Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 11, 2017

S. Midtskogen
A. Fuldseth
M. Zanaty
Cisco

March 10, 2017

Constrained Low Pass Filter
draft-midtskogen-netvc-clpf-04

Abstract

This document describes a low complexity filtering technique which is being used as a low pass loop filter in the Thor video codec.

Table of Contents

1. Introduction 2
2. Definitions 2
2.1. Requirements Language 2
2.2. Terminology 2
3. Filtering Process 3
4. Further complexity considerations 6
5. Performance 6
6. IANA Considerations 7
7. Security Considerations 7
8. Acknowledgements 7
9. References 8
9.1. Normative References 8
9.2. Informative References 8
Authors' Addresses 8

1. Introduction

Modern video coding standards such as Thor [I-D.fuldseth-netvc-thor] include in-loop filters which correct artifacts introduced in the encoding process. Thor includes a deblocking filter which corrects artifacts introduced by the block based nature of the encoding process, and a low pass filter correcting artifacts not corrected by the deblocking filter, in particular artifacts introduced by quantisation errors of transform coefficients and by the interpolation filter. Since in-loop filters have to be applied in both the encoder and decoder, it is highly desirable that these filters have low computational complexity.

2. Definitions

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2.2. Terminology

This document will refer to a pixel X and eight of its neighbouring pixels A - H ordered in the following pattern.

```
+---+---+---+---+---+
|   |   | A |   |   |
+---+---+---+---+---+
|   |   | B |   |   |
+---+---+---+---+---+
| C | D | X | E | F |
+---+---+---+---+---+
|   |   | G |   |   |
+---+---+---+---+---+
|   |   | H |   |   |
+---+---+---+---+---+
```

Figure 1: Filter pixel positions

In Thor the frames are divided into filter blocks (FB) of 128x128, 64x64 or 32x32 pixels, which is signalled for each frame to be filtered. Also, each frame is divided into coding blocks (CB) which range from 8x8 to 128x128 independent of the FB size. The filter described in this draft can be switched on or off for the entire frame or optionally on or off for each FB. CB's that have been coded using the skip mode are not filtered, and if a FB only contains CB's that have been coded in skip mode, the FB will not be filtered and no signal will be transmitted for this FB.

If the frame can't fit a whole number of FB's, the FB's at the right and bottom edges are clipped to fit. For instance, if the frame resolution is 1920x1080 and the FB size is 128x128, the size of the FB's at the bottom of the frame becomes 128x56.

3. Filtering Process

Given a pixel X and its neighbouring pixels described above we can define a general non-linear filter as:

```
X' = X + a*constrain(A-X) + b*constrain(B-X) +
         c*constrain(C-X) + d*constrain(D-X) +
         e*constrain(E-X) + f*constrain(F-X) +
         g*constrain(G-X) + h*constrain(H-X)
```

Figure 2: Equation 1

where constrain(x) is a function limiting the range of x.

If a neighbour pixel is outside the image frame, it is given the same value as the closest pixel within the frame. To avoid dependencies

prohibiting parallel processing, all neighbour pixels must be the unfiltered pixels of the frame being filtered.

Experiments in Thor have shown that a good compromise between complexity and performance is a=c=f=h=1/16, b=d=e=g=3/16, and a good constrain function has been found to be:

```
constrain(x, s, d) =
   sign(x) * max(0, abs(x) - max(0, abs(x) - s +
                              (abs(x) >> (d - log2(s)))))
```

Figure 3: Equation 2

where sign(x) returns 1 or -1 if x is positive or negative respectively, s denotes the strength of the filter by which x will be clipped, and d further constrains the range of x so that the output of the function will linearly approach 0 as abs(x) approaches 2^d. d depends on the frame quality (qp) and is computed as

```
d = bitdepth - 4 + qp/16
```

Figure 4: Equation 4

for the luma plane and

```
d = bitdepth - 5 + qp/16
```

Figure 5: Equation 5

for the chroma planes.

The constrain function can be visualised as follows

```
          s                              ----
                                        /    ----
                                       /         ----
          0 ------------              /              ------------
                        ----         /
                            ----    /
          s                     ----
                     -2^d          -s 0 s           2^d
```

Figure 6: Graph 1

The filter strength s can be 1, 2 or 4 signalled at frame level when the bitdepth is 8. The strengths are scaled according to the bitdepth, so they become 4, 8 and 16 when the bitdepth is 10, and 16, 32 and 64 when the bitdepth is 12. The rounding is to the nearest integer.

This gives us the equation:

```
X' = X + (1*constrain(A-X, s, d) + 3*constrain(B-X, s, d) +
         (1*constrain(C-X, s, d) + 3*constrain(D-X, s, d) +
         (3*constrain(E-X, s, d) + 1*constrain(F-X, s, d) +
         (3*constrain(G-X, s, d) + 1*constrain(H-X, s, d)
```

Figure 7: Equation 6

The filter leaves the encoder 13 different choices for a frame. The filter can be disabled for the entire frame, or the frame is filtered using all distinct combinations of strength (1, 2 or 4 scaled for bitdepth), non-skip FB signal (enabled/disabled) and FB size (32x32, 64x64 or 128x128). Note that the FB size only matters when FB signalling is in use.

The decisions at both frame level and FB level may be based on rate-distortion optimisation (RDO), but an encoder running in a low-complexity mode, or possibly a low-delay mode, may instead assume that a fixed mode will be beneficial. In general, using s=2, a QP dependent FB size and RDO only at the FB level gives good results.

However, because of the low complexity of the filter, fully RDO based decisions are not costly. The distortion of the 13 configurations of the filter can easily be computed in a single pass by keeping track of the distortions of the three different strengths and the bit costs for different FB sizes.

The filter is applied after the deblocking filter.

4. Further complexity considerations

The filter has been designed to offer the best compromise between low complexity and performance. All operations are easily vectorised with SIMD instructions and if the video input is 8 bit, all SIMD operations can have 8 bit lanes in architectures such as x86/SSE4 and ARM/NEON. Clipping at frame borders can be implemented using shuffle instructions.

5. Performance

The table below shows filters effect on the bandwidth for a selection of 10 second video sequences encoded in Thor with uni-prediction only. The numbers have been computed using the Bjontegaard Delta Rate (BDR). BDR-low and BDR-high indicate the effect at low and high bitrates, respectively, as described in BDR [BDR].

The effect of the filter was tested in two encoder low-delay configurations: high complexity in which the encoder strongly favours compression efficiency over CPU usage, and medium complexity which is more suited for real-time applications. The bandwidth reduction is somewhat less in the high complexity configuration.

	MEDIUM COMPLEXITY			HIGH COMPLEXITY		
Sequence	BDR	BDR-low	BDR-high	BDR	BDR-low	BDR-high
Kimono	-2.6%	-2.3%	-3.2%	-1.7%	-1.7%	-1.9%
BasketballDrive	-3.9%	-3.1%	-5.0%	-2.8%	-2.4%	-3.5%
BQTerrace	-7.4%	-4.0%	-10.2%	-4.8%	-2.4%	-6.8%
FourPeople	-5.5%	-3.9%	-8.2%	-3.8%	-2.9%	-5.1%
Johnny	-5.2%	-3.5%	-8.2%	-3.3%	-2.7%	-4.5%
ChangeSeats	-6.4%	-3.9%	-10.2%	-4.7%	-2.6%	-6.9%
HeadAndShoulder	-9.3%	-3.4%	-19.6%	-6.2%	-2.6%	-11.8%
TelePresence	-5.8%	-3.6%	-10.0%	-4.5%	-3.3%	-6.6%
Average	-5.8%	-3.5%	-9.3%	-4.0%	-2.6%	-5.9%

Figure 8: Compression Performance without Biprediction

While the filter objectively performs better at relatively high bitrates, the subjective effect seems better at relatively low bitrates, and overall the subjective effect seems better than what the objective numbers suggest.

If biprediction is allowed, there is generally less bandwidth reduction as the table below shows. These results reflect low-delay biprediction without frame reordering.

	MEDIUM COMPLEXITY			HIGH COMPLEXITY		
Sequence	BDR	BDR-low	BDR-high	BDR	BDR-low	BDR-high
Kimono	-2.2%	-2.0%	-2.7%	-1.4%	-1.3%	-1.5%
BasketballDrive	-3.1%	-3.0%	-3.3%	-1.9%	-2.0%	-1.7%
BQTerrace	-5.4%	-4.3%	-6.5%	-3.9%	-3.6%	-3.8%
FourPeople	-3.8%	-2.8%	-5.2%	-2.4%	-1.8%	-3.0%
Johnny	-3.8%	-3.1%	-4.8%	-2.4%	-2.2%	-2.7%
ChangeSeats	-4.4%	-3.1%	-6.5%	-3.2%	-2.6%	-3.9%
HeadAndShoulder	-4.8%	-3.0%	-8.1%	-3.0%	-2.7%	-3.7%
TelePresence	-3.4%	-2.3%	-5.5%	-2.2%	-1.7%	-3.1%
Average	-3.9%	-2.9%	-5.3%	-2.5%	-2.2%	-2.9%

Figure 9: Compression Performance with Biprediction

6. IANA Considerations

This document has no IANA considerations yet. TBD

7. Security Considerations

This document has no security considerations yet. TBD

8. Acknowledgements

The authors would like to thank Gisle Bjontegaard for reviewing this document and design, and providing constructive feedback and direction.

9. References

9.1. Normative References

[I-D.fuldseth-netvc-thor]
Fuldseth, A., Bjontegaard, G., Midtskogen, S., Davies, T., and M. Zanaty, "Thor Video Codec", draft-fuldseth-netvc-thor-03 (work in progress), October 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

9.2. Informative References

[BDR] Bjontegaard, G., "Calculation of average PSNR differences between RD-curves", ITU-T SG16 Q6 VCEG-M33 , April 2001.

Authors' Addresses

Steinar Midtskogen
Cisco
Lysaker
Norway

Email: stemidts@cisco.com

Arild Fuldseth
Cisco
Lysaker
Norway

Email: arilfuld@cisco.com

Mo Zanaty
Cisco
RTP,NC
USA

Email: mzanaty@cisco.com