Predicting Chroma from Luma using Frequency Domain Intra Prediction in Codecs Based on Lapped Transforms

Nathan E. Egge
Jean-Marc Valin
Intra-Prediction of Chroma

- In 4:2:0 image data, chroma is 50% of luma
- Chroma predicted spatially by signalling a directional mode
  - Reconstructed neighbors must be available to decode a block
  - Limited to predicting from current color plane
- Cross-channel correlation not exploited
- Does not work with codecs using lapped transforms
The intra-prediction modes for 4x4 blocks in WebM (VP8).
Lapped Transforms

prefilter

Mozilla & The Xiph.Org Foundation
Decoding an Intra Frame with Lapped Transforms

Neighboring blocks:

- **Reconstructed Image**
- **Predicted**
- **Unpredicted**
- **Currently Predicting**
- **Needs Post-filter**
- **Prediction Support**
Predicting Chroma from Luma

- Key insight: YUV conversion de-correlates luma and chroma globally, but local relationship exists [1]

- Both encoder and decoder compute linear regression:
  \[
  \alpha = \frac{N \cdot \sum L_i \cdot C_i - \sum L_i \sum C_i}{N \cdot \sum L_i \cdot L_i - \left( \sum C_i \right)^2}
  \]
  \[
  \beta = \frac{\sum C_i - \alpha \cdot \sum L_i}{N}
  \]

- Use reconstructed luma coefficients to predict coincident chroma coefficients:
  \[
  C(u, v) = \alpha \cdot L(u, v) + \beta
  \]

- Not selected for HEVC due to 20-30% increased complexity


Mozilla & The Xiph.Org Foundation
Adapting Chroma from Luma to the Frequency Domain

- Key insight: LT and DCT are both linear transforms so similar relationship exists in frequency domain
- Both encoder and decoder compute linear regression using 4 LF coefficients from Up, Left and Up-Left
- Use reconstructed luma coefficients to predict coincident chroma coefficients:
  \[
  C_{DC} = \alpha_{DC} \cdot L_{DC} + \beta_{DC} \\
  C_{AC}(u, v) = \alpha_{AC} \cdot L_{AC}(u, v)
  \]

<table>
<thead>
<tr>
<th>Block Size</th>
<th>SD-CfL</th>
<th>FD-CfL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adds</td>
<td>Mults</td>
</tr>
<tr>
<td>N x N</td>
<td>4*N+2</td>
<td>8*N+3</td>
</tr>
<tr>
<td>4 x 4</td>
<td>18</td>
<td>35</td>
</tr>
<tr>
<td>8 x 8</td>
<td>34</td>
<td>67</td>
</tr>
<tr>
<td>16 x 16</td>
<td>66</td>
<td>131</td>
</tr>
</tbody>
</table>

Still expensive, but cost constant with block size
Example

Original uncompressed image
Example

Reconstructed luma with predicted chroma using FD-CfL
Frequency Domain CfL

- Adapted CfL algorithm to the frequency domain
  - No signalling overhead
    - Implicitly defined model parameters \((\alpha_{DC}, \beta_{DC}, \alpha_{AC})\)
    - Increased decoder complexity
  - Model parameters could be signalled for use cases
  - Works with existing LT based codecs using scalar quantization
Perceptual Vector Quantization

- Separate “gain” (contrast) from “shape” (spectrum)
  - Vector = Magnitude $\times$ Unit Vector (point on sphere)
- Given prediction vector $\mathbf{r}$
  - “gain” predicted by magnitude
    $$\hat{g} = \gamma_g \cdot Q + \|\mathbf{r}\|$$
  - “shape” predicted using Householder reflection
    $$\mathbf{v} = \frac{\mathbf{r}}{\|\mathbf{r}\|} + s \cdot \mathbf{e}_m$$
    $$\mathbf{z} = \mathbf{x} - 2 \frac{\mathbf{v}^T \mathbf{x}}{\mathbf{v}^T \mathbf{v}} \mathbf{v}$$
Shape Prediction Example

- Input + Prediction
Shape Prediction Example

- **Input + Prediction**
- **Compute Householder Reflection**
- **Apply Reflection**
- **Compute & code angle**
- **Code other dimensions**
PVQ Prediction with CfL

- Consider prediction of 15 AC coefficients of 4x4 Cb
- The 15-dimensional predictor $r$ is scalar multiple of coincident reconstructed luma coefficients $\hat{x}_L$

$$C_{AC}(u, v) = \alpha_{AC} \cdot L_{AC}(u, v) \implies r = \alpha_{AC} \cdot \hat{x}_L$$

- Thus “shape” predictor is almost exactly $\hat{x}_L$

$$\frac{r}{\|r\|} = \frac{\alpha_{AC} \cdot \hat{x}_L}{\|\alpha_{AC} \cdot \hat{x}_L\|} = \text{sgn}(\alpha_{AC}) \frac{\hat{x}_L}{\|\hat{x}_L\|}$$

- Only difference is direction of correlation!
PVQ Chroma from Luma

1: Let \( r = \hat{x}_L \), compute \( \theta \)
2: If \( \theta = 0 \) prediction is exact, code \( \theta \)
3: Else
4: Code a *flip* flag, \( f = \theta > 90^\circ \)
5: If \( f \), let \( r = -\hat{x}_L \)
6: Code \( x_C \) with PVQ using predictor \( r \)
7: End
Still Image Experiment

- Sample of 50 high resolution still images taken from Wikipedia down-sampled to 1 megapixel
- Comparison of No-CfL, FD-CfL and PVQ-CfL
  - Encode with 28 different quantization levels
  - Compute rate/distortion on Cb and Cr planes using four metrics: PSNR, PSNR-HVS, SSIM, FastSSIM
  - Hold all other techniques constant
Still Image Experiment

![Graph showing quality vs rate for different audio coding schemes.](image-url)
Still Image Experiment Cont.

Computation of the Bjontegaard distance (improvement) between two rate-distortion curves

<table>
<thead>
<tr>
<th>Metric</th>
<th>Cb (plane 1)</th>
<th>Cr (plane 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Δ Rate (%)</td>
<td>Δ SNR (dB)</td>
</tr>
<tr>
<td>PSNR</td>
<td>-1.87644</td>
<td>0.07678</td>
</tr>
<tr>
<td>PSNR-HVS</td>
<td>-2.57971</td>
<td>0.13205</td>
</tr>
<tr>
<td>SSIM</td>
<td>-3.09834</td>
<td>0.08842</td>
</tr>
<tr>
<td>FastSSIM</td>
<td>-3.01455</td>
<td>0.06602</td>
</tr>
</tbody>
</table>

Improvement moving from No-CfL to FD-CfL

<table>
<thead>
<tr>
<th>Metric</th>
<th>Cb (plane 1)</th>
<th>Cr (plane 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Δ Rate (%)</td>
<td>Δ SNR (dB)</td>
</tr>
<tr>
<td>PSNR</td>
<td>-3.13262</td>
<td>0.12853</td>
</tr>
<tr>
<td>PSNR-HVS</td>
<td>-5.19186</td>
<td>0.26913</td>
</tr>
<tr>
<td>SSIM</td>
<td>-5.54403</td>
<td>0.15962</td>
</tr>
<tr>
<td>FastSSIM</td>
<td>-6.10963</td>
<td>0.13577</td>
</tr>
</tbody>
</table>

Improvement moving from No-CfL to PVQ-CfL
Conclusions & Future Work

- Introduced 2 algorithms for Chroma-from-Luma intra prediction in codecs using LT
  - FD-CfL suitable for use with scalar quantization
  - PVQ-CfL extends gain-shape quantization
    - No additional per block complexity
    - Improved performance (both rate and quality)

- Can we use both reconstructed Luma and Cb with PVQ to predict Cr?
Resources

- Daala codec website: https://xiph.org/daala/
- Daala Technology Demos: https://people.xiph.org/~xiphmont/demo/daala/
- Git repository: https://git.xiph.org/
- IRC: #daala channel on irc.freenode.net
- Mailing list: daala@xiph.org
Questions?