Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 18, 2013

N. Egge
T. Terriberry
Mozilla Corporation

October 15, 2012

Time Domain Lapped Transforms for Video Coding
draft-egge-videocodec-tdlt-00

Abstract

This proposes the use of Time Domain Lapped Transforms (TDLT) as the transform step for video coding.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

This draft outlines a proposal to adapt the Time-Domain Lapped Transforms (TDLT) for use in video coding. Lapped transforms were proposed for video coding at least as as far back as 1989 [Malv89]. Like the loop filters more commonly found in recent video coding standards, TDLTs use a post-processing filter that runs between block edges to reduce or eliminate blocking artifacts. Unlike a loop filter, the TDLT filter is invertible, allowing the encoder to run the inverse filter on the input video. This decorrelates blocks before they are passed through a normal block transform and quantization step, improving coding gain (which helps in both smooth and highly textured areas), in addition to reducing blocking artifacts.

2. TDLT Defined

The Time-Domain Lapped Transform can be viewed as a set of pre and post filters to an existing block-based DCT transform. The idea is to place an invertible filter along the block boundaries outside an existing block-based DCT encoder.

```
+-----+                 +-----+
|     |        Q        |     |
| P   |+-----+  u  +-----+|P^-1|
|     ||     |  a  |     ||     |
+-----+|DCT |  n  |iDCT|+-----+
|     ||     |  t  |     ||     |
| P   |+-----+  i  +-----+|P^-1|
|     ||     |  z  |     ||     |
+-----+|DCT |  a  |iDCT|+-----+
|     ||     |  t  |     ||     |
| P   |+-----+  i  +-----+|P^-1|
|     |        o        |     |
+-----+        n        +-----+
```

The pre-filter P operates in the time domain, processing block boundaries and removing inter-block correlation. The blocks are then transformed by the DCT into the frequency domain, where the resulting coefficients are quantized and encoded. When decoding, the inverse operator P^-1 is applied as a post-filter to the output of the inverse DCT. This has two benefits:

1. Quantization errors are spread over adjacent blocks via the post-filter P^-1, reducing blocking artifacts. This eliminates the need for a separate deblocking filter.

2. The increased support region of the transform allows it to take advantage of inter-block correlation to achieve a higher coding gain than a non-overlapped DCT. This allows it to more effectively code both smooth and textured regions.

The pre-filter P is defined in [Tran01] as follows:

$$P = \frac{1}{2} \begin{bmatrix} I & J \\ J & -I \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & V \end{bmatrix} \begin{bmatrix} I & J \\ J & -I \end{bmatrix}$$

Here I is the identity matrix and J is the "reversal matrix", obtained by simply re-ordering the rows of the identity matrix in reverse order. The V matrix is a free parameter, and as long as V is invertible, this filter structure guarantees perfect reconstruction, linear phase, and biorthogonality. If V is orthogonal, then the overall transform is also orthogonal instead of just biorthogonal.

For the case of the 4x8 TDLT, we use the following invertible matrix for V:

$$V = \begin{bmatrix} 1 & q\_0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ p\_0 & 1 \end{bmatrix} \begin{bmatrix} s\_0 & 0 \\ 0 & s\_1 \end{bmatrix}$$

Thus for the 4x8 case, the pre-filter and post-filter are completely described by the four parameters q_0, p_0, s_0, and s_1. In general, any invertible V matrix may be used. However, factoring V into a series of lifting steps ensures that it can be implemented efficiently, and can reduce the number of parameters required by the optimization process, since the full flexibility of an arbitrary invertible matrix is not required to achieve good coding gain. [Tran01] proposes two reduced-parameter factorizations, dubbed Type III and Type IV. These are identical in the 4x8 case, but for larger transforms the differ in the order that the p_i and q_i steps are applied: interleaved for Type III and ascending and then descending order for Type IV. While Type III appears to give slightly higher coding gain when unconstrained, when coupled with the ramp constraint discussed below and the constraint that all coefficients be dyadic rationals, the number of feasible solutions is much smaller than with Type IV. The increased number of feasible solutions allows Type IV transforms to achieve higher coding gains than Type III when these constraints are imposed. This definition easily extends to the 8x16 and 16x32 TDLT case with similar parameterizations. In general, we use the Type IV factorization from [Tran01]. For a V matrix of size M, this has (M-1) p_i and (M-1) q_i parameters, and M s_i parameters. For a transform of size Nx2N, this gives a total of 1.5N-2 parameters. This is also the

number of lifting steps that must be performed to implement the V portion of the pre- and post-filters.

3. Lapped-Transform Selection

We would like to find good candidate transform coefficients that perform well within a video coding framework. There are several metrics we can use for evaluating pre-filter parameters. Including

1. Coding Gain - how well energy is compacted into only a few coefficients
2. Side Band Attenuation - how much energy from frequencies outside the passband leaks into each basis function
3. Transform Width - how wide are the basis functions and how much ringing they will cause
4. Orthogonality - how linearly independent the basis functions are

Of these, the most important by far is coding gain as it allows us to directly measure the improvement in bits between different candidate transforms. At high bit rates using an efficient quantizer, every 6.02 dB improvement in coding gain saves a bit of entropy per coefficient.

3.1. Coding Gain

Coding gain is a useful metric for comparing different candidate transforms. Roughly speaking, it is the measure of how well energy is compacted into only a few coefficients. The formula for coding gain of the lapped transform can be found in [Terr12]. Using an AR(1) model with r=0.95, we have

```
                 /                      1                    \
C_g = 10*Log_10 | ----------------------------------------- |
                 \ Prod_i((G*AR(1)*G^T)[i,i]*(H^T*H)[i,i]) /
```

where G is the analysis filter of the lapped transform:

```
        [         ][ P  0 ]
    G = [ 0 DCT 0 ][      ]
        [         ][ 0  P ]
```

and H is the synthesis filter of the lapped transform:

```
        [ P^-1  0   ][  0   ]
    H = [           ][ iDCT ]
        [  0   P^-1 ][  0   ]
```

In [Terr12] the coding gain of the non-lapped DCT is compared with the optimal non-lapped Karhunen-Loeve transform for the same AR(1) model with r=0.95.

	4 point	8 point	16 point
DCT	7.5701 dB	8.8259 dB	9.4555 dB
KLT	7.5825 dB	8.8462 dB	9.4781 dB

Similarly, in [Tran01] the coding gain of the TDLT using fast factorizations with real coefficients produced by unconstrained optimization are

	4x8	8x16	16x32
Type III TDLT	8.6349 dB	9.6115 dB	9.9496 dB
Type IV TDLT	8.6349 dB	9.6005 dB	9.9057 dB

3.2. Transform Width

In general, the wider the transform, the higher the coding gain: a 16-point DCT will always have a higher coding gain than a 4-point DCT. In the case of lapped transform, the width of the transform is more than just counting the number of points, it involves the shape of the basis functions. At equal coding gain, a narrower transform is better because it causes a smaller amount of ringing around edges. We define the width of the transform as

$$w = C \cdot \left(\frac{\sum_{ij} \left(H[i,j]^2 \cdot (j-N+1/2)^4 \right)}{\sum_{ij} \left(H[i,j]^2 \right)} \right)^{1/4},$$

where C=2.991 is a constant calibrated such that the width of the 1024-point non-overlapped DCT is equal to 1024.

4. Optimal Transform Coefficients

Of the four metrics described in Section 3 we chose to optimize our transform parameters for the highest coding gain.

To avoid the use of floating point operations, we use dyadic rationals to represent the parameters of our TDLT. These are the p's, q's and s's that describe the V matrix in the pre-filter. We chose a base of 2^6 because it offered enough resolution to find good

approximations of the optimal values for the p's, q's, and s's and still allowed us to fit the results of multiplications in a 16 bit word. Increasing the base to 2^8 improves the achievable coding gain of the 4x8 transform by less than 0.002 dB. On the other hand, dropping it even one bit to 2^5 lowers the coding gain by 0.037 dB.

4.1. Exhaustive Search

For the smaller lapped transforms, it is possible to simply do an exhaustive search and check all possible transform candidates to find the one with the best coding gain. The limitation that the p's, q's, and s's all be dyadic rationals allows us to simply enumerate all reasonable values. Additional constraints allowed us to further reduce the search space. Because the p's and q's are liftings steps that represent rotations in the plane their, values are between -1.0 and 1.0. Likewise the limitation that the pre- and post-filter steps be reversible requires that the scale factors be &gte; 1.0, otherwise information would be lost during the transform. Finally, all things equal we prefer smaller scale factors as it makes quantizing and encoding the coefficients cheaper. We thus cap the scale factors at 2.0. Based on some limited experimentation, scale factors larger than this do not appear to produce useful transforms according to our metrics, anyway.

With a dyadic rational base of 2^6, the number of possible candidates to consider is

```
|C| = (2*(2^6)+1)^(|p|+|q|)   *  (2^6+1)^|s|
    = (2*(2^6)+1)^(2*(N/2-1)) *  (2^6+1)^(N/2)
```

Thus for the transform sizes we are interested in, the number of candidates is tractable only for the 4x8 case:

	N	|C|
4x8 TDLT	4	68161536
8x16 TDLT	8	7.731400 * 10^19
16x32 TDLT	16	9.947082 * 10^43

An exhaustive search for parameters that give the optimal coding gain for the 4x8 TDLT are below:

p_0	-11/64

q_0	36/64

s_0	91/64
s_1	85/64

4.2. Stochastic Search

For the larger lapped transforms, doing an exhaustive search is not possible. Instead we formulate the optimization problem as an integer programming problem and use a robust industrial solver to find optimal integer values for the p's, q's, and s's.

For the 8x16 TDLT, the parameters are below:

p_0	-23/64
p_1	-18/64
p_2	-6/64

q_0	48/64
q_1	34/64
q_2	20/64

s_0	90/64
s_1	73/64
s_2	72/64
s_3	75/64

For the 16x32 TDLT, the parameters are below:

p_0	-24/64
p_1	-23/64
p_2	-17/64
p_3	-12/64
p_4	-14/64
p_5	-13/64
p_6	-7/64

q_0	50/64
q_1	40/64
q_2	31/64
q_3	22/64
q_4	18/64
q_5	16/64
q_6	11/64

s_0	90/64
s_1	74/64
s_2	73/64
s_3	71/64
s_4	67/64
s_5	67/64
s_6	67/64
s_7	72/64

In order to confirm that the integer approximations found are in fact optimal, we can compare them with the optimal real valued coding gains for the three lapped-transforms we are proposing. In [Tran01] a numeric solver was used to find optimal values for a Type IV lapped transform.

	4x8	8x16	16x32
Real Valued	8.6349 dB	9.6005 dB	9.9057 dB
Approximate	8.63473 dB	9.60021 dB	9.89338 dB
Loss	0.00017 dB	0.00029 dB	0.01232 dB

4.3. Ramp Constraint

It is also possible to constrain the lapped transform so that it is (1,2)-regular [DT03], i.e., that it has one vanishing moment in the analysis filter and two vanishing moments in the synthesis filter.

This allows the synthesis filter to reconstruct any piecewise linear function solely from the DC coefficients. This causes the shape of the DC basis function to be a symmetric linear ramp. This can be particularly useful when it matches the shape of other windowing functions used in the codec. For example, a linear window is commonly used with Overlapped Block Motion Compensation (OBMC), which is one possible approach for avoiding blocking artifacts in the motion-compensation stage of the codec. More vanishing moments are possible, allowing reconstruction of piecewise quadratic or even higher-order functions, but these require additional overlap stages.

This regularity can be enforced solely by enforcing a series of constraints on the scale factors, s_i.

```
     s_0 = N*(1 - q_0)

              N       /                             \
     s_i = ------- * | (q_{i-1} - 1)*p_{i-1} - q_i | , for i > 0
           2*i + 1    \                             /
```

Since 2*i + 1 is odd, but we want s_i to be a dyadic rational value, the remainder of the expression must be evenly divisible by (2*i+1). A similar set of constraints can be derived for Type III, but they involve more of the p's and q's per s_i value, and thus have far fewer admissible solutions when coupled with the dyadic rational constraint.

The additional restrictions described above greatly reduce the number of combinations to consider, both because there are fewer parameters (the s_i's can no longer be chosen independently) and because there are fewer combinations of parameter values which produce dyadic rational coefficients. With these constraints, the number of combinations is small enough that an exhaustive search is now tractable for the 8x16 TDLT.

	N	|C|
4x8 TDLT	4	442
8x16 TDLT	8	331677320

An exhaustive search for parameters that give the optimal coding gain under the ramp and dyadic rational constraints for the 4x8 and 8x16 TDLT are below:

p_0	-16/64

q_0	41/64

s_0	92/64
s_1	93/64

```
+-----+--------+
```

p_0	-24/64
p_1	-20/64
p_2	-4/64

q_0	53/64
q_1	40/64
q_2	24/64

s_0	88/64
s_1	75/64
s_2	76/64
s_3	76/64

Unfortunately, in the 16x32 TDLT case the number of combinations is still not tractable, even with these additional constraints. Again, we use an integer programming model to solve for the integer parameters that optimize coding gain in this context.

p_0	-32/64
p_1	-28/64
p_2	-24/64
p_3	-32/64
p_4	-24/64
p_5	-13/64
p_6	-2/64

q_0	59/64
q_1	53/64
q_2	46/64
q_3	41/64
q_4	35/64
q_5	24/64
q_6	12/64

s_0	80/64
s_1	72/64
s_2	73/64
s_3	68/64
s_4	72/64
s_5	74/64
s_6	74/64
s_7	70/64

	4x8	8x16	16x32
Dyadic	8.63473 dB	9.60021 dB	9.89338 dB
Ramp + Dyadic	8.59886 dB	9.56161 dB	9.78294 dB
Loss	0.03587 dB	0.0386 dB	0.11044 dB

5. Future Work

There are a number of challenges that must be addressed to incorporate TDLTs into a codec.

1. Intra Prediction. Since the final pixel values of a block are not available until after the post-filter runs, they cannot be used to predict neighboring blocks. There are a number of possible solutions to this. For example, one could simply use pixels from outside the overlap region. However, as these pixels are farther away, they are poorer predictors, and the extra distance reduces the range of prediction directions which have enough neighbors available to form an adequate

extrapolation [OP11]. An alternate approach is to perform the prediction in the frequency domain. Initial experiments suggest that this is just as effective as prediction in the time domain, and has similar computational requirements.

2. Motion Compensation. There have been several lapped transform proposals that perform block-by-block motion compensation by simply expanding the size of the prediction region for each block [TT01], [OPT11]. However, in addition to increasing the amount of motion-compensated prediction pixels that must be computed by a factor of four, this also increases the number of applications of the pre- and post-filter by a factor of four, since this must now be done separately for each block, using the motion-compensated frame difference for that block. An alternate approach is simply perform motion compensation of the frame in a completely separate step, prior to any transform, using any method desired. The lapping can then be applied to this motion-compensated prediction, producing per-block predictors. This still allows the prediction mode (inter, intra, bi-prediction, etc.) to be chosen on a block-by-block basis. It also interacts well with other techniques designed to operate in the frequency domain, such as the Pyramid Vector Quantization (PVQ) proposed elsewhere. The downside is that motion estimation in the encoder needs to be performed for regions slightly beyond the current block. However, this is already required by blocking-artifact-free motion compensation techniques, such as Overlapped Block Motion Compensation (OBMC). Experience with OBMC has shown that an encoder can mostly ignore look-ahead and still get acceptable results, unlike other techniques, such as control-grid interpolation (CGI).

3. Multiple Block Sizes. Multiple block size support is important for lapped transforms, since the larger support region increases their susceptibility to ringing artifacts compared to a non-overlapped transform with the same number of coefficients (though it is greatly reduced compared to a non-overlapped transform with a support region of the same size). The simplest approach is to require that the size of the overlap filter be constrained by the smallest block adjacent to a given edge. This also requires some amount of look-ahead in the encoder, but in addition has an effect on the coding syntax, since this constraint must be communicated to the decoder in order for it to perform motion compensation. It is straightforward to make this information non-redundant, i.e., to use it to constraint the actual block sizes of future blocks when they are eventually decoded.

6. IANA Considerations

This document has no actions for IANA.

7. Security Considerations

This draft has no security considerations.

8. Acknowledgments

Thanks to Greg Maxwell and Jean-Marc Valin for their assistance in the experimentation and other valuable contributions to this document.

9. Informative References

[DT03] Dai, W. and T. Tran, "Regularity-Constrained Pre- and Post-Filtering for Block DCT Based Systems", IEEE Transactions on Signal Processing 51(10):2568--2581, October 2003.

[Malv89] Malvar, H. and D. Staelin, "The LOT: Transform Coding Without Blocking Effects", IEEE Transactions on Acoustics, Speech, and Signal Processing , April 1989, <http://research.microsoft.com/apps/pubs/default.aspx?id=102073>.

[OP11] de Oliveria, R. and B. Pesquet-Popescu, "Intra-Frame Prediction with Lapped Transforms for Image Coding", Proc. of the 36th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'11) pp. 805--808, May 2011.

[OPT11] de Oliveria, R., Pesquet-Popescu, B., and M. Trocan, "Inter Prediction Using Lapped Transforms for Advanced Video Coding", Proc. of the 18th IEEE International Conference on Image Processing (ICIP'11) pp. 3705--3708, September 2011.

[Tran01] Tran, T., "Lapped Transform via Time-Domain Pre- and Post-Processing", IEEE Transactions on Signal Processing , October 2001.

[TT01] Tran, T. and C. Tu, "Lapped Transform Based Video Coding", Proc. of the 24th SPIE Conference on Applications of Digital Image Processing vol. 4472, pp. 319--333, July 2001.

[Terr12] Terriberry, T., "Introduction to Video Coding Part 1: Transform Coding", February 2012.

Authors' Addresses

Nathan E. Egge
Mozilla Corporation
650 Castro Street
Mountain View, CA 94041
USA

Email: negge@dgql.org

Timothy B. Terriberry
Mozilla Corporation
650 Castro Street
Mountain View, CA 94041
USA

Phone: +1 650 903-0800
Email: tterribe@xiph.org