Novel Motion-Compensated Spatio-Temporal Filtering Scheme for x265 Open-Source Video Encoder

Ashok Kumar Mishra, Alex Giladi, Dan Grois,
Snehaa Giridharan, Keshav E , Santhoshini Sekar

AGENDA

1 **INTRODUCTION**

2 **BACKGROUND**

3 **NOVEL MOTION COMPENSATED SPATIO-TEMPORAL FILTERING**

- Motion Search
- Hierarchical Motion Estimation
- Motion Compensation & Temporal Filtering

4 **EXPERIMENTS & RESULTS**

5 **SUMMARY & CONCLUSIONS**

Introduction

A video codec is a software or a hardware that compresses and decompresses digital video. A device that only compresses is typically called an encoder, and one that only decompresses is a decoder.

There is a strong demand to decrease the video transmission bitrate without reducing visual quality [2].

x265 – One of the most popular open-source encoders compliant to HEVC standard [3] – [5].

Recent x265 development efforts have been focused on further improving the coding gains.

Specifically, the Motion Compensated Spatio-Temporal Filtering (MCSTF) is useful for pictures that contain a high level of noise.

MCSTF (Motion-Compensated Spatio - Temporal Filtering) is an encoder-only temporal filter done as a pre-processing step.

Background

Adapting QPs on a block-basis based upon the frequency of the block being referenced and the spatial complexity of the block is sub optimal

Spatio-temporal content properties and block-level encoding parameters like QP, pixel intensity are being used for temporal filtering in MCSTF

MCSTF improves compression efficiency of noisy videos (camera captured)

In HM, MCSTF is applied for the first frame of every GOP (Group Of Pictures)

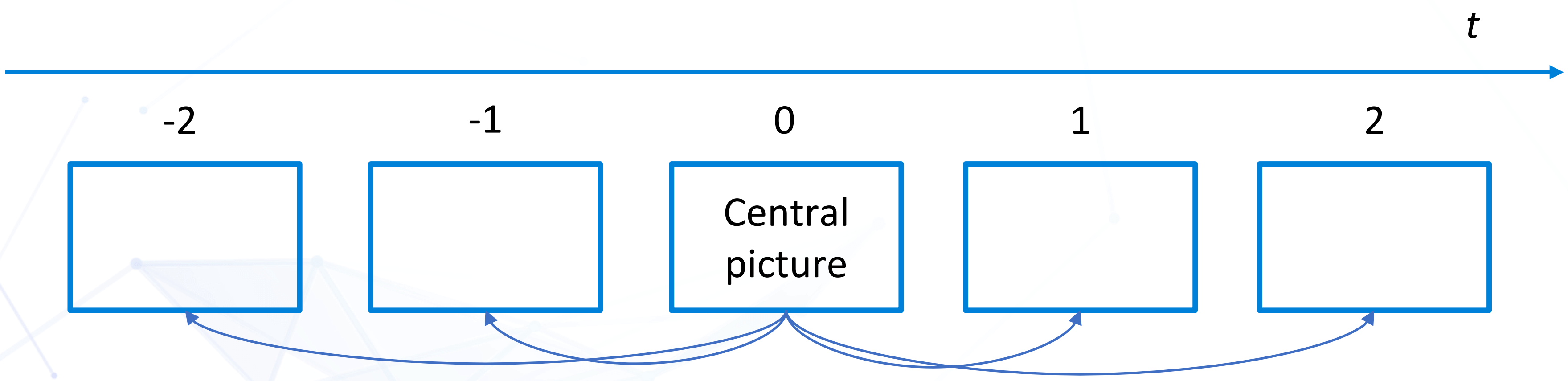
Novel Motion Compensated Spatio-Temporal Filtering

- A temporal filter is introduced directly before encoding.
- During pre-processing, pictures before and after the current picture are read.
- Block-based motion compensation relative to the current is applied on those source pictures.
- Previously generated motion vectors across different resolutions are used to find the best temporal correspondence for low-pass filtering.
- The temporal filtering is applied to every I- and P-frames.
- The filter strength drops off rapidly if the original images are different.
- MCSTF is done in 3 steps:

Hierarchical motion estimation → Motion Compensation → Temporal Filtering

Motion Search

- The motion estimation for temporal filtering is done in a temporal window of 5 adjacent frames: 2 past, 2 future and 1 central picture for producing a single filtered picture.
- Motion estimation is applied between the central picture and each future or past picture
- Multiple motion-compensated predictions are generated and then combined by using adaptive filtering to produce a final noise-reduced picture.



Block-based motion search and compensation

Hierarchical Motion Estimation

A hierarchical motion estimation scheme is employed (layers *L0*, *L1* and *L2* as illustrated)

Subsampled pictures generated for all reference pictures and original picture

↓

Motion estimation done for each 16x16 block in L2

↓

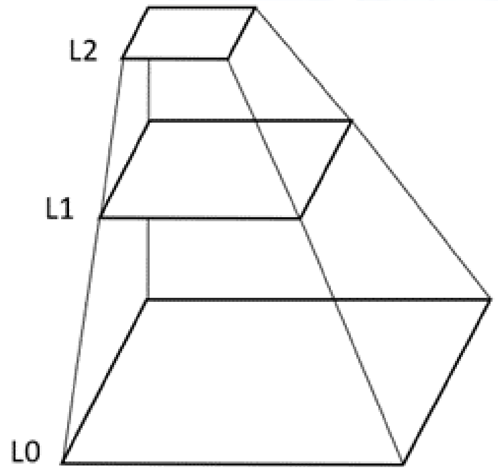
Motion estimation done for each 16x16 block in L1

↓

Motion estimation done for each 16x16 block in L0

↓

Subpixel motion is estimated for each 8x8 block by using an interpolation filter on L0



Different layers of the hierarchical motion estimation [1]

Motion Compensation & Temporal Filtering

- The motion compensation is applied on the pictures before and after the original picture according to the best matching motion for each block.
- The Interpolation filter is then applied to the current pixels, and after that, the filtered picture is encoded.
- The new sample value, is calculated using below equation [1] where,
  - I_0 is the original pixel, $I_r(i)$ is the intensity of the corresponding pixel within the motion compensated picture *i*, and $w_r(i, a)$ is the weight of the motion compensated picture where *a* is the number of available motion compensated pictures.

$$I_n = \frac{I_o + \sum_{i=0}^{3} w_r(i, a) I_r(i)}{1 + \sum_{i=0}^{3} w_r(i, a)}$$

Experimental Setup

- Experiments were conducted to compare with and without MCSTF in x265
- 1080p-8bit and 1080p-10bit videos tested with CRF values => 24, 25, 26, 27 at medium preset for random access and intra-period = 250 frames to compute BD% values [11]

SAMPLE COMMAND:

With MCSTF

./x265 --input Netflix_Tango.yuv --input-res 1920x1080 --fps 60 --input-depth 8 --input-csp i420 --output Netflix_Tango_mcstf.hevc --csv Netflix_Tango _mctf.csv --crf 24 --psnr --ssim --preset slow --profile main --bframes 5 --frame-threads 1 --mcstf

Without MCSTF

./x265 --input Netflix_Tango.yuv --input-res 1920x1080 --fps 60 --input-depth 8 --input-csp i420 --output Netflix_Tango_mcstf.hevc --csv Netflix_Tango _mctf.csv --crf 24 --psnr --ssim --preset slow --profile main --bframes 5 --frame-threads 1

Experimental Setup

Video Title	Video Resolution	Video FPS	Video Length (sec)
Tears of Steel	1920x1080	60	48
Netflix_Tango	1920x1080	60	5
Netflix_FoodMarket	1920x1080	60	10
Netflix RitualDance	1920x1080	60	10
Netflix_RollerCoaster	1920x1080	60	20

Results

x265: MCSTF vs no-MCSTF for 8bit videos

Video Title	BD-RATE (%)	BD-PSNR (dB)
Tears of Steel	-9.9	0.3
Netflix_Tango	-4.2	0.2
Netflix_FoodMarket	-13.8	0.4
Netflix RitualDance	-5.5	0.2
Netflix_RollerCoaster	-7.8	0.3

On an average MCSTF provides **8.24%** bitrate savings in x265 at medium preset with 8-bit videos

Results

x265: MCSTF vs no-MCSTF for 10bit videos

Video Title	BD-RATE (%)	BD-PSNR (dB)
Tears of Steel	-12.6	0.4
Netflix_Tango	-8.9	0.4
Netflix_FoodMarket	-15.3	0.5
Netflix_RitualDance	-6.8	0.3
Netflix RollerCoaster	-9.2	0.4

On an average MCSTF provides **10.56%** bitrate savings in x265 at medium preset with 10-bit videos

Experimental Setup

Video Title	Video Resolution	Video FPS	Video Length (sec)
Ritual Dance	1920x1080	60	10
Market Place	1920x1080	60	10
Daylight	3840x2160	60	5
Catrobot	3840x2160	60	5

Results

x265: MCSTF vs no-MCSTF for 10bit videos (CQP – 22, 27, 32, 37 for veryslow preset) for random access main-10 and intra period of 64 frames

Title	Video Resolution	Video-FPS	BD-RATE (%)	BD-PSNR (dB)
Ritual Dance	1920x1080	60	-6.0	0.3
Market Place	1920x1080	60	-14.8	0.4
Daylight	3840x2160	60	-31.5	0.5
Catrobot	3840x2160	60	-25.9	0.6

On an average MCSTF provides **19.5%** bitrate savings in x265 for veryslow preset

Results

HM 16.26: MCSTF vs no-MCSTF for 10bit videos (CQP – 22, 27, 32, 37 for random access main-10 and intra period of 64 frames)

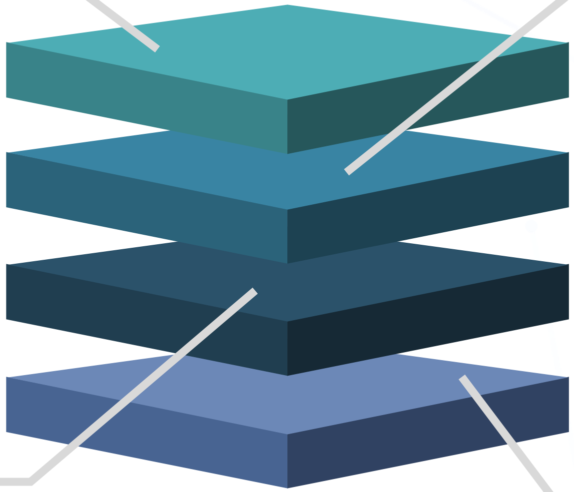
Title	Video Resolution	Video-FPS	BDRATE(%)	BD-PSNR(dB)
Ritual Dance	1920x1080	60	-1.2	0.1
Market Place	1920x1080	60	-5.9	0.2
Daylight	3840x2160	60	-11.9	0.2
Catrobot	3840x2160	60	-8.9	0.1

On an average MCSTF provides **6.9%** bitrate savings in HM

Summary & Conclusions

MCSTF is an encoder only preprocessing temporal filter.

It is used to reduce noise in video.



The novel MCSTF scheme significantly improves compression efficiency without loss in quality.

MCSTF introduces additional computational complexity, requires optimization to improve speed.

References

[1] Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 35th Meeting: Geneva, CH, 22–27 March 2019
[2] “Cisco Visual Networking Index: Forecast and Methodology, 2018–2023”, Online: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf, Cisco Systems Inc., 9 Mar. 2020.
[3] P. Ramachandran et al., "Speed-Distortion Optimization: Tradeoffs in Open-Source HEVC Encoding," in *SMPTE Motion Imaging Journal*, vol. 129, no. 7, pp. 17-25, Aug. 2020.
[4] Projects from VideoLAN, x265 software library and application, Online: https://www.videolan.org/developers/x265.html.
[5] x265 Documentation, Online : https://x265.readthedocs.io/en/stable/index.html
[6] ITU-T, Recommendation H.265 (04/13), Series H: Audiovisual and Multimedia Systems, Infrastructure of audiovisual services – Coding of Moving Video, High Efficiency Video Coding.
[7] Projects from VideoLAN, x264 software library and application, Online: http://www.videolan.org/developers/x264.html
[8] FFmpeg multimedia framework, Online: https://ffmpeg.org/
[9] GStreamer multimedia framework, Online: https://gstreamer.freedesktop.org/
[10] HandBrake video transcoder, Online: https://handbrake.fr/
[11] G. Bjøntegaard, “Calculation of average PSNR differences between RD-curves”, ITU-T Q.6/SG16 VCEG 13th Meeting, Document VCEG-M33, Austin, USA, Apr. 2001.

connect with us

www.multicorewareinc.com

www.facebook.com/multicoreware

www.twitter.com/multicoreware

www.linkedin.com/company/multicoreware-inc

www.instagram.com/multicoreware

www.youtube.com/multicoreware