Energy Preservation in PVQ-Based Video Coding

Jean-Marc Valin

February 20, 2012

1 Introduction

This mini-paper describes a proposal for adapting the CELT energy conservation principle to video coding based on a pyramid vector quantizer (PVQ). One potential advantage of conserving energy of the AC coefficients in video coding is preserving textures rather than low-passing them. Also, by introducing a fixed-resolution PVQ-type quantizer, we automatically gain a simple activity masking model.

The main challenge of adapting this scheme to video is that we have a good prediction (the reference frame), so we are essentially starting from a point that is already on the PVQ hyper-sphere, rather than at the origin like in CELT. Other challenges are the introduction of a quantization matrix and the fact that we want the reference (motion predicted) data to perfectly correspond to one of the entries in out codebook.

2 Encoder

Let vector $\mathbf{x}_d$ denote the (pre-normalization) DCT band to be coded in the current block and vector $\mathbf{r}_d$ denote the corresponding reference after motion compensation, the encoder computes and encodes the "band gain"

$$g = \sqrt{\mathbf{x}_d^T \mathbf{x}_d} \,. \tag{1}$$

Let $\mathbf{Q}$ be a diagonal matrix containing the quantization step size for each element of $\mathbf{x}_d$, the normalized band is computed as

$$\mathbf{x} = \frac{\mathbf{Q}^{-1}\mathbf{x}_d}{\left\|\mathbf{Q}^{-1}\mathbf{x}_d\right\|} \,, \tag{2}$$

with the normalized reference $\mathbf{r}$ similarly computed based on $\mathbf{r}_d$. The encoder then finds the position and sign of the maximum value in $\mathbf{r}$

$$m = \underset{i}{\operatorname{argmax}} \left|r_i\right| \tag{3}$$

$$s = \operatorname{sgn}\left(r_m\right) \tag{4}$$

and computes the Householder reflection that reflects $\mathbf{r}$ to $-s\mathbf{e}_m$. The reflection vector is given by

$$\mathbf{v} = \frac{\mathbf{r} + s\mathbf{e}_m}{\|\mathbf{r} + s\mathbf{e}_m\|} . \tag{5}$$

The encoder reflects the normalized band to find

$$\mathbf{x}_r = \mathbf{x} - 2\mathbf{v}\left(\mathbf{v}^T\mathbf{x}\right) . \tag{6}$$

Let N be the number of dimensions in $\mathbf{x}$ and K be the number of pulses in our codebooks, we search for the codebook entry

$$q = \underset{i}{\operatorname{argmax}} \frac{\mathbf{p}_i^T\mathbf{x}_r}{\sqrt{\mathbf{p}_i^T\mathbf{p}_i}} , \tag{7}$$

where $\mathbf{p}_i$ is the i^{th} combination of magnitudes and signs that satisfies $\|\mathbf{p}_i\|_{L1} = K$.

The encoder first transmits the magnitude and sign of any pulse at position m. Assuming that prediction from the reference frame is good, the magnitude should be close to K and the sign should be $-s$, so entropy coding should be highly efficient for these values. The remaining values can be coded as a PVQ vector of dimension $N-1$. It would seem like the best approach for coding $\mathbf{p}_q$ would be from SILK-like recursive splitting of the vector (with entropy coding), but this is made more difficult by the fact that we subtracted one dimension. One possible work around is to leave the extra dimension as a zero (or leave a few bits from the magnitude encoding and code those on the second step).

By using a fixed K, the codec will roughly optimise for SSIM in that each block will have a near-constant SNR (not PSNR). This may be a bit too extreme, so we probably want to allow some variation on K. In fact, some of that variation should probably be automatic, and based on the gain value g for the band. The higher the gain, the larger K should be, though K should probably not grow linearly as a function of g.

3 Decoder

The decoder starts by decoding the codebook entry $\mathbf{p}_q$ and uses it to reconstruct the unit-norm reflected band as

$$\tilde{\mathbf{x}}_r = \frac{\mathbf{p}_q}{\sqrt{\mathbf{p}_q^T\mathbf{p}_q}} . \tag{8}$$

Because the decoder has access to exactly the same reference as the encoder, it is able to apply (3)-(5) to obtain the same $\mathbf{v}$ as used in the encoder. The decoded normalized band is

$$\tilde{\mathbf{x}} = \tilde{\mathbf{x}}_r - 2\mathbf{v}\left(\mathbf{v}^T\tilde{\mathbf{x}}_r\right) . \tag{9}$$

The renormalized band is computed by taking into account the quantization resolution:

$$\tilde{\mathbf{x}}_d = g\frac{\mathbf{Q}\tilde{\mathbf{x}}}{\|\mathbf{Q}\tilde{\mathbf{x}}\|} . \qquad (10)$$

4 Bit Allocation and SSIM

According to Wikipedia, the SSIM metric is defined as

$$\mathrm{SSIM}\left(x,y\right) = \left(\frac{\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}\right)\cdot\left(\frac{\sigma_{xy} + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}\right) .$$

Where μ_x and μ_y are the DC of images x and y and σ_x and σ_y are the RMS value of the AC coefficients of images x and y. From now on, we will consider x to be the reference image and y to be the coded image. Now, let's ignore the DC for now and define a Simplified SSIM metric as

$$\mathrm{SSSIM}\left(x,y\right) = \frac{2\sigma_{xy} + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} .$$

This is the metric we'll try optimizing here. First, let $g = \sigma_y/\sigma_x$ be gain that the codec causes on the AC coefficients and $\hat{y} = y/g$. Solving for

$$\frac{d}{dg}\mathrm{SSSIM}\left(x,y\right) = \frac{d}{dg}\frac{2g\sigma_{x\hat{y}} + c_2}{\sigma_x^2\left(1+g^2\right) + c_2} = 0$$

we find that the optimal gain that maximizes SSSIM is $g_{max} \approx 1 - \frac{c_2}{2\sigma_x^2}\cdot\left(\frac{\sigma_x^2}{\sigma_{x\hat{y}}} - 1\right)$. This means that conserving energy ($g_{max} = 1$) is a good thing to do as long as the contrat is high enough ($\frac{c_2}{2\sigma_x^2}$ is small) or the bit-rate is high enough ($\frac{\sigma_x^2}{\sigma_{x\hat{y}}}$ close to 1).

Now, let's consider a spherical horse in simple harmonic motion... or to be more exact, let's consider that the PVQ codebook is perfectly uniform over the sphere and that $g_{max} = 1$. We get

$$\mathrm{SSSIM}\left(x,y\right) = \frac{\sigma_{xy} + c_2/2}{\sigma_x^2 + c_2/2} ,$$

where $\sigma_{xy}/\sigma_x^2 = \cos\theta$ is the cosine distance between x and y. Assuming a uniform quantizer, we have

$$\theta \propto 2^{-b/(N-1)} ,$$

where b is the number of bits allocated and N is the number of AC coefficients. Let $c' = c/(2\sigma_x^2)$...

<FIXME: This needs to be cleaned up>

$$\mathrm{SSSIM}\left(x,y\right) = \frac{\cos\theta + c'}{1 + c'} \approx \frac{1 - \theta^2 + c'}{1 + c'} ,$$

Trying to make SSIM equal for two blocks:

$$\frac{1 + c'_1 - 2^{-2b_1/(N-1)}}{1 + c'_1} = \frac{1 + c'_2 - 2^{-2b_2/(N-1)}}{1 + c'_2}$$

The optimal bit offset is

$$b = -\frac{N-1}{2} \log_2 \left(1 + 2c/\sigma_x^2\right)$$

From this (theoretically) optimal offset, we can encode only the deviation from the optimal allocation. In practice, b would not be an exact bit allocation like for CELT, but only the "quantization step exponent".

5 Conclusion

While it seems like a good idea, none of this has actually been tested yet.