表，只考虑位于给定半径的球面上的格点。使用多个不同的半径，就可以生成多码率码表。

基于分裂表的矢量量化方法框图见图 38，图中的变量 header 表示码头，split header 表示分裂量的码头，even_flag 则表示偶标志位。基于分裂表的矢量量化方法过程如下：

- 在格中找到待编码数据 x 的最近邻点 y ；
- 判断 y 是否在基础码书 C 中，如果在，则直接计算索引 i ，并编码输出到码流；
- 若 y 不在基础码书 C 中，判断 y 是否属于 $2D_8$ 。如果 y 属于 $2D_8$ ，则将其减 1，置偶标志位 (even_flag) 为 1；如果 y 不属于 $2D_8$ ，则使用分裂表编码；
- 再判断 y 是否在基础码书 C 中，如果在，则直接计算索引 i ，并编码输出到码流；如果不在此，则使用分裂表编码；
- 使用分裂表编码时，将 y 中的每一个分量 $y(i)$ 分裂为 $c(i)$ 与分裂表中的某一个值 $y'(i)$ 的和。要选择合适的 $y'(i)$ 使得 $c(i)$ 的绝对值最小，并且使得生成的八个 $c(i)$ 组成的八维矢量为基础码书 C 中的某一矢量 c 。分裂之后再检测八个 $y'(i)$ 值的大小，若均小于等于一级扩展阈值，则采用一级扩展编码方式，否则采用二级扩展编码方式。

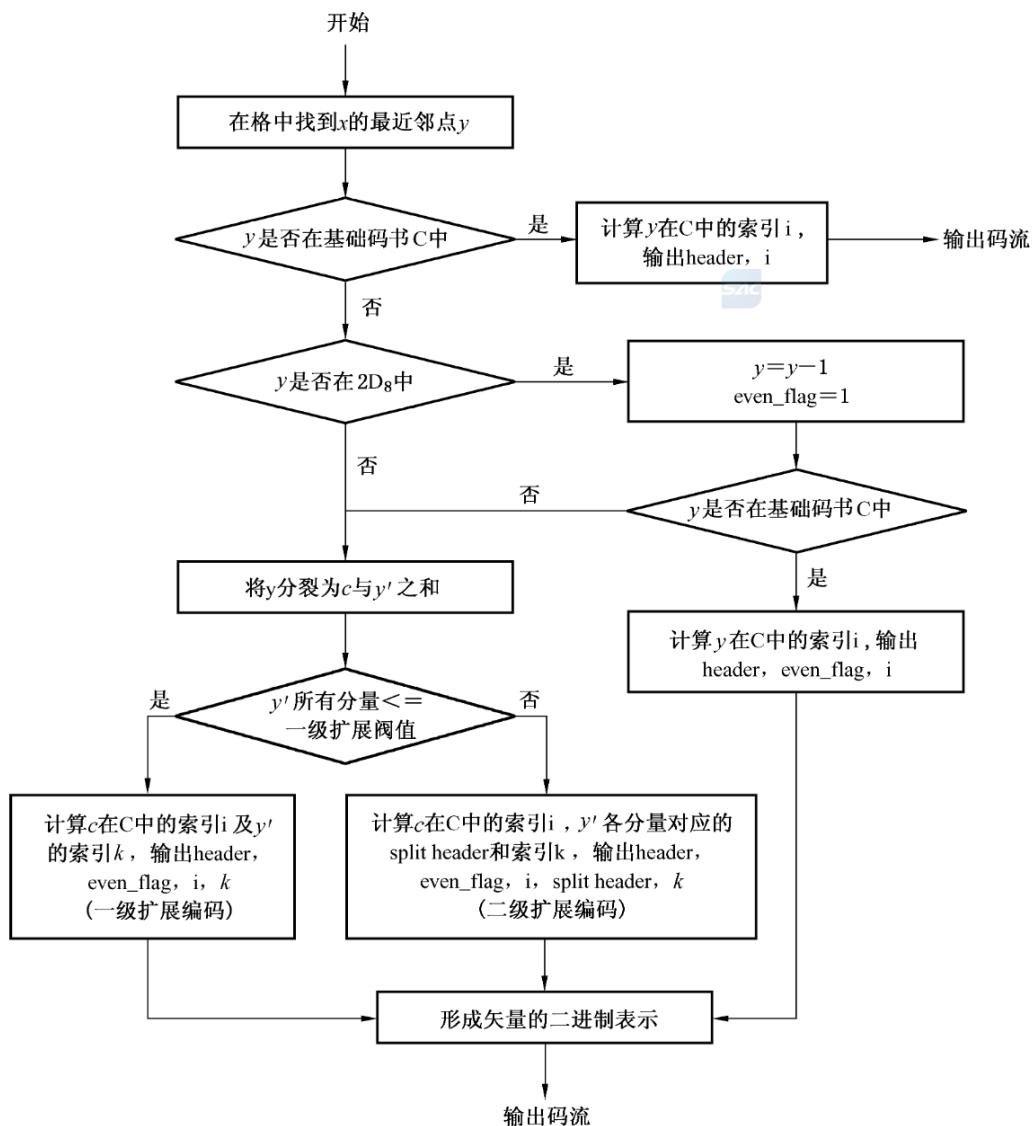


图 38 基于分裂表的矢量量化方法编码示意图

基于分裂表的矢量量化方法,首先判断待编码数据是否在基础码书中,如果在则直接利用基础码书编码;否则,尝试将其分裂为基础码书中的码字和分裂表中一个分裂量的和,再对基础码字和分裂量分别编码。详细过程见 6.2.4.8.2 至 6.2.4.8.5。

6.2.4.8.2 寻找最近邻点

将预整形后的频谱数据分组,每八个数为一组,组成一个八维的矢量 x ,在格中寻找与该矢量最接近的点,即最近邻点 y 。

6.2.4.8.3 基础码书的选取

如果编码矢量不在基础码书中,基于分裂表的矢量量化方法尝试进行分裂处理。为适应分裂处理,选取如下的 RE_8 基础码书:

$$RE_8 = 2D_8 \cup \{2D_8 + (1, 1, \dots, 1)\} \text{ 其中 } D_8 = \{(x_1, x_2, \dots, x_8) \in Z^8 \mid x_1 + \dots + x_8 \text{ 为偶}\}$$

RE_8 集合中所有数据之和是 4 的倍数,并且奇偶性相同。

基础码书 Q_0, Q_2, Q_3, Q_4 和 inv_Q_4 的特征码字定义见表 158。

表 158 基础码书的特征码字定义

| 特征码字 | Q_0 | Q_2 | Q_3 | Q_4 | inv_Q_4 |
|-------------------|-------|-------|-------|-------|------------|
| (0,0,0,0,0,0,0,0) | ✓ | | | | |
| (2,0,0,0,0,0,0,0) | | ✓ | ✓ | | |
| (1,1,1,1,1,1,1,1) | | ✓ | ✓ | | |
| (2,2,0,0,0,0,0,0) | | ✓ | ✓ | | |
| (2,2,2,2,0,0,0,0) | | | ✓ | | |
| (3,1,1,1,1,1,1,1) | | | ✓ | | |
| (4,0,0,0,0,0,0,0) | | | ✓ | | |
| (3,3,1,1,1,1,1,1) | | | | ✓ | |
| (1,1,3,3,3,3,3,3) | | | | | ✓ |
| (4,2,2,0,0,0,0,0) | | | ✓ | | |
| (3,3,3,1,1,1,1,1) | | | | ✓ | |
| (1,1,1,3,3,3,3,3) | | | | | ✓ |
| (4,4,0,0,0,0,0,0) | | | ✓ | | |
| (5,1,1,1,1,1,1,1) | | | | ✓ | |
| (3,3,3,3,1,1,1,1) | | | | ✓ | |
| (5,3,1,1,1,1,1,1) | | | | ✓ | |
| (6,2,0,0,0,0,0,0) | | | ✓ | | |
| (5,3,3,1,1,1,1,1) | | | | ✓ | |
| (5,5,1,1,1,1,1,1) | | | | ✓ | |
| (7,1,1,1,1,1,1,1) | | | | ✓ | |
| (7,3,1,1,1,1,1,1) | | | | ✓ | |
| (3,3,3,3,3,3,3,1) | | | | ✓ | |

表 158 (续)

| 特征码字 | Q_0 | Q_2 | Q_3 | Q_4 | $\text{inv_}Q_4$ |
|--------------------|-------|-------|-------|-------|-------------------|
| (3,3,3,3,3,3,3,3) | | | | ✓ | |
| (9,1,1,1,1,1,1,1) | | | | ✓ | |
| (11,1,1,1,1,1,1,1) | | | | ✓ | |
| (13,1,1,1,1,1,1,1) | | | | ✓ | |

不同的基础码书是用一个不同长度的二进制数来标识的,即基础码书的标识位 header,具体的表示方式如下:

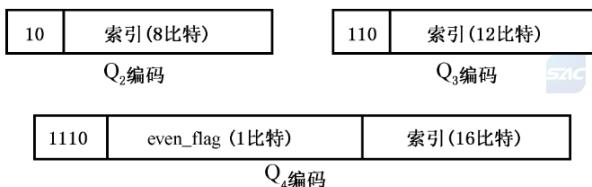
| | |
|------------------|-------------------------|
| header = 0 | 基础码书为 Q_0 |
| header = 10 | 基础码书为 Q_2 |
| header = 1100 | 基础码书为 Q_3 |
| header = 1110 | 基础码书为 Q_4 |
| header = 1111110 | 基础码书为 $\text{inv_}Q_4$ |

6.2.4.8.4 基础码书编码

基础码书编码包括直接编码,特殊特征码字编码,2D₈数据的奇化编码,以及基于缓存的常见特征码字快速码书搜索,如下:

a) 直接编码

首先在基础码书中查找格中 x 的最近邻点 y 。如果在基础码书 Q_0, Q_2, Q_3, Q_4 中,则直接计算码字 y 在基础码书中的索引 i。将索引 i 和基础码书的标识位 header 打包输出,输出格式见图 39。由于基础码书 Q_0 只包括一个特征码字(0,0,0,0,0,0,0,0),因此只编码码书的 header,不编码码书索引。

图 39 Q_2, Q_3, Q_4 输出编码格式

索引 i 的计算和编码过程如下:

- 分解初始矢量,获得符号和初始矢量绝对值;
- 对符号进行编码,获得符号编码;
- 对初始绝对值矢量进行分层组合编码,获得绝对值矢量编码;
- 组合符号编码及绝对值矢量编码,获得初始矢量的编码;
- 初始矢量的编码加上矢量所属特征码字在整个基础码书中偏移量,得到基础码书的索引。

b) 特殊特征码字的编码

在对基础码书 $\text{inv_}Q_4$ 进行搜索时,若检测到待编码数据符合特殊特征码字(1,1,1,3,3,3,3)或(1,1,3,3,3,3,3),则将其进行位反转为(3,3,3,1,1,1,1)和(3,3,1,1,1,1,1),并置反转标志位,即给数据加码头 header:“1111110”。然后按直接编码方法计算(3,3,3,1,1,1,1)和(3,3,1,1,1,1,1)在码书 Q_4 中的索引,此时的输出格式见图 40。

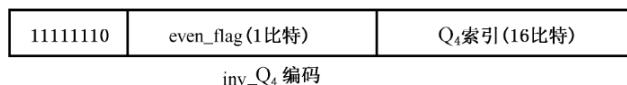


图 40 特殊特征码字的输出编码格式

c) 2D₈数据的奇化编码

在对基础码书进行搜索时,若码字 y 不在基础码书中,则还需判断它是否为 2D₈ 数据。若为 2D₈ 数据,则置偶标志位为 1,即 even_flag = 1。

置偶标志位后,将 y 的每一个分量都减 1,得到一个各分量均为奇数的矢量。此时再判断码字 y 是否在基础码书 Q₄ 中。若在,则按直接编码方法计算其在 Q₄ 中的索引作为输出;若为特殊特征码字,则按特殊特征码字的编码方法编码;若仍不在基础码书中,则使用分裂表编码。

d) 基于缓存的常见特征码字快速码书搜索

基于缓存的快速码书搜索方法,是借鉴缓存使用机理的一种 VQ 快速搜索方法。该方法基于 RE₈ 格矢量量化中八位一组有许多是重复的数据,如果能够有效击中这些重复的数据,那么将大大提高搜索速度,并在可能的范围内节省比特。编码流程如图 41 所示。主要步骤为:

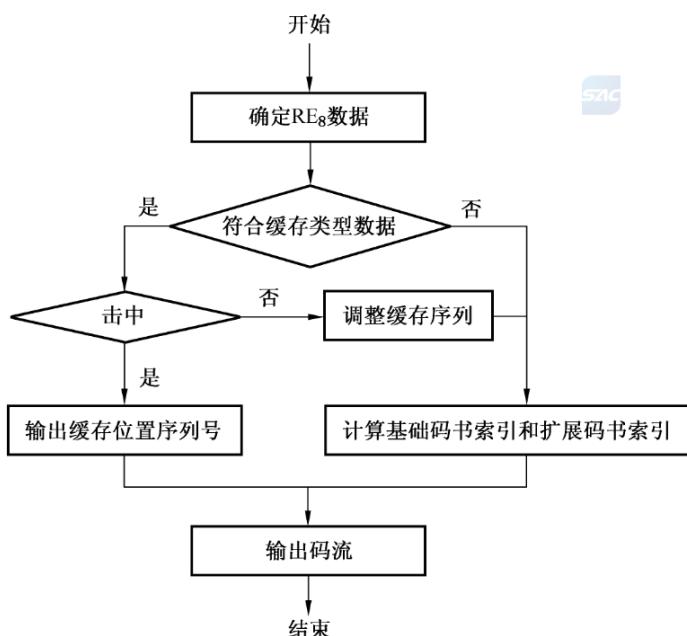


图 41 基于缓存的快速码书搜索编码流程图

第 1 步: 确定 RE₈ 数据

将预整形后的频谱数据分组,每八个数为一组,根据就近原则将这八个数量化为 RE₈ 集合上的点。

第 2 步: 确定缓存类型数据和缓存空间

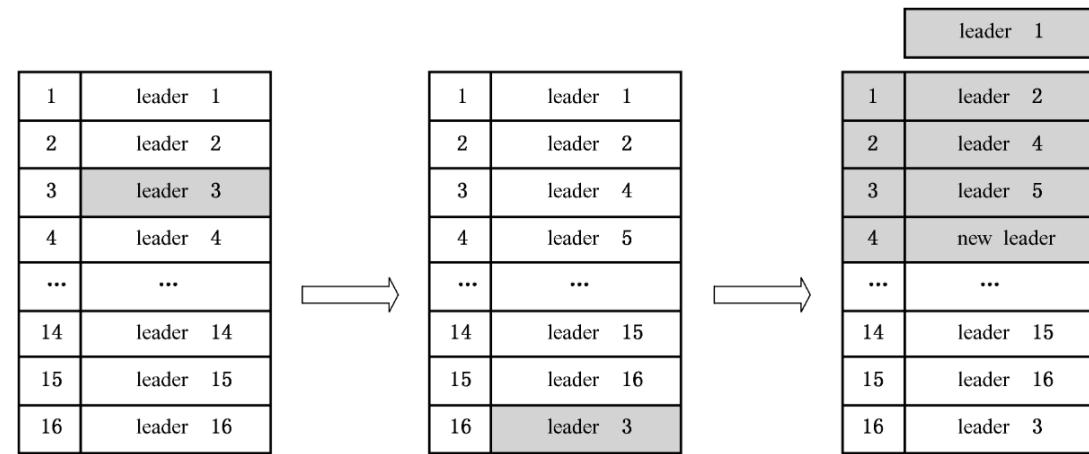
将码书中出现机率最高的特征码字选出来,遇到这类数据时便对其进行与缓存相关的操作,这些数据称之为缓存类型数据。选取特征码字为(1,1,1,1,1,1,1,1)和(2,2,0,0,0,0,0,0)的数据作为缓存类型数据。选取了 16 个空间作为缓存空间,即如果用二进制表示,需要 4 比特。

编码时遇到这类数据,在缓存中查找是否有与其相同的数据。如果待编数据不是缓存类型数据,则直接计算其码书索引并输出到码流。如果是缓存类型数据,则将其缓存序列号输出到码流。

第3步:编码时缓存中的数据操作

当判断得知待编码数据为缓存类型后,在缓存中查找是否有与其相同的数据。如果有,称之为“击中”,没有则称之为“未击中”。对于击中了的数据,输出其相应的缓存序列号;对于未击中的数据则将其放入缓存中。此时如果缓存已占满,则需要有一个数据被替换出来。采用的替换原则是:如果数据刚入缓存,则编号为4;如果缓存中的数据被击中,则沉入最底部,编号为16;编号为1的数据,最先被替换,每次数据被击中或者刚进来,编号都会有所改变。

具体操作过程见图42。leader1到leader16及New leader均为特征码字的标号。

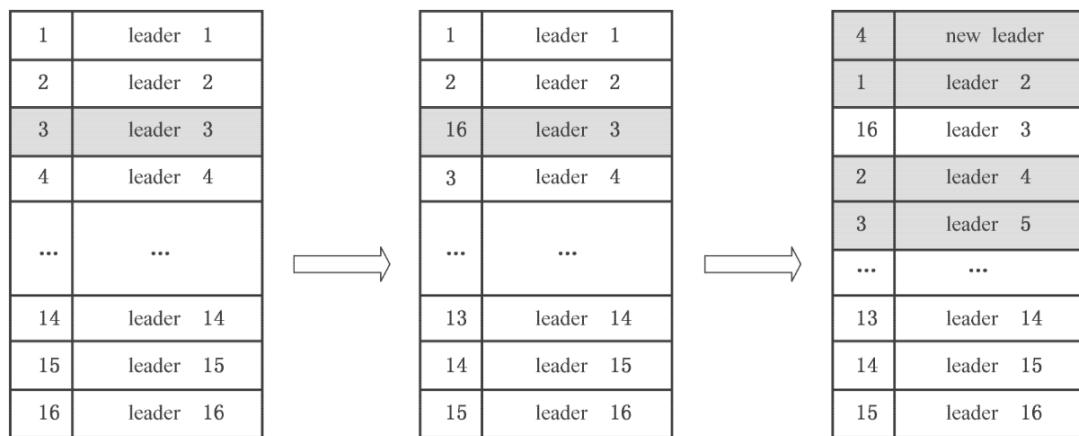


- a) 某状态下缓存数据存放情况 b) 特征码字 leader 3 被击中后数据更新情况 c) 在 b)状态下新的数据进来后未被击中时缓存数据替换情况

图 42 缓存中的数据替换过程

图42a)为某状态下缓存中的数据存储情况。图中16个缓存空间均已填满。第1列为缓存空间编号,第2列为所有的特征码字。此时,若下一个数据与缓存中的某个数据相同,即击中该特征码字。比如击中第3个特征码字leader3,则进行被击中数据沉入底部的操作,见图42b)。leader3沉入第16个缓存空间,而leader3之后的数据均往上浮1个位置,leader4上浮到了第3个空间,leader16上浮到了第15个空间。对于未击中的情况,则见图42c)。图42c)表示新进来1个缓存数据后未被击中时缓存数据的更新过程。新数据进入第4个缓存空间,原来第4个空间里的数据上浮到第3个空间,第3个空间里的数据上浮到第2个空间,第2个空间里的数据上浮到第1个空间,而原来第1个空间里的数据则被替换出来了,不再存放在缓存中。第5个空间到第16个空间里的数据则不作变动。

而在实际的编解码过程中,只是改变缓存空间的编号,并不变换数据在缓存中的存放位置。具体过程见图43。图43与图42表示的是同样的过程。



a) 某状态下缓存数据存放情况 b) 特征码字 leader 3 被击中后数据更新情况 c) 在 b) 状态下新的数据进来后未被击中时缓存数据替换情况

图 43 实际编码中缓存数据替换过程

图 43a) 为某状态下缓存里的数据存放情况, 图 43b) 表示击中特征码字 leader 3 后缓存中的数据处理过程, 图 43c) 表示未击中情况下新存入 1 个数据的过程。与图 43 不同的是, 这里并不改变数据在缓存中的存放位置, 而只改变空间的编号。具体过程为: 击中编号为 3 的特征码字时, 与它相应的缓存编号变为 16, 而原来编号从 4~16 的特征码字的编号都相应减 1; 未击中时, 将新特征码字存入原来编号为 1 的特征码字的位置, 而原来编号为 1 的特征码字则被替换, 不再存放在缓存中, 原来编号为 2、3、4 的特征码字编号都减 1, 其余特征码字对应的编号不变。

第 4 步: 缓存标识位的编码

为了表示特征码字是否被击中, 需要在码流中增加缓存标识位, 可通过修改基础码书的标识位 header 实现。修改后的 header 表示如下:

| | |
|---------------|-------------|
| header = 0 | 基础码书为 Q_0 |
| header = 10 | 基础码书为 Q_2 |
| header = 1100 | 基础码书为 Q_3 |
| header = 1101 | 数据为击中了的缓存数据 |
| header = 1110 | 基础码书为 Q_4 |

第 5 步: 编码输出

对缓存类型数据的编码输出, 如果击中了, 则输出其在缓存中的地址, 见图 43a)~b), 击中了 leader 3, 则输出为 leader 3 在缓存中的地址编号“3”; 若未击中, 则计算其码书索引作为输出。

6.2.4.8.5 使用分裂表的编码

对于待编码数据未在基础码书中(也包括奇化后仍不在基础码书中的 2D₈ 数据和非特殊特征码字), 使用分裂表进行编码。

一个码字由于其分量的绝对值太大, 不在基础码书中。对于这样的矢量编码方法是: 将绝对值大的分量减去一个分裂量, 得到一个绝对值足够小的差, 使这个差成为基础码书中的一个码字。将这个码字在基础码书中的索引和在分裂表中分裂量的索引编码输出。

分裂表的编码见图 38, 在分裂表扩展处理过程中, 包括两个扩展级编码方式: 一级扩展编码、二级扩展编码。下面分别对两个扩展级的编码方法进行具体描述:

a) 一级扩展编码：

一级扩展编码采用等比特数来编码各维分裂量。一级扩展分裂码表见表 159，分裂量取 0 或 4 两种数值。表中第三列为用二进制编码的分裂量索引，即取分裂量 0 时，编码输出为索引“0”；取分裂量 4 时，编码输出为索引“1”。这样，对于一个八维矢量中的八个分量，每一个分量都需要 1 比特来表示其分裂量的索引，共 8 比特。

一级扩展编码输出格式如图 44 所示，包括码头、偶标志位(even_flag)、基础码书的索引 i 和分裂量的索引 k 组成，其中码头如表 160 所示。

表 159 一级扩展编码的分裂码表

| 分裂量索引 | 分裂量 | 分裂量索引编码(二进制) | |
|-------|------|--------------|--------------|
| 0 | 0 | 0 | |
| 1 | 4 | 1 | |
| 码头 | 偶标志位 | 基础码书索引 i | 分裂量索引 k(8比特) |

图 44 一级扩展编码的输出编码格式

表 160 一级扩展编码的码头定义表

| 码头编码(二进制) | 码头编码(十六进制) | 基础码书 |
|-----------|------------|------------|
| 11110 | 0x1E | Q_4 |
| 1111110 | 0x7E | Q_3 |
| 111111110 | 0x3FE | inv_ Q_4 |

b) 二级扩展编码：

二级扩展编码采用不等长比特数来编码各维分裂量。二级扩展编码的分裂码表见表 161，列出了二级扩展使用的分裂级 k0 及 k2~k5 中的分裂量定义，k6、k7 依此类推。表中各个分裂量的大小为 4 的整数倍。

二级扩展使用的分裂表包括各个不同的级别，给各个级别的分裂表再定义一个码头信息，即分裂量码头(split header)，定义见表 162。

表 161 二级扩展编码的分裂码表

| 分裂量索引 | k0 中分裂量 | k0 中分裂量索引编码(二进制) | k2 中分裂量 | k2 中分裂量索引编码(二进制) | k3 中分裂量 | k3 中分裂量索引编码(二进制) | k4 中分裂量 | k4 中分裂量索引编码(二进制) | k5 中分裂量 | k5 中分裂量索引编码(二进制) |
|-------|---------|------------------|---------|------------------|---------|------------------|---------|------------------|---------|------------------|
| 0 | 0 | 无 | 4 | 0 | 12 | 00 | 28 | 000 | 60 | 0000 |
| 1 | | | 8 | 1 | 16 | 01 | 32 | 001 | 64 | 0001 |
| 2 | | | | | 20 | 10 | 36 | 010 | 68 | 0010 |
| 3 | | | | | 24 | 11 | 40 | 011 | 72 | 0011 |

表 161 (续)

| 分裂量索引 | k0 中分裂量 | k0 中分裂量索引编码(二进制) | k2 中分裂量 | k2 中分裂量索引编码(二进制) | k3 中分裂量 | k3 中分裂量索引编码(二进制) | k4 中分裂量 | k4 中分裂量索引编码(二进制) | k5 中分裂量 | k5 中分裂量索引编码(二进制) |
|-------|---------|------------------|---------|------------------|---------|------------------|---------|------------------|---------|------------------|
| 4 | | | | | | | 44 | 100 | 76 | 0100 |
| 5 | | | | | | | 48 | 101 | 80 | 0101 |
| 6 | | | | | | | 52 | 110 | 84 | 0110 |
| 7 | | | | | | | 56 | 111 | 88 | 0111 |
| 8 | | | | | | | | | 92 | 1000 |
| 9 | | | | | | | | | 96 | 1001 |
| 10 | | | | | | | | | 100 | 1010 |
| 11 | | | | | | | | | 104 | 1011 |
| 12 | | | | | | | | | 108 | 1100 |
| 13 | | | | | | | | | 112 | 1101 |
| 14 | | | | | | | | | 116 | 1110 |
| 15 | | | | | | | | | 120 | 1111 |

表 162 二级扩展编码的分裂量码头表

| 分裂量码头编码(二进制) | 分裂量码头编码(十六进制) | 分裂级 |
|--------------|---------------|-----|
| 0 | 0x0 | k0 |
| 10 | 0x2 | k2 |
| 110 | 0x6 | k3 |
| 1110 | 0xE | k4 |
| 11110 | 0x1E | k5 |
| 111110 | 0x3E | k6 |
| 1111110 | 0x7E | k7 |

若取分裂量为 0 时,即该维数据未进行分裂,则输出一个比特“0”作为分裂量码头来标识,且无需输出分裂量在 k0 中的索引;若取分裂量为 4 时,该分裂量在 k2 中,编码输出其分裂量码头索引“10”,以及其在 k2 中的索引“0”;若取分裂量为 20 时,该分裂量在 k3 中,编码输出其分裂量码头索引“110”,以及其在 k3 中的索引“10”。

二级扩展编码输出格式见图 45,包括码头、偶标志位(even_flag)、基础码书的索引、8 个分量的分裂量码头,以及对应的 8 个分裂量索引。二级扩展编码的码头定义见表 163。

如果某一个分裂量码头为“0”时,其对应的索引不需要编码,占用 0 比特。分裂量索引的长度依据分裂量码头而定,分裂量在 k0,k2~k7 中时,所消耗的比特数见表 164。

| 码头 | 偶标 志位 | 基础 码书 索引i | 第一个分 量的分裂 量码头 | 第一个分 量的分裂 量索引 | 第二个分 量的分裂 量码头 | 第二个分 量的分裂 量索引 | | 第八个分 量的分裂 量码头 | 第八个分 量的分裂 量索引 |
|----|----------|-----------------|---------------------|---------------------|---------------------|---------------------|-------|---------------------|---------------------|
|----|----------|-----------------|---------------------|---------------------|---------------------|---------------------|-------|---------------------|---------------------|

图 45 二级扩展编码的输出编码格式

表 163 二级扩展编码的码头定义表

| 码头编码(二进制) | 码头编码(十六进制) | 基础码书 |
|------------|------------|--------------------|
| 111110 | 0x3E | Q ₄ |
| 11111110 | 0xFE | Q ₃ |
| 1111111111 | 0xFF | inv_Q ₄ |

表 164 分裂量所消耗比特数表

| 分裂级 | 分裂量码头消耗比特数 | 分裂量索引消耗比特数 | 单个分裂量消耗的总比特数 |
|-----|------------|------------|--------------|
| k0 | 1 | 0 | 1 |
| k2 | 2 | 1 | 3 |
| k3 | 3 | 2 | 5 |
| k4 | 4 | 3 | 7 |
| k5 | 5 | 4 | 9 |
| k6 | 6 | 5 | 11 |
| k7 | 7 | 6 | 13 |

6.2.4.9 增益平衡

由于对不同频带样值采用了不同的缩放因子,重建信号时需要消除缩放因子的影响,即增益平衡。假设量化后输出的频谱样值为 $X_q(1, 2, \dots, N)$, 则增益平衡后输出值见式(86):

$$X_{\text{balance}} = \left[X_q(0, 1, \dots, N/2), \frac{g_2}{g_1} X_q(N/2 + 1, N/2 + 2, \dots, N) \right] \quad \dots\dots\dots (86)$$

6.2.4.10 峰值逆整形

峰值逆整形的原理见峰值预整形部分,算法流程见图 46。



图 46 峰值逆整形示意图

处理步骤如下:

- 标记频谱中的峰值集合 $\{p_i\}$ 。 p_i 定义为整形频谱段幅值的局部最大值;
- 计算参考值 ref_{\max} 。这里的参考值应该与峰值预整形中一致;
- 计算峰值 p_i 的缩小因子 $R_i = ref_{\max}/p_i$;
- 在峰值集合 $\{p_i\}$ 中除去 ref_{\max} 相关的峰值点(保证 ref_{\max} 的不变性)。对剩余的峰值点 p_i 进

行缩小 $p_i = p_i / R_i$ 。

6.2.4.11 逆时频变换

量化频谱 $\hat{X}(k)$ 进行逆变换得到时域量化信号 $\hat{x}(n)$, IDFT 变换定义见式(87):

$$\hat{x}(n) = \frac{1}{L_{\text{DFT}}} \sum_{k=0}^{L_{\text{DFT}}-1} \hat{X}(k) e^{\frac{2jnk\pi}{L_{\text{DFT}}}} \quad \dots \dots \dots \quad (87)$$

式中:

L_{DFT} ——IDFT 变换的长度, 等于 288 个样本点。具体实现时可使用 IFFT。

6.2.4.12 计算和量化全局增益

最佳全局增益使得量化加权信号与原始加权信号之间的均方误差最小。假设原始加权信号为 x , 量化加权信号为 \hat{x} , 则最佳全局增益见式(88):

$$g^* = \frac{\sum_{n=0}^{L_{\text{DFT}}-1} x(n) \hat{x}(n)}{\sum_{n=0}^{L_{\text{DFT}}-1} \hat{x}(n) \hat{x}(n)} \quad \dots \dots \dots \quad (88)$$

增益 g^* 使用对数量化为 7 比特索引值, 具体过程如下:

- 计算量化加权信号能量: $E = \sum_{n=0}^{L_{\text{DFT}}-1} \hat{x}^2(n)$;
- 计算 RMS 值: $rms = \sqrt{\frac{E}{L_{\text{DFT}}}}$;
- 计算归一化的增益: $G = g^* \times rms$;
- 计算索引: $index = floor(28 \log_{10} G + 0.5)$;
- $if(index < 0) index = 0; if(index > 127) index = 127$ 。

在编码器和解码器端, 量化的全局增益 \hat{g}^* 计算见式(89):

$$\hat{g}^* = 10^{\frac{index}{28}} / rms \quad \dots \dots \dots \quad (89)$$

6.2.5 高频信号编码(BWE)

高频信号编码框图见图 47 所示, 图中 I 表示每个音频超帧所包含子帧的个数, 当音频超帧长度为 512 时, $I=4$ 。高频信号是指输入信号中频率大于 $F_s/4$ 的信号分量, 高频信号的带宽取决于输入信号的采样频率。

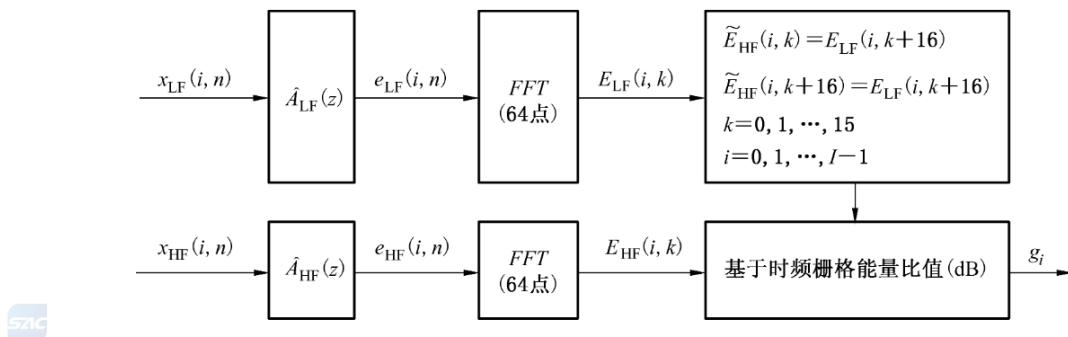


图 47 高频编码框图

编码步骤如下:

a) 低频和高频分析滤波器

低频信号的分析滤波器 $\hat{A}_{LF}(z)$ 直接从低频编码器获得。

高频信号的分析滤波器 $\hat{A}_{HF}(z)$: 对高频信号 $x_{HF}(i, n)$ 的每帧(256个样本)求取一组八阶的LP系数, 将LP系数转换为ISP系数, ISP系数又进一步变换为ISF系数并用9比特量化。每个子帧(64个样本)按照6.2.3.2.7插值公式(37)对ISP系数进行内插, 这样就得到高频信号的分析滤波器 $\hat{A}_{HF}(z)$ 。

b) 残差计算

低频信号 $x_{LF}(i, n)$ 通过量化的分析滤波器 $\hat{A}_{LF}(z)$ 得到低频残差 $e_{LF}(i, n)$;

高频信号 $x_{HF}(i, n)$ 通过量化的分析滤波器 $\hat{A}_{HF}(z)$ 得到高频残差 $e_{HF}(i, n)$;

c) FFT 变换

低频残差 $e_{LF}(i, n)$ 经过FFT变换得到低频残差谱 $E_{LF}(i, k)$, FFT变换长度采用64点。高频残差 $e_{HF}(i, n)$ 也经过FFT变换得到高频残差谱 $E_{HF}(i, k)$, FFT变换长度也采用64点。

d) 频谱拷贝

低频信号在时间上以64点为一个子帧, 在频率上将低频残差谱 $E_{LF}(i, k)$ 等分为低低频残差谱 $E_{LFL}(i, k)$ 和低高频残差谱 $E_{LFH}(i, k)$ 。因为低低频残差谱 $E_{LFL}(i, k)$ 存在很强的弦性, 所以在高频拷贝中不用。而是将低高频残差谱 $E_{LFH}(i, k)$ 复制两次来得到估计的高频残差谱 $\tilde{E}_{HF}(i, k)$ 。

e) 高频信号的时频栅格划分

当低频编码器使用ACELP时, 高频信号的时频划分总是采用最高时间分辨率, 即64点; 相应的频率分辨率最小, 其时频划分见图48。

当低频编码器使用TAC时, 高频的时频栅格划分依赖于超帧长度。当超帧长度为512时, 只能使用图48的时频栅格划分结构。

f) 增益计算

由低频残差信号代替高频残差信号, 然后计算重建高频残差信号和原始高频残差信号之间增益因子: 先分别计算两个信号所对应栅格覆盖的时频数据的能量, 最后计算能量比值, 并转换以dB为单位。这样每个超帧就得到I个增益($g_0, g_1, g_2, \dots, g_{I-1}$)。

g) 增益调整

对每个时频划分块, 判断原始信号和拷贝信号的弦性, 如果原始信号有弦而拷贝信号无弦, 则对相应块的增益进行适当的减小, 以防止噪声水平过分抬高。

h) 增益矢量量化编码

最后每个超帧的连续4个增益系数组成一个增益矢量, 并用7比特进行量化。

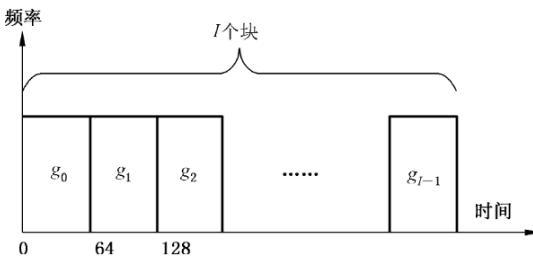


图 48 时频栅格划分图

每个超帧BWE参数的总比特消耗为: $16 \times (I/4)$, 其中包括 $I/4$ 组高频LP系数(每组9比特), 以

及 $I/4$ 个增益矢量(每个矢量 7 比特)。

6.2.6 识别特征参数编码

6.2.6.1 识别特征参数提取

6.2.6.1.1 去直流偏置

信号经过高通滤波器，目的是为了滤掉信号中的直流分量。高通滤波器的传递函数如式(90)：

$$H_{h1}(z) = \frac{b_0 - b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} + a_2 z^{-2}} \quad \dots \dots \dots \quad (90)$$

式中：

$$a_1 = -1.988\ 89 \cdot$$

$$a_2 = 0.988\ 95 \pm$$

$$b_0 = 0.99446;$$

$$b_1 = -1.988\ 9$$

$$b_0 = 0.99446$$

6.2.6.1.2 喘声消除

噪声消除模块主要作用是降低背景噪声,提高信号的信噪比。无论语音识别还是声纹识别算法,噪声对识别结果影响很大。因此在识别特征参数提取之前,应先对信号进行降噪处理。噪声消除算法描述参见附录J。

6.2.6.1.3 波形预处理

6.2.6.1.3.1 基音周期搜索

波形预处理应用于降噪后的信号。降噪模块的输出(每帧 160 个样本)存储在包含 480 个样本的缓存中, 波形预处理模块处理的窗大小 N_{in} 为缓存中前 400 个样本。图 49 描述了波形预处理模块的原理。

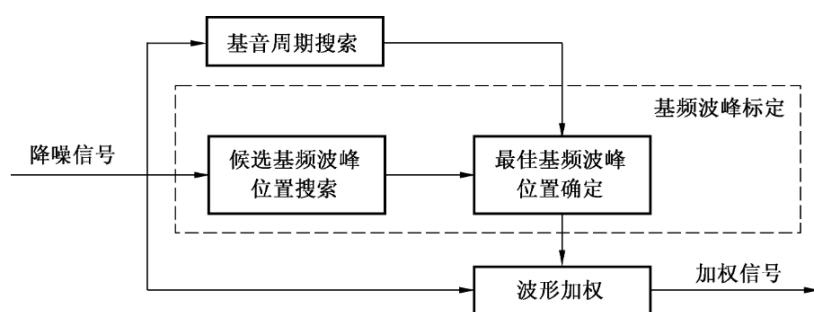


图 49 波形预处理原理框图

利用信号的归一化自相关对降噪后的信号估计基音周期,归一化自相关计算见式(91):

$$corr = \frac{\sum_{n=0}^{239} s_{nr}(n)s_{nr}(n - delay)}{\sqrt{\sum_{n=0}^{239} s_{nr}^2(n) \sum_{n=0}^{239} s_{nr}^2(n - delay)}}, \quad 40 \leqslant delay \leqslant 160 \quad(91)$$

详细基音周期搜索算法描述见 6.2.3.4.1。

6.2.6.1.3.2 候选基频波峰位置搜索

在波形预处理窗内搜索绝对值最大的 M 个样本点所对应的位置作为候选基频波峰位置, 记为: $Pc[i], i=1, 2, \dots, M$; 按该位置所对应的信号样本绝对值从大到小顺序排列, 见式(92):

$$Pc[1], \dots, Pc[M] = \underset{n=0, \dots, L-1}{\operatorname{argmax}} (\left| s_{nr}(n) \right|) \quad \text{最大的 } M \text{ 个值} \quad (92)$$

式中:

$M=3$ 。

6.2.6.1.3.3 最佳基频波峰位置确定

该模块用来通过基音周期和候选基频波峰位置来确定最佳基频波峰位置, 根据基音周期 T_p 在本帧范围内延拓出的所有对应位置上样本的平均幅度值, 计算每个候选基频波峰位置, 见式(93):

$$Mp_avg[i] = \frac{\left| \sum_j (Pc[i] + j \times T_p) \right|}{D}, \quad i=1, \dots, M \quad (93)$$

式中:

D ——满足 $0 \leq Pc[i] + j \times T_p \leq N_{in}-1$ 的 j 的个数;

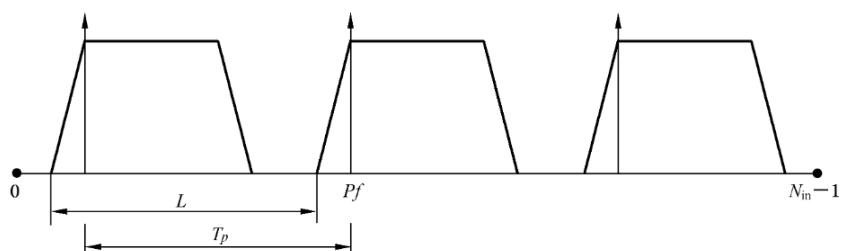
$Mp_avg[i]$ ——根据第 i 个候选基频波峰位置所计算的平均幅度值, M 为候选基频波峰位置个数。

假设最佳的基频波峰位置为 Pf , 见式(94):

$$Pf = \underset{i=1, \dots, M}{\operatorname{argmax}} (Mp_avg[i]) \quad (94)$$

6.2.6.1.3.4 波形加权

波形加权就是将基频波峰位置附近的信号提升, 其他位置的信号减小, 从而起到提高信噪比同时增强基频的作用。方法是在该帧信号内的每个基音周期内用窗函数加权。波峰附近的权值大, 其他位置的权值小, 同时窗函数要尽量保证处理后的信号连续。如图 50 所示。



说明:

L ——窗函数的长度;

T_p ——基音周期;

Pf ——最佳的基频波峰位置。

图 50 波形加权处理框图

窗函数的定义见式(95):

$$w_{\text{swp}}(n) = \begin{cases} 0.8, & n = 0 \\ 1.0, & n = 1 \\ 1.2, & 2 \leq n \leq 0.6T_p \\ 1.0, & n = 0.6T_p + 1 \\ 0.8, & 0.6T_p + 2 \leq n \leq T_p - 1 \end{cases} \dots \quad (95)$$

按照每个基音周期进行加权,见式(96):

$$\begin{aligned} s_{\text{swp}}(n) &= w_{\text{swp}}(i) \times s_{\text{nr}}(n), & n = Pf_n + i - 2, & 0 \leq i < T_p \\ Pf_n &+= T_p \end{aligned} \dots \quad (96)$$

第一个基音周期和最后一个基音周期边界处理,保证 $0 \leq n < N_{\text{in}}$ 。 Pf_n 表示按基音周期扩展的基频波峰位置,其初值通过式(97)获得:

$$\begin{aligned} Pf_n &= Pf \\ \text{While}(Pf_n > 0) &\quad Pf_n -= T_p \end{aligned} \dots \quad (97)$$

6.2.6.1.4 倒谱计算

6.2.6.1.4.1 对数能量计算

对波形加权后的信号 $s_{\text{swp}}(n)$ 的每帧能量参数取对数,见式(98)和式(99):

$$\ln E = \begin{cases} \ln(E_{\text{swp}}), & E_{\text{swp}} \geq E_{\text{THRESH}} \\ \ln(E_{\text{THRESH}}), & E_{\text{swp}} < E_{\text{THRESH}} \end{cases} \dots \quad (98)$$

$$E_{\text{THRESH}} = \exp(-1), E_{\text{swp}} = \sum_{n=0}^{N_{\text{in}}-1} S_{\text{swp}}(n) \times S_{\text{swp}}(n) \dots \quad (99)$$

6.2.6.1.4.2 预加重

预加重滤波器处理后的信号 $s_{\text{swp_pe}}(n)$ 见式(100):

$$s_{\text{swp_pe}}(n) = s_{\text{swp}}(n) - 0.9 \times s_{\text{swp}}(n - 1) \dots \quad (100)$$

式中:

$s_{\text{swp_pe}}(-1)$ ——上一帧的最后一个样本,如果是第一帧,则其值为 0。

6.2.6.1.4.3 加窗



对预加重处理模块输出信号进行加窗处理,窗类型为长度 $N_{\text{in}} = 400$ 的海明窗,见式(101):

$$s_{\text{swp_w}}(n) = \left[0.54 - 0.46 \times \cos\left(\frac{2\pi \times (n + 0.5)}{N_{\text{in}}}\right) \right] \times s_{\text{swp_pe}}(n), \quad 0 \leq n \leq N_{\text{in}} - 1 \dots \quad (101)$$

6.2.6.1.4.4 FFT 变换和功率谱估计

通过后面补零将 N_{in} 个样本扩展为 512 个样本。用长度 $N_{\text{FFT}} = 512$ 的 FFT 计算出信号频谱 $X_{\text{swp}}(\text{bin})$,见式(102):

$$X_{\text{swp}}(\text{bin}) = \text{FFT}\{s_{\text{swp_w}}(n)\}, \quad 0 \leq \text{bin} \leq N_{\text{FFT}}/2 \dots \quad (102)$$

相应的功率谱 $P_{\text{swp}}(\text{bin})$ 见式(103):

$$P_{\text{swp}}(\text{bin}) = |X_{\text{swp}}(\text{bin})|^2, \quad 0 \leq \text{bin} \leq N_{\text{FFT}}/2 \dots \quad (103)$$

6.2.6.1.4.5 Mel 滤波

Mel 滤波模块将线性频率频谱表示为 Mel 刻度频谱。信号有效频带位于 f_{start} 与 $f_{\text{samp}}/2$ 之间,在 Mel 域分为 K_{FB} 个子带,每个子带对应一个三角形频率窗,相邻子带有 50% 重叠。

定义 FFT 索引 $bin = N_{\text{FFT}}$ 对应的频率是 f_{samp} , 则线性频率转换为 FFT 索引公式见式(104):

$$\text{index}\{f\} = \text{round}\left\{\frac{f}{f_{\text{samp}}} \times N_{\text{FFT}}\right\} \quad \dots \dots \dots \quad (104)$$

Mel 函数计算公式见式(105):

$$\text{Mel}\{x\} = A \times \log_{10}\left(1 + \frac{x}{\mu}\right) = \lambda \times \ln\left(1 + \frac{x}{\mu}\right), \quad \lambda = \frac{A}{\ln(10)} \quad \dots \dots \dots \quad (105)$$

Mel 反函数计算公式见式(106):

$$\text{Mel}^{-1}\{y\} = \mu \times \left(\exp\left(\frac{y}{\lambda}\right) - 1\right) \quad \dots \dots \dots \quad (106)$$

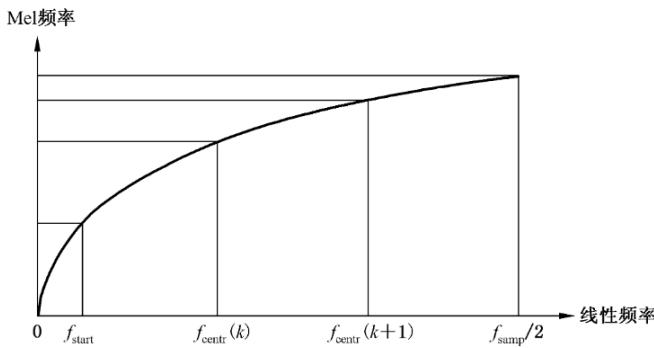
为了将 Mel 域的频带等间隔划分, 通过计算 Mel 函数得到滤波器的中心频率。

图 51 线性频率和 Mel 频率映射

中心频率的计算见式(107):

$$f_{\text{centr}}(k) = \text{Mel}^{-1}\left\{\text{Mel}\{f_{\text{start}}\} + k \times \frac{\text{Mel}\{f_{\text{samp}/2}\} - \text{Mel}\{f_{\text{start}}\}}{K_{\text{FB}} + 1}\right\}, \quad 1 \leqslant k \leqslant K_{\text{FB}} \quad \dots \dots \dots \quad (107)$$

式中:

$$f_{\text{start}} = 64 \text{ Hz};$$

$$f_{\text{samp}} = 16\,000 \text{ Hz};$$

$$\mu = 700;$$

$$A = 2\,595;$$

$$\lambda = 1\,127;$$

$$K_{\text{FB}} = 32.$$

将 Mel 滤波器的中心频率表示为 FFT 索引, 见式(108):

$$bin_{\text{centr}}(k) = \text{index}\{f_{\text{centr}}(k)\} = \text{round}\left\{\frac{f_{\text{centr}}(k)}{f_{\text{samp}}} \times N_{\text{FFT}}\right\}, \quad 1 \leqslant k \leqslant K_{\text{FB}} \quad \dots \dots \dots \quad (108)$$

对第 k 个 Mel 子带, 频率窗可分两个部分。前一部分[频率为 $f_{\text{centr}}(k-1) < f < f_{\text{centr}}(k)$]宜增加权重, 后一部分[频率为 $f_{\text{centr}}(k) < f < f_{\text{centr}}(k+1)$]宜降低权重, 对功率谱 $P_{\text{swp}}(bin)$ 加上这些频率窗。根据每个子带中频谱线相对于中心频率的位置, 计算每个子带的频率窗权重, 见式(109):

$$W_{\text{left}}(i, k) = \frac{i - bin_{\text{centr}}(k-1) + 1}{bin_{\text{centr}}(k) - bin_{\text{centr}}(k-1) + 1}, \quad 1 \leqslant k \leqslant K_{\text{FB}}, bin_{\text{centr}}(k-1) \leqslant i \leqslant bin_{\text{centr}}(k)$$

$$W_{\text{right}}(i, k) = 1 - \frac{i - bin_{\text{centr}}(k)}{bin_{\text{centr}}(k+1) - bin_{\text{centr}}(k) + 1}, \quad 1 \leqslant k \leqslant K_{\text{FB}}, bin_{\text{centr}}(k) < i \leqslant bin_{\text{centr}}(k+1)$$

$$\dots \dots \dots \quad (109)$$

其他情况下,权重值为 0。

Mel 滤波器的输出为每个子带的功率谱值 $P_{\text{swp}}(bin)$ 的加权和,见式(110):

$$E_{\text{FB}}(k) = \sum_{i=\text{bin centr}(k-1)}^{\text{bin centr}(k)} W_{\text{left}}(i, k) \times P_{\text{swp}}(i) + \sum_{i=\text{bin centr}(k)+1}^{\text{bin centr}(k+1)} W_{\text{right}}(i, k) \times P_{\text{swp}}(i), \quad 1 \leq k \leq K_{\text{FB}} \dots (110)$$

6.2.6.1.4.6 高频聚合

6.2.6.1.4.5 求出了 32 个子带的 Mel 滤波后的输出,由于高频信号容易受到噪声的干扰,较多的子带划分影响了参数的鲁棒性,因此将高频的 9 个子带聚合成 3 个子带,聚合方法采用加权平均的方法,见式(111)~式(113):

$$E_{\text{FB}}(24) = \frac{\alpha E_{\text{FB}}(24) + \beta E_{\text{FB}}(25) + \gamma E_{\text{FB}}(26)}{\alpha + \beta + \gamma} \dots (111)$$

$$E_{\text{FB}}(25) = \frac{\alpha E_{\text{FB}}(27) + \beta E_{\text{FB}}(28) + \gamma E_{\text{FB}}(29)}{\alpha + \beta + \gamma} \dots (112)$$

$$E_{\text{FB}}(26) = \frac{\alpha E_{\text{FB}}(30) + \beta E_{\text{FB}}(31) + \gamma E_{\text{FB}}(32)}{\alpha + \beta + \gamma} \dots (113)$$

式中:

$$\alpha = \beta = \gamma = 1.$$

6.2.6.1.4.7 非线性变换(计算对数)

对 Mel 滤波器的输出取对数,见式(114):

$$S_{\text{FB}}(k) = \ln(E_{\text{FB}}(k)), \quad 1 \leq k \leq K_{\text{FB}} \dots (114)$$

式中:

$$K_{\text{FB}} = 26.$$

限制对数滤波器组的输出不能小于 -1。

6.2.6.1.4.8 DCT 变换

对非线性变换模块的输出作 DCT 得到 13 个倒谱系数,见式(115):

$$c(i) = \sum_{k=1}^{K_{\text{FB}}} S_{\text{FB}}(k) \times \cos\left(\frac{i \times \pi}{K_{\text{FB}}} \times (k - 0.5)\right), \quad 0 \leq i \leq 12 \dots (115)$$

式中:

$$K_{\text{FB}} = 26.$$

6.2.6.1.4.9 倒谱计算输出

倒谱计算得到的特征矢量由 14 个系数组成:1 个对数能量系数 $\ln E$ 和 13 个倒谱系数 $c(0)$ 到 $c(12)$ 。对数能量系数和倒谱系数 $c(0)$ 都表示信号短时能量。

6.2.6.1.4.10 去信道干扰

对于卷积特性的信道干扰,在倒谱域为相加。因此将信号倒谱系数减去信道倒谱系数,即可得到去除信道干扰后均衡的信号倒谱系数,见式(116):

$$c_{\text{eq}}(i) = c(i) - TranCep(i), \quad i = 1, \dots, 12 \dots (116)$$

式中:

$c_{\text{eq}}(i)$ —— 均衡的信号倒谱系数;

$TranCep(i)$ —— 信道倒谱系数,初值为 0。

采用低通滤波的方法来估计信道倒谱分量 $TranCep(i)$, 可采用一阶 IIR 滤波, 见式(117) :

$$TranCep(i) = TranCep(i)(1 - \alpha) + (c(i) - RefCep(i))\beta_1 + c(i)\beta_2 \quad \dots \dots \dots (117)$$

式中:

$RefCep(i)$ —— 均衡信道下的语音信号倒谱特征矢量的统计均值。如下:

$$RefCep(1) = -6.618909, RefCep(2) = 0.198269, RefCep(3) = -0.740308,$$

$$RefCep(4) = 0.055132, RefCep(5) = -0.227086, RefCep(6) = 0.144280,$$

$$RefCep(7) = -0.112451, RefCep(8) = -0.146940, RefCep(9) = -0.327466,$$

$$RefCep(10) = 0.134571, RefCep(11) = 0.027884, RefCep(12) = -0.114905,$$

α —— IIR 低通滤波器的参数, 且满足当信噪比 $SNR \geq 0$ dB 时, $\alpha = 0.01$; 否则, $\alpha = 0.005$ 。

β_1, β_2 —— $\beta_1 + \beta_2 = \alpha$, 且满足: 当 SNR 高时, $\beta_1 \gg \beta_2$; 当 SNR 低时, $\beta_1 \ll \beta_2$, 见表 165 设置。

表 165 低通滤波器系数定义表

| SNR (dB) | $SNR > 20$ | $15 \leq SNR < 20$ | $10 \leq SNR < 15$ | $5 \leq SNR < 10$ | $0 \leq SNR < 5$ | $-10 \leq SNR < 0$ | $SNR < -10$ |
|-----------|----------------|--------------------|--------------------|-------------------|------------------|--------------------|----------------|
| β_1 | $100\% \alpha$ | $90\% \alpha$ | $80\% \alpha$ | $70\% \alpha$ | $50\% \alpha$ | $20\% \alpha$ | 0 |
| β_2 | 0 | $10\% \alpha$ | $20\% \alpha$ | $30\% \alpha$ | $50\% \alpha$ | $80\% \alpha$ | $100\% \alpha$ |

SNR 由噪声消除模块提供, 如果得不到 SNR , 则将参数设置为:

$$\alpha = 0.01; \beta_1 = \alpha; \beta_2 = 0.$$

6.2.6.2 识别特征参数压缩

6.2.6.2.1 特征矢量

特征矢量就是 6.2.6.1 中从每个短时分析帧中提取的参数, 包括 12 个 MFCC 系数, 见式(118) :

$$C_{eq}(t) = [c_{eq}(1, t), c_{eq}(2, t), \dots, c_{eq}(12, t)]^T \quad \dots \dots \dots (118)$$

式中:

t —— 帧索引。

特征矢量还包括 MFCC 系数 $c(0, t)$, 对数能量系数 $\ln E(t)$ 和 VAD 标志位。

最终编码的特征矢量表示见式(119) :

$$y(t) = \begin{bmatrix} C_{eq}(t) \\ VAD(t) \\ c(0, t) \\ \ln E(t) \end{bmatrix} \quad \dots \dots \dots (119)$$

6.2.6.2.2 矢量量化

特征矢量 $y(t)$ 使用分裂的矢量量化。14 个系数 [$c(1) \sim c(12)$, $c(0)$ 和 $\ln E$] 两个 1 组, 被分成 7 组, 每组都用独立的 VQ 码书进行量化, VAD 标志位作为 1 个独立比特进行传输。矢量量化所使用的量化失真度量是加权欧氏距离, 见式(120)和式(121):

$$d_j^{i,i+1} = \begin{bmatrix} y_i(t) \\ y_{i+1}(t) \end{bmatrix} - q_j^{i,i+1} \quad \dots \dots \dots (120)$$

$$idx^{i,i+1}(t) = \underset{0 \leq j \leq (N^{i,i+1} - 1)}{\operatorname{argmin}} \{(d_j^{i,i+1})^T W^{i,i+1} (d_j^{i,i+1})\}, \quad i = \{0, 2, 4, \dots, 12\}$$

$$\dots \dots \dots (121)$$

式中：

$q_j^{i,i+1}$ —— 码书 $Q^{i,i+1}$ 的第 j 个码字;

$(N^{i,i+1} - 1)$ —— 码书大小；

$W^{i,i+1}$ ——码书 $Q^{i,i+1}$ 的加权矩阵;

$idx^{i,i+1}(t)$ ——量化 $[y_i(t), y_{i+1}(t)]^T$ 所得到的码书索引。

加权矩阵 $W^{12,13}$ 见(122):

其他加权矩阵为单位阵(即对角线元素为 1, 其他为 0)。

识别特征编码提供两种编码模式,因此矢量量化的码表也需要两组:一组是直接量化特征矢量(见表 166),另一组是量化残差矢量(见表 167)。

表 166 直接编码模式下量化码表

| 码书 | 码书大小 | 量化子矢量 |
|-------------|------|------------------|
| $Q^{0,1}$ | 64 | $[c(1), c(2)]$ |
| $Q^{2,3}$ | 64 | $[c(3), c(4)]$ |
| $Q^{4,5}$ | 64 | $[c(5), c(6)]$ |
| $Q^{6,7}$ | 64 | $[c(7), c(8)]$ |
| $Q^{8,9}$ | 64 | $[c(9), c(10)]$ |
| $Q^{10,11}$ | 32 | $[c(11), c(12)]$ |
| $Q^{12,13}$ | 256 | $[c(0), \ln E]$ |

表 167 预测编码模式下量化码表

| 码书 | 码书大小 | 量化子矢量 |
|-------------|------|------------------|
| $Q^{0,1}$ | 16 | $[c(1), c(2)]$ |
| $Q^{2,3}$ | 16 | $[c(3), c(4)]$ |
| $Q^{4,5}$ | 16 | $[c(5), c(6)]$ |
| $Q^{6,7}$ | 16 | $[c(7), c(8)]$ |
| $Q^{8,9}$ | 16 | $[c(9), c(10)]$ |
| $Q^{10,11}$ | 8 | $[c(11), c(12)]$ |
| $Q^{12,13}$ | 64 | $[c(0), \ln E]$ |

6.2.7 打包格式

6.2.7.1 音频编码参数打包

512 个样本点音频帧的编码参数被写入了 1 个二进制包,包的具体格式同编码模式相关,如图 52 所示。打包的编码参数包括:4 比特的模式信息,5 比特填充位(填充“11111”),1 比特的感知加权标志,ACELP 参数或 TVC 参数,以及高频编码参数。对于 ACELP 编码模式,模式位为 0;对于 TVC 编码模式,模式位为 1;其他模式位保留,以备将来扩展。

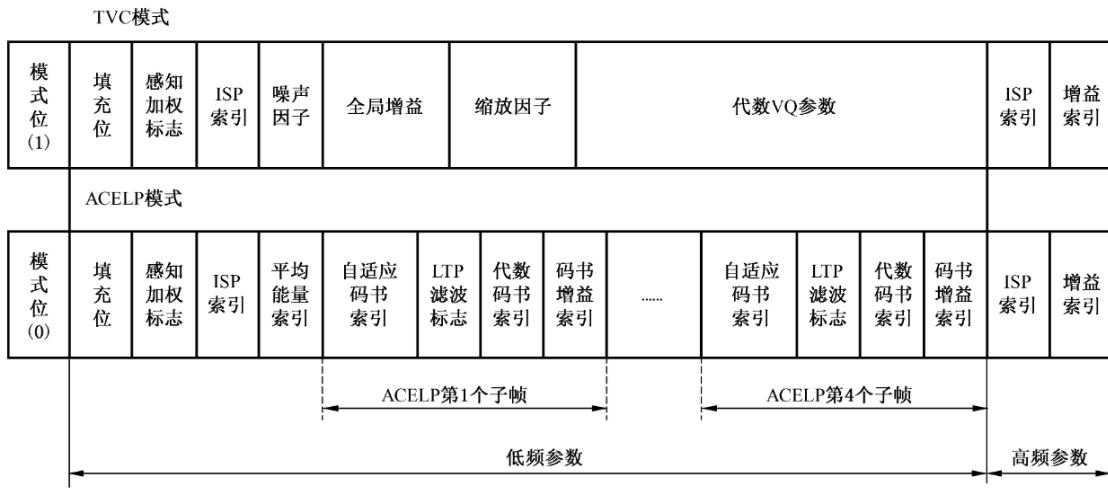


图 52 音频帧编码参数的数据格式

6.2.7.2 识别特征参数打包

在 6.2.6.2.2 对每帧得到的特征矢量进行矢量化,只需传输码书索引,1 比特的 VAD 标志和 1 个 CRC 校验和。对于直接编码模式,使用 4 比特的 CRC 校验和(CRC 生成多项式为 $g(X)=1+X+X^4$);对于预测编码模式,使用 2 比特的 CRC 校验和[CRC 生成多项式为 $g(X)=1+X+X^2$]。每帧识别特征参数压缩后打包格式见图 53 和图 54 所示。

| 比特 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 字节 |
|----|------------------|----------------|------------------|---|------------------|------------------|----------------|---|----|
| | $idx^{2,3}(t)$ | | | | $idx^{0,1}(t)$ | | | | 1 |
| | | $idx^{4,5}(t)$ | | | | $idx^{2,3}(t)$ | | | 2 |
| | | | $idx^{6,7}(t)$ | | | | $idx^{4,5}(t)$ | | 3 |
| | $idx^{10,11}(t)$ | VAD(t) | | | $idx^{8,9}(t)$ | | | | 4 |
| | | | $idx^{12,13}(t)$ | | | $idx^{10,11}(t)$ | | | 5 |
| | CRC | | | | $idx^{12,13}(t)$ | | | | 6 |

图 53 直接编码模式帧的数据格式(48 比特)

| 比特 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 字节 |
|----|-----|----------------|------------------|--------|------------------|----------------|---|---|----|
| | | $idx^{2,3}(t)$ | | | $idx^{0,1}(t)$ | | | | 1 |
| | | | $idx^{6,7}(t)$ | | | $idx^{4,5}(t)$ | | | 2 |
| | | | $idx^{10,11}(t)$ | VAD(t) | | $idx^{8,9}(t)$ | | | 3 |
| | CRC | | | | $idx^{12,13}(t)$ | | | | 4 |

图 54 预测编码模式帧的数据格式(32 比特)

6.3 解码器功能描述

6.3.1 低频信号解码

6.3.1.1 ACELP 解码

6.3.1.1.1 ACELP 解码过程

解码器解码接收到的参数并合成得到重建信号,见图 55 所示。

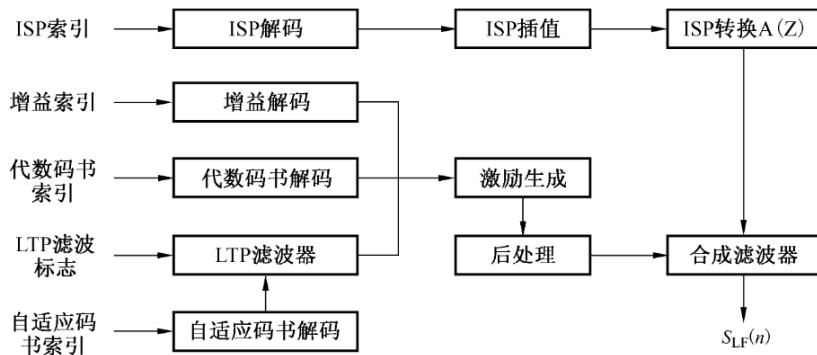


图 55 ACELP 解码合成框图

6.3.1.1.2 解码 LP 滤波器参数

接收到的 ISP 索引用来重构已量化的 ISP 系数,然后按照 6.2.3.2.7 中 ISP 系数的插值方法计算每个子帧的 ISP 系数。对于每个子帧,插值后 ISP 系数被转换为 LP 系数,用于 LP 合成滤波器来重建信号。

6.3.1.1.3 解码自适应码书矢量

接收到的自适应码书索引(基音延迟索引)用来搜索基音延迟的整数部分和小数部分。自适应码书矢量 $v(n)$ 使用过去的激励 $u(n)$ 通过 FIR 滤波器来生成。接收到的 LTP 滤波标志被用来确定解码的自适应码书矢量是 $v_1(n)=v(n)$ 还是 $v_2(n)=0.26v(n)+0.48v(n-1)+0.26v(n-2)$ 。

6.3.1.1.4 解码代数码书矢量

接收到的代数码书索引用来求取激励脉冲的位置、幅度以及代数码书矢量 $c(n)$ 。如果基音延迟的整数部分小于子帧长度 64,则通过自适应滤波器 $F(z)$ 对 $c(n)$ 进行基音周期锐化滤波。 $F(z)$ 定义见 6.2.3.4.6 中预滤波器。

6.3.1.1.5 解码自适应和代数码书增益

接收到的增益索引提供了自适应码书增益 \hat{g}_p ,代数码书增益校正因子 $\hat{\gamma}$ 和代数码书的平均能量 \bar{E}_s 。代数码书增益的重建过程为:

- 按 6.2.3.4.9 中式(72)求得代数码书激励矢量的能量 E_i ;
- 按 6.2.3.4.9 中式(73)代数码书的预测增益 g'_c ;
- 得到量化的代数码书增益 $\hat{g}_c = \hat{\gamma} g'_c$ 。

6.3.1.1.6 代数码书增益平滑

代数码书增益平滑先计算当前帧的 ISF 参数变化因子，并对当前帧的代数码书增益进行初始化修正；然后按照 ISF 参数变化因子确定当前帧的状态；最后利用初始化修正后的代数码书增益以及当前帧的平滑因子，对当前帧的代数码书增益进行平滑。具体步骤如下：

- a) 计算 ISF 变化因子 isf_diff , 见式(123):

$$isf_diff = \frac{\sum_{i=0}^{14} (isf_new_i - isf_old_i)^2}{400\ 000} \dots \dots \dots \quad (123)$$

式中：

isf_new ——当前帧的 ISF;

isf_old ——上一帧的 ISF。

- b) 对代码书增益 \hat{g}_c 进行初始化修正, 见式(124):

式中：

g_0 ——当前帧初始化修正后的代数码书增益；

g_{-1} ——前一帧初始化修正后的代数码书增益。

- c) 根据 isf_diff 将当前帧分为两类状态, 根据不同状态类型, 用不同平滑因子进行的增益平滑, 见式(125);

$$\hat{g}_c = \begin{cases} 0.17g_0 + 0.83\hat{g}_c, & isf_diff > 0.58 \\ 0.83g_0 + 0.17\hat{g}_c, & isf_diff \leq 0.58 \end{cases} \dots \quad (125)$$

6.3.1.1.7 激励信号生成

每个子帧的激励信号 $\hat{u}(n)$ 由式(126)得到:

6.3.1.1.8 信号合成

每个子帧的重建信号通过式(127)计算:

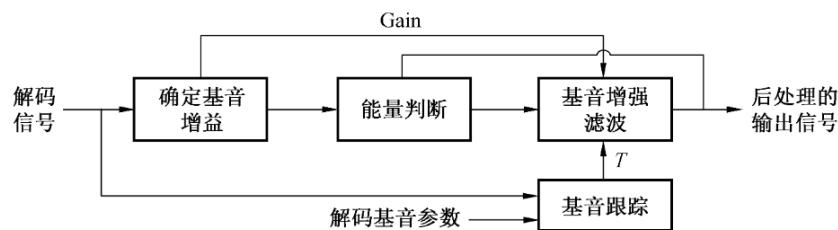
式中：

$\hat{\alpha}_i$ ——子帧的 LP 系数。

6.3.1.1.9 合成信号基音增强

在低码率下,解码音频信号仍会有一定的噪声,因此采取后处理滤波以抑制噪声。基音增强技术通过增强基频及其整数倍数的各谐波分量,抑制位于非谐波分量处的噪声,以提高解码信号的感知质量。基音增强后处理只针对 ACELP 模式解码的低频合成信号,对于 TAC 模式解码信号不进行此后处理。

该基音增强后处理算法如图 56 所示。



说明：

T ——解码信号的基音周期；

Gain ——解码信号的基音增益。

图 56 基音增强后处理算法框图

T 由该子帧解码的闭环基音周期给出。基音跟踪模块为避免出现跟踪倍数基音周期问题,需要计算延迟为 $T/2$ 处信号的归一化自相关值。如果该归一化自相关值大于 0.95, 则将 $T/2$ 作为后处理中的基音周期值。

该后处理算法按每 64 个样点子帧对解码信号 $\hat{s}(n)$ 进行处理, 具体过程如下:

- a) 计算解码信号的基本增益 G_{ai}

确定相邻基音周期的信号能量比值,该比值的计算公式见式(128):

$$E_c = \sqrt{\frac{\sum_{n=0}^{63} \hat{s}^2(n)}{\sum_{n=0}^{63} \hat{s}^2(n-T)}} \quad \dots \dots \dots \quad (128)$$

将该比值 E_c 与从码流中解码获得的基音增益进行比较, 取其中较小的一个作为最终解码信号的基音增益 Gain。

- b) 判断 a) 中计算的能量比值 E_c 是否超过预定阈值 E_t 。若超过，则执行 c)，进行基音增强滤波处理；否则不进行基音增强滤波处理，直接将解码信号 $\hat{s}(n)$ 作为最终的信号 $s_E(n)$ 输出，见式 (129)：

$$s_E(n) = \begin{cases} \hat{s}(n), & E_c < E_t \\ \hat{s}(n) \otimes h(n), & E_c \geq E_t \end{cases} \quad \dots \dots \dots (129)$$

武中。

$h(n)$ —基音增强滤波器的脉冲响应函数。

E_c ——预定阈值, 其值固定为 0.6。

- ### c) 基音增强滤波

基音增强滤波器的传输函数见式(130):

武中

G_1 ——滤波器的全局增益：

λ ——局部调整因子,其值固定为 0.1。

金属增益 G_1 的计算式见(131):

式中：

Gain——a)由計算的甚麼增益

基音增强滤波的输出信号见式(132):

6.3.1.2 TVC 解码

6.3.1.2.1 TVC 解码过程

TVC解码过程如图57所示。

解码器获得的码流信息：缩放因子，量化频谱值，全局增益和噪声因子。

TVC 的解码步骤简述如下：

- a) 解包, 获取 TVC 编码参数;
 - b) 量化频谱值进行基于变长分裂表的反矢量量化;
 - c) 增益平衡, 消除缩放因子的影响;
 - d) 峰值逆整形;
 - e) 逆时频变换, 信号由频域变换到时域, 得到的时域信号与全局增益相乘;
 - f) 加自适应窗和重叠加, 为下一帧保存重叠信号, 自适应窗的窗长和窗型定义见 6.2.4.3;
 - g) 通过逆感知加权滤波器得到合成音频信号;
 - h) 如果上一帧使用 ACELP 模式编码, 那么需要进行帧间平滑处理。

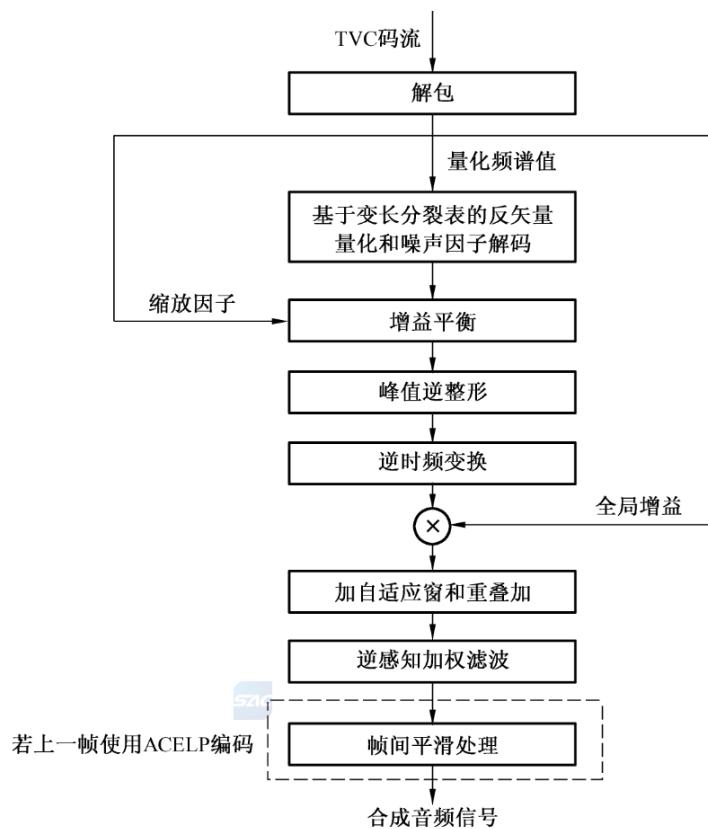


图 57 TVC 解码过程图

6.3.1.2.2 量化频谱解码

频谱的矢量量化, TVC 使用了基础码书编码、一级分裂扩展编码和二级分裂扩展编码三种方法, 详

见 6.2.4.8。因此在解码中,需要根据不同编码方法进行解码操作。首先解包码流中的编码参数,通过每组数据中参数 header 值来判断这组数据使用的是哪种编码方法。如果为基础码书编码,则根据 header 和索引 i 的值直接计算码字 y,然后由偶标志位来判断是否对码字进行加 1 的处理,并将其输出;如果为分裂表编码,则先计算相对应的基础码书中的码字 c,再加上分裂量还原为 y,然后由偶标志位判断是否进行加 1 的处理,最后输出 y。图 58 给出了量化频谱解码流程。

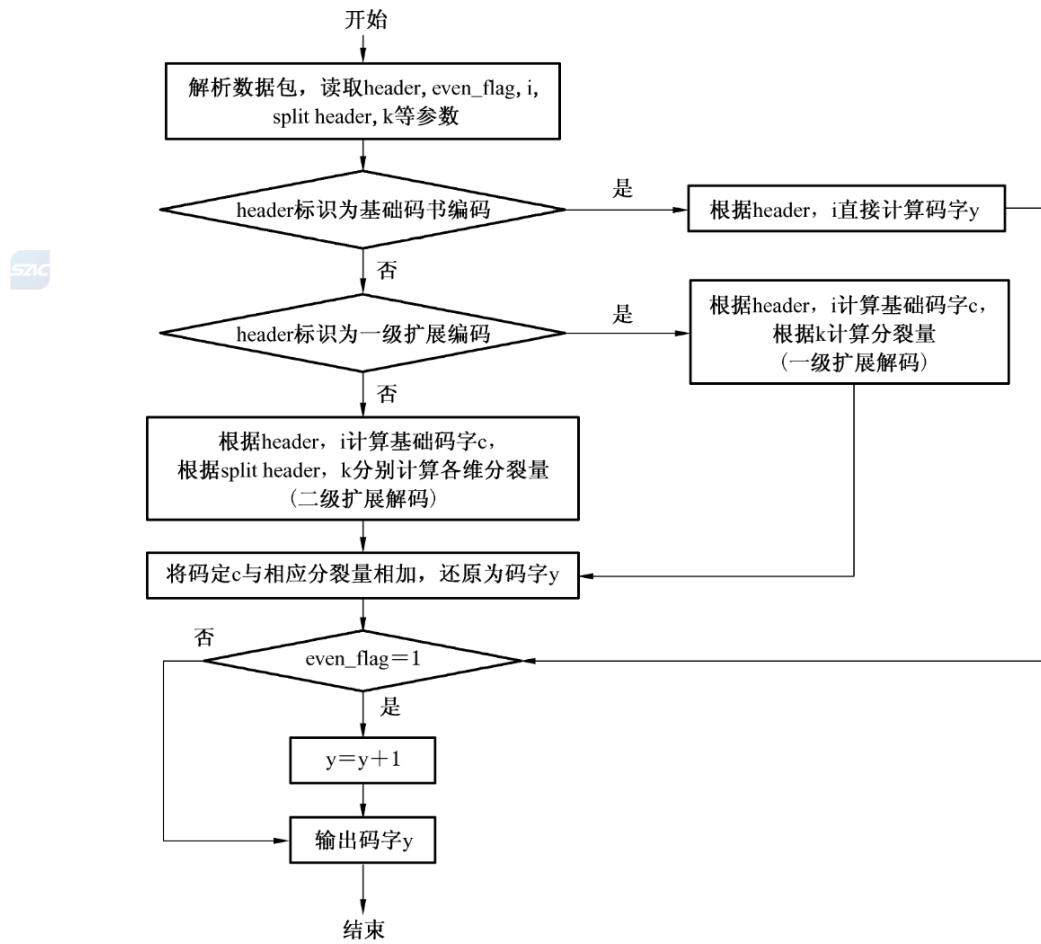


图 58 量化频谱解码过程图

6.3.1.2.3 解码噪声因子

噪声因子采用 3 比特量化,见 6.2.4.6。对接收到的 3 比特编码索引($0 \leqslant idx \leqslant 7$),解码的噪声因子为 $\delta_{noise} = 0.1 \times (8 - idx)$ 。对于 $K/6 \leqslant k \leqslant K$ 范围内的频谱矢量,如果解码的频谱矢量为零矢量,则使用下面的 8 维随机矢量代替,见式(133):

$$\delta_{noise} \times [\cos\theta_1 \quad \sin\theta_1 \quad \cos\theta_2 \quad \sin\theta_2 \quad \cos\theta_3 \quad \sin\theta_3 \quad \cos\theta_4 \quad \sin\theta_4] \dots \dots \dots \quad (133)$$

式中相位 $\theta_1, \theta_2, \theta_3$ 和 θ_4 是随机选取。

6.3.1.2.4 增益平衡及峰值逆整形

反量化频谱值通过增益平衡,即通过解码得到的两个缩放因子信息,消除缩放因子的影响,经峰值逆整形和逆时频变换后得到时域信号,最后乘以全局增益即可得到重建的时域信号,其对应解码框图见图 59 所示。增益平衡见 6.2.4.9,峰值逆整形见 6.2.4.10,逆时频变换见 6.2.4.11,全局增益因子的反量

化见 6.2.4.12。

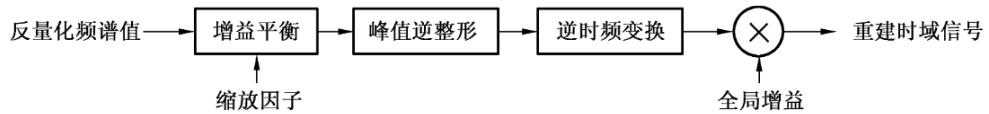


图 59 增益平衡和频谱逆整形的解码框图

6.3.1.2.5 帧间平滑处理

如果上一帧采用 ACELP 编码,为了消除编码模式切换的影响,则需要对上一帧 ACELP 解码得到的音频信号和当前帧 TVC 解码得到的音频信号的重叠部分进行加窗和重叠相加操作。重叠部分采用的三角窗如图 60,对 TVC 帧重叠的音频信号用 $w_1(n)$ 加权,对 ACELP 帧重叠的音频信号用 $w_2(n)$ 加权,其定义见式(134):

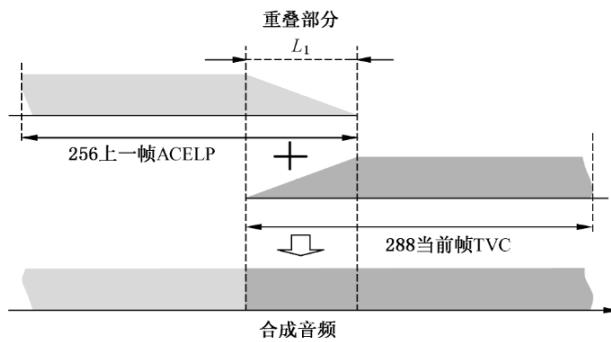


图 60 帧间平滑处理示意图

6.3.2 高频信号解码

6.3.2.1 高频信号解码过程

高频信号解码使用一种带宽扩展机制，并且需要用到低频信号解码中的一些数据，高频信号解码过程见图 6-1 所示。高频信号解码分为两步：计算高频激励信号和合成滤波。解码过程为：对低频激励信号进行增益调整得到高频的激励信号，然后通过高频 LP 合成滤波器得到合成信号。

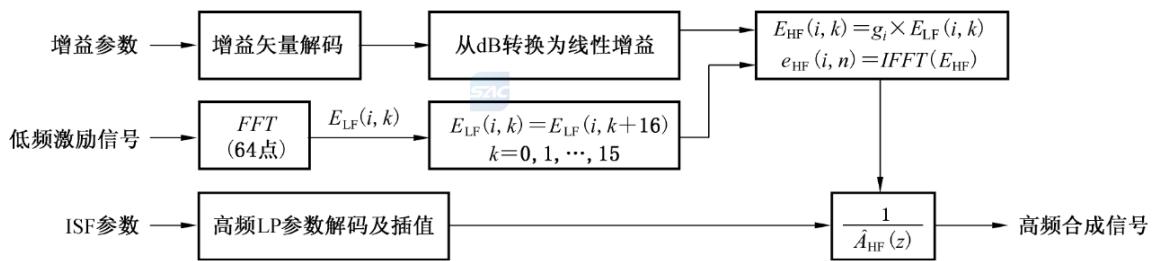


图 61 高频解码过程

6.3.2.2 高频参数解码

高频解码的参数包括：ISF 参数和增益参数。ISF 参数用来生成合成滤波器 $1/\hat{A}_{HF}(z)$ ，增益参数用来对低频激励信号进行整形。

ISF 矢量 isf_{hf_q} 编码使用的是基于预测的 MSVQ。定义 2 比特索引 i_1 表示第一级码书索引，7 比特索引 i_2 表示第二级码书，则 isf_{hf_q} 的计算见式(135)：

$$isf_{hf_q} = cb1(i_1) + cb2(i_2) + mean_isf_hf + \mu_{isf_hf} \times mem_isf_hf \quad \dots\dots\dots(135)$$

式中：

$cb1(i_1)$ ——第一级码书的第 i_1 个码矢量；

$cb2(i_2)$ ——第二级码书的第 i_2 个码矢量；

$mean_isf_hf$ ——ISF 矢量的平均值；

$\mu_{isf_hf} = 0.5$ ——预测系数；

mem_isf_hf ——ISF 预测器的存储器，它的更新如下：

$$mem_isf_hf = isf_{hf_q} - mean_isf_hf \quad (mem_isf_hf \text{ 初始值为 } 0)$$

解码的 ISF 参数先转换为 ISP 系数，再按照 6.2.3.2.7 中 ISP 系数的插值方法计算每个子帧 ISP 系数，然后将插值后的 ISP 系数转换为 LP 系数，用于 LP 合成滤波器来重建信号。

根据 4 维的高频残差增益 VQ 码表的 7 比特索引，残差增益解码公式见式(136)：

$$(g_0, g_1, g_2, g_3) = cb_gain_hf(idx) \quad \dots\dots\dots(136)$$

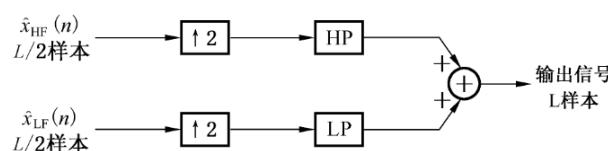
式中：

$cb_gain_hf(idx)$ ——码书 cb_gain_hf 的第 idx 个码矢量。

最终每个子帧的高频增益转换线性表示为 $10^{g_i/20}$ 。通过增益与相应低频拷贝频谱乘积获得高频残差分量，最后通过高频合成滤波输出高频信号。

6.3.3 解码后处理

将 $F_s/2$ 采样的低频解码信号 $\hat{x}_{LF}(n)$ 和高频解码信号 $\hat{x}_{HF}(n)$ 经过上采样恢复到 F_s 采样频率，然后相加就得到全带输出信号，见图 62。



L——音频超帧长度。

图 62 低频和高频信号合成

如果输出信号的采样频率不同于 F_s ，则需要进行采样频率转换。

6.3.4 识别特征参数的解码

从编码码流中提取码书索引，VAD 标志位以及 CRC 校验和。根据码书索引，从 VQ 码书中查找得到估计矢量，见式(137)：

$$\begin{bmatrix} \hat{y}_i(t) \\ \hat{y}_{i+1}(t) \end{bmatrix} = q_{idx_i, i+1(t)}^{i, i+1}, \quad i = \{0, 2, 4, \dots, 12\} \quad \dots\dots\dots(137)$$

在直接编码模式下,估计矢量就是解码的识别特征参数。在预测编码模式下,估计矢量只是残差矢量,需要对音频解码器输出的重建音频信号提取特征矢量,最后同解码得到的残差矢量相加得到解码的识别特征参数。

6.4 比特分配描述

6.4.1 音频编码器的比特分配描述

表 168 和表 169 给出了 ACELP 和 TVC 编码模式下音频码流的比特分配表,这些表给出了音频编码器产生的比特流的顺序,在输出比特时编码器先输出每个编码参数的 MSB。

表 168 ACELP 编码模式下的比特分配表

单位为比特每帧

| 描述 | 比特(MSB-LSB) | | | | | | | |
|---------------|-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | 488 | 424 | 392 | 344 | 312 | 280 | 248 | 216 |
| 模式位 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 |
| 填充位 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 |
| 感知加权标志位 | b9 | b9 | b9 | b9 | b9 | b9 | b9 | b9 |
| 第 1 个 ISP 子矢量 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 |
| 第 2 个 ISP 子矢量 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 |
| 第 3 个 ISP 子矢量 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 |
| 第 4 个 ISP 子矢量 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 |
| 第 5 个 ISP 子矢量 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 |
| 平均能量索引 | b56~b57 | b56~b57 | b56~b57 | b56~b57 | b56~b57 | b56~b57 | b56~b57 | b56~b57 |
| 子帧 1 | | | | | | | | |
| 自适应码书索引 | b58~b66 | b58~b66 | b58~b66 | b58~b66 | b58~b66 | b58~b66 | b58~b66 | b58~b66 |
| LTP 滤波标志 | b67 | b67 | b67 | b67 | b67 | b67 | b67 | b67 |
| 代数码书索引 | b68~b155 | b68~b139 | b68~b131 | b68~b119 | b68~b111 | b68~b103 | b68~b95 | b68~b87 |
| 码书增益索引 | b156~b162 | b140~b146 | b132~b138 | b120~b126 | b112~b118 | b104~b110 | b96~b102 | b88~b94 |
| 子帧 2 | | | | | | | | |
| 自适应码书索引 | b163~b168 | b147~b152 | b139~b144 | b127~b132 | b119~b124 | b111~b116 | b103~b108 | b95~b100 |
| LTP 滤波标志 | b169 | b153 | b145 | b133 | b125 | b117 | b109 | b101 |
| 代数码书索引 | b170~b257 | b154~b225 | b146~b209 | b134~b185 | b126~b169 | b118~b153 | b110~b137 | b102~b121 |
| 码书增益索引 | b258~b264 | b226~b232 | b210~b216 | b186~b192 | b170~b176 | b154~b160 | b138~b144 | b122~b128 |
| 子帧 3 | | | | | | | | |
| 自适应码书索引 | b265~b273 | b233~b241 | b217~b225 | b193~b201 | b177~b185 | b161~b169 | b145~b153 | b129~b137 |
| LTP 滤波标志 | b274 | b242 | b226 | b202 | b186 | b170 | b154 | b138 |
| 代数码书索引 | b275~b362 | b243~b314 | b227~b290 | b203~b254 | b187~b230 | b171~b206 | b155~b182 | b139~b158 |
| 码书增益索引 | b363~b369 | b315~b321 | b291~b297 | b255~b261 | b231~b237 | b207~b213 | b183~b189 | b159~b165 |

表 168 (续)

单位为比特每帧

| 描述 | 比特(MSB-LSB) | | | | | | | |
|-----------|-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | 488 | 424 | 392 | 344 | 312 | 280 | 248 | 216 |
| 子帧 4 | | | | | | | | |
| 自适应码书索引 | b370~b375 | b322~b327 | b298~b303 | b262~b267 | b238~b243 | b214~b219 | b190~b195 | b166~b171 |
| LTP 滤波标志 | b376 | b328 | b304 | b268 | b244 | b220 | b196 | b172 |
| 代数码书索引 | b377~b464 | b329~b400 | b305~b368 | b269~b320 | b245~b288 | b221~b256 | b197~b224 | b173~b192 |
| 码书增益索引 | b465~b471 | b401~b407 | b369~b375 | b321~b327 | b289~b295 | b257~b263 | b225~b231 | b193~b199 |
| 带宽扩展 | | | | | | | | |
| 高频 ISP 索引 | b472~b480 | b408~b416 | b376~b384 | b328~b336 | b296~b304 | b264~b272 | b232~b240 | b200~b208 |
| 高频增益索引 | b481~b487 | b417~b423 | b385~b391 | b337~b343 | b305~b311 | b273~b279 | b241~b247 | b209~b215 |

表 169 TVC 编码模式下的比特分配表

单位为比特每帧

| 描述 | 比特(MSB-LSB) | | | | | | | |
|---------------|-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | 488 | 424 | 392 | 344 | 312 | 280 | 248 | 216 |
| 模式位 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 | b0~b3 |
| 填充位 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 | b4~b8 |
| 感知加权标志位 | b9 | b9 | b9 | b9 | b9 | b9 | b9 | b9 |
| 第 1 个 ISP 子矢量 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 | b10~b19 |
| 第 2 个 ISP 子矢量 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 | b20~b28 |
| 第 3 个 ISP 子矢量 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 | b29~b37 |
| 第 4 个 ISP 子矢量 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 | b38~b46 |
| 第 5 个 ISP 子矢量 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 | b47~b55 |
| 噪声因子 | b56~b58 | b56~b58 | b56~b58 | b56~b58 | b56~b58 | b56~b58 | b56~b58 | b56~b58 |
| 全局增益 | b59~b65 | b59~b65 | b59~b65 | b59~b65 | b59~b65 | b59~b65 | b59~b65 | b59~b65 |
| 缩放因子 | b66~b72 | b66~b72 | b66~b72 | b66~b72 | b66~b72 | b66~b72 | b66~b72 | b66~b72 |
| 代数 VQ 参数 | b73~b471 | b73~b407 | b73~b375 | b73~b327 | b73~b295 | b73~b263 | b73~b231 | b73~b199 |
| 带宽扩展 | | | | | | | | |
| 高频 ISP 索引 | b472~b480 | b408~b416 | b376~b384 | b328~b336 | b296~b304 | b264~b272 | b232~b240 | b200~b208 |
| 高频增益索引 | b481~b487 | b417~b423 | b385~b391 | b337~b343 | b305~b311 | b273~b279 | b241~b247 | b209~b215 |

6.4.2 识别特征参数编码的比特分配描述

见 6.2.7.2 的描述。

6.5 存储、传输接口格式

6.5.1 编码模式和码率

表 170 规定了音频编码的编码模式和码率。

表 170 编码模式和码率

| 编码模式 | 每帧比特数 | 25.6 kHz 码率 kbps |
|------|-------|---------------------|
| 0 | 216 | 10.8 |
| 1 | 248 | 12.4 |
| 2 | 280 | 14.0 |
| 3 | 312 | 15.6 |
| 4 | 344 | 17.2 |
| 5 | 392 | 19.6 |
| 6 | 424 | 21.2 |
| 7 | 488 | 24.4 |

每个音频帧由 512 个样本组成,但时间长度却依赖于内部采样频率 F_s ,表 171 列出了音频编码器支持的内部采样频率。

表 171 内部采样频率和对应的每帧时长

| 内部采样频率索引 | 内部采样频率 Hz | 帧长度 ms | 内部采样频率因子 |
|----------|--------------|-----------|----------|
| 0 | 保留 | 保留 | 保留 |
| 1 | 12 800 | 40 | 1/2 |
| 2 | 16 000 | 32 | 5/8 |
| 3 | 24 000 | 21.33 | 15/16 |
| 4 | 25 600 | 20 | 1 |
| 5 | 32 000 | 16 | 5/4 |
| 6 | 38 400 | 13.33 | 3/2 |

表 171 中的最后一列采样频率因子 = 内部采样频率/25 600。内部采样频率索引在码流中用来指明每帧使用了哪种内部采样频率。

表 172 列出的音频编码的帧类型用来表示每个音频帧的编码模式和字节数。编码帧类型和内部采样频率这两个参数共同决定了编码码率。

表 172 音频编码的帧类型

| 帧类型 | 编码模式 | 25.6 kHz 码率 kbps | 每帧字节数 |
|------|------|---------------------|-------|
| 0 | 0 | 10.8 | 27 |
| 1 | 1 | 12.4 | 31 |
| 2 | 2 | 14.0 | 35 |
| 3 | 3 | 15.6 | 39 |
| 4 | 4 | 17.2 | 43 |
| 5 | 5 | 19.6 | 49 |
| 6 | 6 | 21.2 | 53 |
| 7 | 7 | 24.4 | 61 |
| 8~15 | | 保留 | |

识别特征参数编码提供两种编码模式：直接编码模式和预测编码模式。直接编码模式码率为 4.8 kbps，对应每帧数据 48 比特，而预测编码模式码率为 3.2 kbps，对应每帧数据 32 比特。

码流数据打包时，为了节省帧头开销，每个帧头后紧跟一个音频超帧数据。每个音频超帧由若干音频帧组成，目前本标准规定音频超帧中只包含一个音频帧。在每个音频超帧数据中，先打包所有的音频帧码流，再打包所有的识别特征参数帧码流。由于识别特征参数帧的时间长度和音频帧的时间长度并不相同，因此在帧头中编码 1 比特标志位来表示当前音频超帧包括了 N 帧的识别特征参数码流还是 N+1 帧的识别特征参数码流。N 取值同音频编码的内部采样频率和音频超帧的长度有关，计算如下：

$$N = \text{Floor} \left(\frac{\text{超帧的样本点数}}{\text{内部采样频率(Hz)} \times 0.01} \right)$$

音频支持两种码流格式：RAW 格式和 NAL 格式。NAL 格式是在 RAW 格式上增加了 NAL 层封装。字节流 NAL 单元语法见 5.2.3.1。

6.5.2 音频数据 RAW 格式

6.5.2.1 音频数据单元格式

音频数据单元 audio_data_unit() 定义见表 173，一个音频数据单元打包一个音频超帧数据。

表 173 音频数据单元的格式

| audio_data_unit() { | 描述符 | 说明 |
|---------------------------------|-----|------------------------|
| audio_frame_header() | | 帧头，包括编码参数和码流标志位 |
| if(extension_flag) | | |
| audio_frame_extension_header() | | 扩展帧头，包括编码扩展信息 |
| audio_frame_data() | | 音频超帧数据，包括音频码流和识别特征参数码流 |
| } | | |

帧头 audio_frame_header() 定义见表 174。

表 174 帧头格式

| audio_frame_header(){ | 描述符 | 说明 |
|------------------------|------|-------------------------------------|
| version_id | u(1) | 编码器的版本号,0 表示 1.0,其他值保留 |
| profile_id | u(2) | 码流的档次,档次定义见表 G.1 |
| level_id | u(4) | 码流的级别,级别定义见表 G.2 |
| bitstream_type | u(1) | 码流的内容:0 表示只有音频码流,1 表示有音频码流和识别特征参数码流 |
| isf_index | u(3) | 音频编码采用的内部采样频率索引,具体索引定义见表 92 |
| frame_type | u(4) | 音频编码的帧类型,具体帧类型定义见表 93 |
| extension_flag | u(1) | 扩展帧头标志位:0 表示不编码,1 表示编码 |
| } | | |

扩展帧头 audio_frame_extension_header() 定义见表 175。

表 175 扩展帧头格式

| audio_frame_extension_header(){ | 描述符 | 说明 |
|----------------------------------|-------|--|
| feature_type | u(2) | 编码的识别特征参数:0 表示 MFCC,其他值保留 |
| feature_mode | u(1) | 识别特征参数的编码模式:0 表示直接编码,1 表示预测编码 |
| feature_bs_flag | u(1) | 音频超帧中识别特征参数码流帧数的标志位:0 表示 N 帧识别特征参数码流,1 表示 N+1 帧识别特征参数码流 |
| ref_time_flag | u(1) | 编码绝对时间信息标志位:0 表示不编码,1 表示编码。规定绝对时间表示音频超帧的第 1 个样本的时间 |
| event_flag | u(1) | 编码异常声音事件类型标志位:0 表示不编码,1 表示编码 |
| direction_flag | u(1) | 编码声源定向信息标志位:0 表示不编码,1 表示编码 |
| reserve_bit | u(1) | 保留位,固定为 0 |
| if(ref_time_flag){ | | |
| hour_bits | u(5) | 小时信息,取值范围在 0~23 之间(包括 0 和 23) |
| minute_bits | u(6) | 分钟信息,取值范围在 0~59 之间(包括 0 和 59) |
| second_bits | u(6) | 秒信息,取值范围在 0~59 之间(包括 0 和 59) |
| second_fraction_bits | u(14) | 秒的分数信息,以 1/16 384 s 为单位,取值范围在 0~16 383 之间(包括 0 和 16 383) |
| ref_date_flag | u(1) | 编码日期信息标志位:0 表示不编码,1 表示编码 |
| if(ref_date_flag){ | | |
| year_minus2000_bits | u(7) | 加 2000 表示年份信息,Year = year_minus2000_bits + 2000,取值范围在 0~127 之间(包括 0 和 127) |
| month_minus1_bits | u(4) | 加 1 表示月份信息,Month = month_minus1_bits + 1,取值范围在 0~11 之间(包括 0 和 11) |

表 175 (续)

| audio_frame_extension_header(){ | 描述符 | 说明 |
|----------------------------------|------|---|
| date_minus1_bits | u(5) | 加 1 表示日期信息, Date = date_minus1_bits + 1, 取值范围在 0~30 之间(包括 0 和 30) |
| } | | |
| } | | |
| if(event_flag) | | |
| abnormal_sound_event_type | u(8) | 异常声音事件类型, 具体类型定义见附录 H 中的表 H.1, 取值范围应在 0~255 之间(包括 0 和 255) |
| if(direction_flag){ | | |
| azimuth | u(8) | 声源定向的水平方向角, 编码格式见 6.5.2.2 |
| elevation | u(8) | 声源定向的仰角, 编码格式见 6.5.2.2 |
| } | | |

6.5.2.2 声源定向信息编码格式

声源定向信息包括水平方向的方位角 α 和垂直方向的仰角 β , 角度定义如图 63 所示, 角度有效范围为 $0^\circ \sim 359^\circ$ (360° 等效于 0°)。方位角和仰角分别使用 8 比特编码, 有效范围 $0 \sim 255$, 编码角度的分辨率为 1.40625° 。

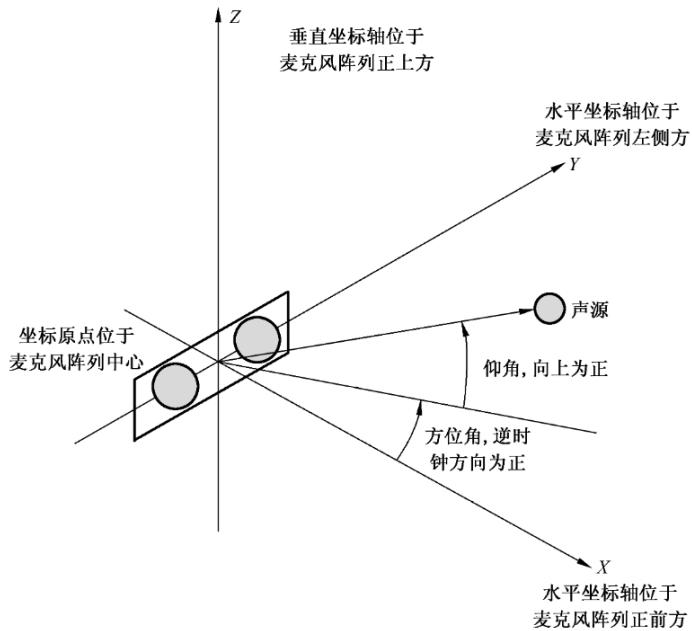


图 63 麦克风阵列的坐标系统

编码时角度量化公式为:

$$\hat{\alpha} = \text{round}(\alpha / 1.40625) \quad \hat{\beta} = \text{round}(\beta / 1.40625)$$

解码时角度的反量化公式为：

$$\alpha' = \text{round}(\hat{\alpha} \times 1.406\ 25) \quad \beta' = \text{round}(\hat{\beta} \times 1.406\ 25)$$

如果前后相邻两帧的声源定向信息没有变化，则置当前帧的扩展帧头中 direction_flag 为零，即当前帧的扩展帧头中不编码声源定向信息，当前帧的声源定向信息同前一帧。

6.5.3 音频数据 NAL 格式

6.5.3.1 音频数据 RBSP 单元

字节流 NAL 单元语法见 5.2.3.1，定义 nal_unit_type 等于 13 的 NAL 单元负载为音频数据 RBSP 单元 audio_data_rbsp()。

音频数据 RBSP 单元 audio_data_rbsp() 定义见表 176。

表 176 音频数据 RBSP 单元

| audio_data_rbsp() | 描述符 | 说明 |
|----------------------------------|-----|---|
| for(i=0; i<audio_unit_num; i++){ | | audio_unit_num 表示一个音频数据 RBSP 单元包括的音频数据单元个数，取值范围应该在 1~256 之间（包括 1 和 256） |
| audio_data_unit() | | 音频数据单元 |
| } | | |
| rbsp_trailing_bits() | | 表示 RBSP 字节流的结束 |
| } | | |

6.5.3.2 音频数据的绝对时间扩展单元

音频数据采用 NAL 格式后，为了同视频部分保持一致，绝对时间信息将采用单独的 NAL 单元来传输。音频数据单元 audio_data_unit() 中不包括绝对时间信息，即 audio_frame_extension_header() 中 ref_time_flag 置为零。

基于 5.2.3.5 中定义的监控扩展数据单元 surveillance_extension_rbsp()，定义扩展单元标识(extension_id) 等于 5 的监控扩展数据单元包含音频数据的绝对时间信息。

音频数据的绝对时间扩展单元 audio_time_extension() 定义见表 177，其中绝对时间信息中未加说明的各语法元素定义同表 26。

表 177 音频数据的绝对时间扩展单元

| audio_time_extension() | 描述符 | 说明 |
|------------------------|-------|-------|
| extension_id | u(8) | 固定为 5 |
| hour_bits | u(5) | |
| minute_bits | u(6) | |
| second_bits | u(6) | |
| second_fraction_bits | u(14) | |
| ref_date_flag | u(1) | |
| if(ref_date_flag) { | | |
| year_minus2000_bits | u(7) | |

表 177 (续)

| audio_time_extension() { | 描述符 | 说明 |
|---------------------------|------|---|
| month_minus1_bits | u(4) | |
| date_minus1_bits | u(5) | |
| } | | |
| frame_num | u(8) | 绝对时间所对应的音频数据 RBSP 单元中的音频数据单元的序号, 规定绝对时间表示所对应音频数据单元的第一个样本的时间, 取值范围应该在 0~255 之间(包括 0 和 255) |
| } | | |

在音频 NAL 单元传输和存储时, 规定音频绝对时间的 NAL 单元同后面最靠近的音频数据的 NAL 单元保持对应关系, 不允许改变这两个 NAL 单元的先后顺序。

附录 A
(规范性附录)
假设参考解码器(HRD)

A.1 概述

本附录定义了假设参考解码器(以下简称 HRD)。每一组 HRD 参数确定一个 HRD 操作模式。HRD 包含一个编码图像缓存区(CPB),一个瞬时解码器和一个解码图像缓存区(DPB),见图 A.1。

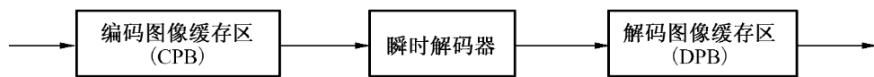


图 A.1 假设参考解码器(HRD)

CPB 大小(比特数)为 $CpbSize[SchedSelIdx]$, DPB 的大小(帧数)为 $\text{Max}(1, \text{max_dec_frame_buffering})$ 。HRD 可被某缓存周期 SEI 消息进行初始化。一旦初始化后,随后的缓存周期 SEI 消息不能再初始化 HRD。如果序列参数集改变,则 HRD 需重新初始化。在初始化后,CPB 与 DPB 均为空。编码图像数据按规定的到达时间表注入 CPB 中。图像序号以 n 表示,从 0 开始,在解码过程中,每当一个图像解码完毕,图像序号加 1。当在 CPB 移除时间到达时,图像数据被移除并由实时解码过程进行实时解码,并放入到 DPB 中。如果该图像需要在 CPB 移除时间时输出,并且为非参考图像,则不放入 DPB 中。DPB 可包含多个帧缓存,这些图像或作为后续解码图像的参考图像,或保留等待以后输出(重排序或延时)。对于放入 DPB 的图像,在其从 DPB 输出时,或不再作为参考图像时,从 DPB 中移除。CPB 的操作在 A.2 中规定,DPB 的操作在 A.3 中规定。

本附录中所有的操作都为实数操作,不存在舍入误差,例如 HRD 缓存区中的比特数可以不是整数值。HRD 使用两种的时间基准为:一个是 90 kHz 时钟,只在 HRD 收到缓存周期 SEI 消息时使用;另一个是序列参数集中定义的时钟,其时钟基准定义为, $tc = \text{num_units_in_tick} \div \text{time_scale}$, 可以认为是图像采样的最短时间间隔。

缓存周期 SEI 消息及其周期内的编码图像数据,应关联于相同的序列参数集。如果码流中包括多序列参数集,其 HRD 参数应一致。

A.2 编码图像缓存区(CPB)

A.2.1 编码图像到达时间

图像 n 的第一个比特进入 CPB 的时间定义为起始到达时间 $t_{ai}(n)$,最后一个比特进入 CPB 的时间定义为最终到达时间 $t_{af}(n)$ 。图像 n 的第一个比特最早进入 CPB 的时间定义为最早到达时间 $t_{ai,earliest}(n)$ 。
 $t_{ai}(n)$ 的计算如下:

如果为第一幅图像,即图像 0,则 $t_{ai}(0)=0$;

否则(为图像 $n, n > 0$),按照如下规则:

——如果 $cbr_flag[SchedSelIdx]$ 等于 1,即恒定比特率情况下,

$$t_{ai}(n) = t_{af}(n-1)$$

——否则($cbr_flag[SchedSelIdx]$ 等于 0),变比特率情况下,

$$t_{ai}(n) = \text{Max}(t_{af}(n-1), t_{ai,earliest}(n))$$

$$t_{ai,earliest}(n) = t_c \times \sum_{m=0}^{n-1} \text{cpb_removal_delay}[m]$$

其中 $\text{cpb_removal_delay}(n)$ 见 D.2.2 定义。

$t_{af}(n)$ 的计算如下：

$$t_{af}(0) = b(0) \div \text{BitRate}[SchedSelIdx]$$

$$t_{af}(n) = t_{ai}(n) + b(n) \div \text{BitRate}[SchedSelIdx]$$

其中 $b(n)$ 为图像 n 的码字长度。

如果图像 n 与图像 $n-1$ 的序列参数集不同, HRD 参数需要重新初始化。如果图像 n 重新初始化了 HRD 参数, 则有

- $\text{BitRate}[SchedSelIdx]$ 在 $t_{ai}(n)$ 时更新;
- 对于 $\text{CpbSize}[SchedSelIdx]$, 如果 $\text{CpbSize}[SchedSelIdx]$ 的新值超过原 CPB 的大小, 在 $t_{ai}(n)$ 时更新;
- 否则 $\text{CpbSize}[SchedSelIdx]$ 的新值在 $t_r(n)$ 时更新。

A.2.2 编码图像移除时间

图像 n 从 CPB 中移除的标定时间定义为标定移除时间 $t_{r,n}(n)$, 从 CPB 中移除的实际时间定义为实际移除时间 $t_r(n)$ 。

$t_{r,n}(n)$ 的计算如下:

对于图像 0,

$$t_{r,n}(0) = \text{initial_cpb_removal_delay}[SchedSelIdx] \div 90\,000$$

对于图像 n ,

$$t_{r,n}(n) = t_{r,n}(n-1) + t_c \times \text{cpb_removal_delay}(n)$$

$t_{r,n}(n-1)$ 为当前缓存周期前一个图像的标定移除时间, $\text{cpb_removal_delay}(n)$ 为与图像 n 关联的图像定时 SEI 消息中规定的 cpb_removal_delay 的值。如果图像 n 为未初始化 HRD 的缓存周期的第一个图像, 则上式中的 $t_{r,n}(n-1)$ 为前一个缓存周期的最后一个图像的标定移除时间。

$t_r(n)$ 的计算如下:

如果 $\text{low_delay_hrd_flag}$ 等于 0 或 $t_{r,n}(n) \geq t_{af}(n)$,

$$t_r(n) = t_{r,n}(n)$$

否则 ($\text{low_delay_hrd_flag}$ 等于 1 或 $t_{r,n}(n) < t_{af}(n)$),

$$t_r(n) = t_{r,n}(n) + t_c \times \text{Ceil}((t_{af}(n) - t_{r,n}(n)) \div t_c)$$

后一种情况防止 $b(n)$ 很大时的非正常移除。

A.3 解码图像缓存区(DPB)

A.3.1 图像的输出和移除

图像 n 解码后, 其 DPB 输出时间 $t_{o,dpb}(n)$ 如下:

$$t_{o,dpb}(n) = t_r(n) + t_c \times \text{dpb_output_delay}(n)$$

如果无延迟, $t_{o,dpb}(n) = t_r(n)$, 当前图像直接输出。如果为参考图像, 则还需将当前图像存储于 DPB 中; 否则 ($t_{o,dpb}(n) > t_r(n)$), 当前图像延迟输出, 并存储于 DPB。

当满足如下两个条件时, DPB 中的某图像 m 将被移除:

- 图像 m 不作为后续解码图像的参考图像;
- 图像 m 的 DPB 输出时间小于或等于 CPB 中当前图像 n 的移除时间, 即 $t_{o,dpb}(m) \leq t_r(n)$ 。

当帧缓存里所有数据从 DPB 里移除后,DPB 填充度减 1。

A.3.2 IDR 图像的插入

如果解码图像为 IDR 图像,有如下操作:

- a) 所有在 DPB 中的图像均不作为后续解码图像的参考图像;
- b) 当其不是第一幅 IDR 图像,且根据序列参数集得到的 FrameWidth, FrameHeight, max_dec_frame_buffering 的值与前面图像对应的序列参数集中定义的值不同时,DPB 中所有的帧缓存清空且不输出,DPB 填充度重置为 0。

A.3.3 图像的标记和存储

如果当前图像为参考图像,或者为非参考图像但 $t_{o,dpb}(n) > t_r(n)$,则需要把当前图像存储于 DPB 中。如果需要存储,则当前解码图像存在一个空的帧缓存中,DPB 填充度加 1。



附录 B
(规范性附录)
字节流的格式

B.1 概述

本附录规定了字节流格式的语法与语义,用于将 NAL 单元流生成有序的字节流,并且 NAL 单元边界位置应能够从字节流中被识别。对于面向比特的传送,字节流中的比特顺序起始于第一个字节的 MSB,处理至第一个字节的 LSB,然后为第二个字节的 MSB,以此类推。

字节流格式由一系列字节流 NAL 单元语法结构组成。每个字节流 NAL 单元语法结构包含有一个起始码前缀,后面跟随一个 NAL 单元。字节流 NAL 单元语法结构中可能包含一个额外的 zero_byte 语法元素,也可能包含一个或多个额外的 trailing_zero_8bits 语法元素。第一个字节流 NAL 单元语法结构中还可能包含一个或多个额外的 leading_zero_8bits 语法元素。

B.2 字节流 NAL 单元语法与语义

B.2.1 字节流 NAL 单元语法

字节流 NAL 单元语法见表 B.1。

表 B.1 字节流 NAL 单元语法表

| 描述符 |
|---|
| byte_stream_nal_unit(NumBytesInNALunit) { |
| while(next_bits(24) != 0x000001 && next_bits(32) != 0x00000001) |
| leading_zero_8bits /* 应等于 0x00 */ |
| if(next_bits(24) != 0x000001) |
| zero_byte /* 应等于 0x00 */ |
| start_code_prefix_one_3bytes /* 应等于 0x000001 */ |
| nal_unit(NumBytesInNALunit) |
| while(more_data_in_byte_stream() && next_bits(24) != 0x000001 && next_bits(32) != 0x00000001) |
| trailing_zero_8bits /* 应等于 0x00 */ |
| } |

B.2.2 字节流 NAL 单元语义

字节流 NAL 单元中 NAL 单元的顺序应遵循 NAL 单元解码顺序,见 5.2.4.3.2。

leading_zero_8bits 应等于 0x00。

注: leading_zero_8bits 语法元素只可能在流的第一个字节流 NAL 单元里出现。

zero_byte 应等于 0x00。

当下述任一个条件满足时,应有 zero_byte 语法元素。

——NAL 单元中的 nal_unit_type 等于 7(序列参数集)或 8(图像参数集)。

——字节流 NAL 单元中包含一个基本编码图像的第一个 NAL 单元,见 5.2.4.3.2.3。

start_code_prefix_one_3bytes 为一个 3 字节的固定值序列,等于 0x000001,该语法元素称为起始码前缀。

trailing_zero_8bits 应等于 0x00。

B.3 字节流 NAL 单元解码过程

本过程的输入为字节流,该字节流由一系列字节流 NAL 单元语法结构组成。

本过程的输出为一系列的 NAL 单元。

在解码过程开始时,解码器把其当前位置初始化为字节流的起始位置。然后提取并丢弃每一个 **leading_zero_8bits** 语法元素(如果存在的话),并相应移动当前位置,直到当前位置紧接的四个字节为 0x00000001。

解码器此时重复执行下述按步骤的过程,对字节流中每一个 NAL 单元语法结构进行提取与解码,直到字节流中最后一个 NAL 单元也已经解码:

- a) 当字节流里的紧接的四个字节构成四字节序列 0x00000001,对比特流中下一个字节(为 **zero_byte** 语法元素)进行提取并丢弃时,字节流的当前位置设为紧接被丢弃的字节的字节位置;
- b) 提取与丢弃比特流中下一个三字节序列(为 **start_code_prefix_one_3bytes**),且比特流当前位置设为此紧接被丢弃的 3 字节序列的字节的位置;
- c) NumBytesInNALUnit 设为自当前字节位置起至满足下述任一条件的位置的最后一个字节,且包括最后一个字节的编号;
 - 1) 一个三字节序列的排列等于 0x000000,或
 - 2) 一个三字节序列的排列等于 0x000001,或
 - 3) 字节流的结束。
- d) 该 NumBytesInNALUnit 字节从比特流中移除,字节流的当前位置前移 NumBytesInNALUnit 字节。这个字节序列为 **nal_unit**(NumBytesInNALUnit),并用 NAL 单元解码过程进行解码;
- e) 当字节流中的当前位置不为字节流的结尾,且字节流中一个字节不是等于 0x000001 开始的三字节序列,也不是等于 0x00000001 开始的四字节序列,解码器提取并丢弃每一个 **trailing_zero_8bits** 语法元素,并相应移动当前位置,直到字节流里的当前位置接下的四个字节构成四字节的序列 0x00000001 或已至字节流的结尾。

注:字节流的结束的判断方法不在本标准中规定。

附录 C
(规范性附录)
视频档次与级别

C.1 概述

档次与级别规定了对比特流的限制,因此也限制了比特流解码所需的能力。每一个档次定义了一个算法特征的子集,并限定所有与该档次一致的解码器均应支持。每一个级别定义了对本标准中的语法元素取值的限制集合。相同的级别定义集合用于所有的档次,但单独的应用对所支持的档次可能支持不同的级别。一般来说,对于特定的一个档次,不同的级别对应于对解码器负荷和存储器容量的不同要求。

本附录描述了视频不同档次和级别所对应的各种限制。所有未被限定的语法元素和参数可以取任何本标准所允许的值。如果一个解码器能对某个档次和级别所规定的语法元素的所有允许值正确解码,则称此解码器在这个档次和级别上符合本标准。如果一个比特流中不存在某个档次和级别所不允许的语法元素,并且其所含有的语法元素的值不超过此档次和级别所允许的范围,则认为此比特流在这个档次和级别上符合本标准。

profile_id 和 level_id 定义了比特流的档次和级别。

注:解码器不宜因为 profile_id 或 level_id 的取值落在本标准所规定的值之间,就推定这个值所代表的能力处于规定的档次与级别之间。

C.2 视频档次

C.2.1 视频档次的定义

视频档次的定义见表 C.1。

表 C.1 视频档次

| profile_id | 档次 |
|------------|------|
| 0x00 | 禁止 |
| 0x11 | 基准档次 |
| 0x22 | 保留 |
| 0x33 | 高级档次 |

C.2.2 基准档次

比特流若与基准档次相一致,应遵循如下限制:

- profile_id 的值应为 0x11;
- NAL 单元流中不应包含 nal_unit_type 的取值为 3、4 和 15 的 NAL 单元;
- 序列参数集中的参数 ldp_mode_flag 的取值应为 1;
- 序列参数集中的参数 chroma_format_idc 的取值应为 1;
- 序列参数集中的参数 bit_depth 的取值应为 0;

- 序列参数集中的参数 refs_per_frame 的取值应不大于 3；
- 序列参数集中的参数 extended_sb_size_flag 的取值应为 0；
- 序列参数集中的参数 alf_enable 的取值应为 0；
- 序列参数集中的参数 spatial_svc_flag 的取值应为 0；
- 解码所需的参考帧缓冲区个数应不大于 4。

profile_id 的取值为 0x11 时，比特流与基准档次相一致。基准档次的比特流支持的级别包括 6.0, 7.0 和 8.0。

C.2.3 高级档次

比特流若与高级档次相一致，应遵循如下限制：

- profile_id 的值应为 0x33；
- 序列参数集中的参数 ldp_mode_flag 的取值应为 0 或 1；
- 序列参数集中的参数 chroma_format_idc 的取值应为 1 或 2；
- 序列参数集中的参数 bit_depth 的取值应在 0~2 之间，包括 0 和 2；
- 序列参数集中的参数 refs_per_frame 的取值应不大于 5；
- 序列参数集中的参数 extended_sb_size_flag 的取值应为 0 或 1；
- 序列参数集中的参数 alf_enable 的取值应为 0 或 1；
- 序列参数集中的参数 spatial_svc_flag 的取值应为 0 或 1；
- 解码所需的参考帧缓冲区个数应不大于 8。

profile_id 的取值为 0x33 时，比特流与高级档次相一致。高级档次支持的级别有 6.0, 6.2, 7.0, 7.2, 8.0 和 8.2。

C.3 视频级别

C.3.1 视频级别的定义

视频级别的定义见表 C.2。

表 C.2 视频级别

| level_id | 级别 |
|----------|-----|
| 0x00 | 禁止 |
| 0x40 | 6.0 |
| 0x42 | 6.2 |
| 0x50 | 7.0 |
| 0x52 | 7.2 |
| 0x60 | 8.0 |
| 0x62 | 8.2 |
| 其他 | 保留 |

C.3.2 级别的参数限制

不同级别对应的参数限制见表 C.3~表 C.5。

表 C.3 级别的参数限制

| 参数 | 级别 | |
|---------------------|------------|-----------------------|
| | 6.0 | 6.2 |
| 每行最大样点数 | 1 920 | 1 920 |
| 每帧最大行数 | 1 088 | 1 088 |
| 每秒最大帧数 | 30 | 30 |
| 亮度样点速率 | 62 668 800 | 62 668 800 |
| 最大比特率 MaxBR(bps) | 15 000 000 | 25 000 000 |
| 图像格式 | 4 : 2 : 0 | 4 : 2 : 0 或 4 : 2 : 2 |
| DPB 最大容量 MaxDPB(kB) | 12 240 | 24 480 |
| CPB 最大容量 MaxCPB(kB) | 20 000 | 40 000 |

表 C.4 级别的参数限制

| 参数 | 级别 | |
|---------------------|-------------|-----------------------|
| | 7.0 | 7.2 |
| 每行最大样点数 | 2 592 | 2 592 |
| 每帧最大行数 | 1 944 | 1 944 |
| 每秒最大帧数 | 30 | 30 |
| 亮度样点速率 | 151 165 440 | 151 165 440 |
| 最大比特率 MaxBR(bps) | 30 000 000 | 50 000 000 |
| 图像格式 | 4 : 2 : 0 | 4 : 2 : 0 或 4 : 2 : 2 |
| DPB 最大容量 MaxDPB(kB) | 29 525 | 59 049 |
| CPB 最大容量 MaxCPB(kB) | 40 000 | 80 000 |

表 C.5 级别的参数限制

| 参数 | 级别 | |
|---------------------|-------------|-----------------------|
| | 8.0 | 8.2 |
| 每行最大样点数 | 4 096 | 4 096 |
| 每帧最大行数 | 2 304 | 2 304 |
| 每秒最大帧数 | 30 | 30 |
| 亮度样点速率 | 283 115 520 | 283 115 520 |
| 最大比特率 MaxBR(bps) | 50 000 000 | 80 000 000 |
| 图像格式 | 4 : 2 : 0 | 4 : 2 : 0 或 4 : 2 : 2 |
| DPB 最大容量 MaxDPB(kB) | 56 624 | 113 248 |
| CPB 最大容量 MaxCPB(kB) | 80 000 | 160 000 |

注：表 C.3~表 C.5 中 DPB 最大容量 (MaxDPB) 以 4 : 2 : 0 格式 8 比特样点为例。其中 1 kB 等于 1 024 字节。

附录 D
(规范性附录)
视频可用性信息(VUI)

D.1 视频可用性信息(VUI)语法

D.1.1 视频可用性信息(VUI)参数语法

VUI 参数语法见表 D.1。

表 D.1 VUI 参数语法表

| | 描述符 |
|------------------------------------|-------|
| vui_parameters() | |
| timing_info_present_flag | u(1) |
| if(timing_info_present_flag) { | |
| num_units_in_tick | u(32) |
| time_scale | u(32) |
| fixed_frame_rate_flag | u(1) |
| } | |
| hrd_parameters_present_flag | u(1) |
| if(hrd_parameters_present_flag) | |
| hrd_parameters() | |
| if(hrd_parameters_present_flag) | |
| low_delay_hrd_flag | u(1) |
| max_dec_frame_buffering | ue(v) |
| } | |

D.1.2 假设参考解码器(HRD)参数语法

HRD 参数语法见表 D.2。

表 D.2 HRD 参数语法表

| hrd_parameters() | 描述符 |
|--|-------|
| cpb_cnt_minus1 | ue(v) |
| bit_rate_scale | u(4) |
| cpb_size_scale | u(4) |
| for(SchedSelIdx = 0; SchedSelIdx <= cpb_cnt_minus1; SchedSelIdx++) { | |
| bit_rate_value_minus1[SchedSelIdx] | ue(v) |
| cpb_size_value_minus1[SchedSelIdx] | ue(v) |
| cbr_flag[SchedSelIdx] | u(1) |
| } | |
| initial_cpb_removal_delay_length_minus1 | u(5) |
| pb_removal_delay_length_minus1 | u(5) |
| dpb_output_delay_length_minus1 | u(5) |
| } | |

D.2 视频可用性信息(VUI)语义

D.2.1 视频可用性信息(VUI)参数语义

timing_info_present_flag 等于 1 表示 num_units_in_tick, time_scale 和 fixed_frame_rate_flag 在码流中存在, timing_info_present_flag 等于 0 表示不存在上述参数。

num_units_in_tick 表示在频率为 time_scale Hz 的 clock tick 计数器的时间单元的个数。该值应大于 0,一个时钟 tick 为编码数据表示的最小时时间间隔。例如,如果视频信号的频率为(30 000 ÷ 1001) Hz, time_scale 可以是 30 000, num_units_in_tick 可以是 1001。

time_scale 表示一秒钟内时间单元的总数。该值应大于 0。例如,一个时钟频率为 27MHz 的定时系统的 time_scale 应为 27 000 000。

fixed_frame_rate_flag 等于 1, 表示连续图像的 HRD 输出时间的时域间隔是固定的, 该间隔为 num_units_in_tick ÷ time_scale。fixed_frame_rate_flag 等于 0 则表示无此限制。

hrd_parameters_present_flag 等于 1 表示码流中后面紧跟 HRD 参数。否则表示后续码流中不存在 HRD 参数。另外,两个临时变量 HrdBpPresentFlag(用于缓存周期 SEI 消息)和 CpbDpbDelaysPresentFlag(用于图像定时 SEI 消息)的取值应与 hrd_parameters_present_flag 相同。

low_delay_hrd_flag 定义了 HRD 的一种操作模式。当 fixed_frame_rate_flag 等于 1 时, low_delay_hrd_flag 应等于 0。

max_dec_frame_buffering 定义了 HRD DPB 的帧缓存的最大个数。其取值范围从 0 到 MaxDPB, 如果码流中没有该语法元素, 则为 MaxDPB(见附录 C)。

D.2.2 假设参考解码器(HRD)参数语义

cpb_cnt_minus1 加 1 定义了可供选择的 CPB 参数的个数。cpb_cnt_minus1 的取值范围为 0~31(包括 0 和 31)。如果 low_delay_hrd_flag 等于 1, cpb_cnt_minus1 应等于 0。如果码流中没有 cpb_cnt_minus1, 则默认其值等于 0。

bit_rate_scale, bit_rate_value_minus1[SchedSelIdx] 两个参数一起定义了第 SchedSelIdx 个 CPB 的最大输入码率。bit_rate_value_minus1[SchedSelIdx] 的取值范围 $0 \sim (2^{32} - 2)$ (包括 0 和 $2^{32} - 2$), 并且随索引号 SchedSelIdx 的变大而变大。码率公式为:

$$\text{BitRate}[SchedSelIdx] = (\text{bit_rate_value_minus1}[SchedSelIdx] + 1) \times 2^{(6 + \text{bit_rate_scale})}$$

如果 bit_rate_value_minus1[SchedSelIdx] 不存在, 则默认其值等于 $1\ 200 \times \text{MaxBR}$ 。

cpb_size_scale, cpb_size_value_minus1[SchedSelIdx] 两个参数一起定义了第 SchedSelIdx 个 CPB 的 CPB 的大小。cpb_size_value_minus1[SchedSelIdx] 的取值范围 $0 \sim (2^{32} - 2)$ (包括 0 和 $2^{32} - 2$), 并且随索引号 SchedSelIdx 的变大而变小。CPB 大小公式为:

$$\text{CpbSize}[SchedSelIdx] = (\text{cpb_size_value_minus1}[SchedSelIdx] + 1) \times 2^{(4 + \text{cpb_size_scale})}$$

如果 cpb_size_value_minus1[SchedSelIdx] 不存在, 则默认其值等于 $1\ 200 \times \text{MaxCPB}$ 。

cbr_flag[SchedSelIdx] 定义了第 SchedSelIdx 个 CPB 参数定义的 HRD 工作模式。如果其值等于 0, 则为可变比特率模式(VBR), 如果其值等于 1, 则为恒定比特率模式(CBR)。如果码流中不存在 cbr_flag[SchedSelIdx], 则默认其值等于 0。

initial_cpb_removal_delay_length_minus1 加 1 等于缓存周期 SEI 消息中的语法元素 initial_cpb_removal_delay[SchedSelIdx] 的码字长度。如果码流中不存在 initial_cpb_removal_delay_length_minus1, 则默认其值等于 23。

cpb_removal_delay_length_minus1 加 1 等于图像定时 SEI 消息中的语法元素 cpb_removal_delay 的码字长度。如果码流中不存在 cpb_removal_delay_length_minus1, 则默认其值等于 23。

dpb_output_delay_length_minus1 加 1 等于图像定时 SEI 消息中的语法元素 dpb_output_delay 的码字长度。如果码流中不存在 dpb_output_delay_length_minus1, 则默认其值等于 23。