Hardware Acceleration of Video Quality Metrics

Deepa Palamadai Sundar, Visala Vaduganathan, Xing C. Chen
Facebook Inc, 1 Hacker Way, Menlo Park, CA, USA 94025

ABSTRACT

Quality Metrics (QM) provide an objective way to measure perceived video quality. These metrics are very compute intensive and are currently done in software. In this paper, we propose an accelerator that can compute metrics like single scale and multi-scale Structural Similarity Index (SSIM, MS_SSIM) and Visual Information Fidelity (VIF). The proposed accelerator offers an energy efficient solution compared to traditional CPUs. It improves memory bandwidth utilization by computing multiple Quality metrics simultaneously.

Keywords: Video transcoding, accelerator, SSIM, MS-SSIM, VIF, PSNR, ASIC

1. INTRODUCTION

With the advancement in digital media and growing demand for video content, objective quality metrics is becoming increasingly important. Video transcoding is a very common use-case in many data centers and deploying video transcoding at scale requires on the fly computation of objective quality metrics to serve optimal videos to the users. This process often requires these objective metrics to be computed at a variety of viewport resolutions. These objective metrics are often corelated with subjective opinion scores on test data sets that represent the target use-cases. Over the last several years, several objective metrics evolved to closely represent subjective video quality. SSIM (Structural Similarity Index Measure) [1], MS-SSIM (Multi-Scale SSIM) [2], and VMAF (Video Multimethod Assessment Fusion) [3] are some of the better-known and most widely used objective metrics being used in the industry. Our work also includes the more traditional image/video pixel quality metrics, PSNR (Peak Signal-to-Noise Ratio), but because of its simple and straightforward nature, we will make limited reference to it throughout this manuscript.

Objective image and video quality measurements fall under three main categories. They are:

(1) Full reference metric - A complete reference image is available to compute the distorted image quality

(2) Partial reference metric - Partial information like a set of parameters of reference image is available

(3) No-reference metric - No reference image is available

These metrics can be very compute intensive and sometimes consume lot more resources than the encoding process itself – a topic that was discussed extensively at the Video@Scale-2019 event [4]. For example, the SSIM metric consumes a comparable amount of CPU resource to that of H.264 encoding process on traditional CPUs.

Although the CPU processing power has increased tremendously in recent years, the video usage has also grown exponentially. In response to the tremendous growth, on May 14, 2019, Facebook publicly announced a video transcoding custom-chip design, codenamed ``Mount Shasta'', to be deployed in our datacenters for the acceleration and compute optimization of our various video products [5].

The main functional blocks offered by this video transcoding ASIC (application-specific integrated circuit) are:

- Decoder – Accepts uploaded video; outputs uncompressed RAW video stream
- Scaler – Block to resize (shrink) the uncompressed video
- Encoders – Outputs compressed (encoded) video
- Quality measurement – Logic to measure the degradation in video quality after the encoding step

Video quality computations are very power hungry and can quickly limit the amount of processing that can be performed along with transcoding process. Moreover, the data transfer between HW transcoding to SW metrics calculation introduces

significant overhead that can severely limit the efficiency of the whole system. Thus, offloading video quality metrics calculation to an accelerator would offer an energy efficient solution. In this paper, we propose a hardware architecture that offers support for full reference as well as no-reference metric computation.

The paper is organized as follows. Section 2 covers supported full reference and no-reference metrics in hardware. Section 3 goes over the proposed hardware architecture in detail. Section 4 covers experimental results. Section 5 covers future enhancements and Section 6 concludes the paper.

2. SUPPORTED ALGORITHMS IN HARDWARE

Below are the quality metric algorithms supported by the proposed hardware.

(1) PSNR (Peak Signal-to-Noise Ratio)

(2) SSIM (Structural Similarity Index)

(3) MS-SSIM (Multi Scale SSIM)

(4) VIF (Visual Information Fidelity)

(5) No-reference blurriness metric

All the floating-point operations used in the above algorithms are modified to fixed-point equivalents for efficient hardware implementation. As it will be shown later, experiments have been conducted and verified that the error margin in calculating these metrics between the floating point and its fixed-point equivalent implementation is within acceptable limits.

2.1.1 SSIM/MS-SSIM

SSIM is one of the most popular full reference metrics that is used to measure perceptual image quality [1]. To do this, three components – luminance (l), contrast (c) and structure(s) – are calculated, and these are combined to obtain the overall similarity measure. Single and multi-scale computations are both available. MS-SSIM [2] offers better flexibility than single scale for accounting the variations in viewing conditions.

The overall SSIM index is given by:

$$\mathrm{SSIM}(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \tag{1}$$

where

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{2}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{3}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{4}$$

μ_x, μ_y, σ_x, σ_y and σ_{xy} are the local means, standard deviations and cross-covariance for images x, y. C_1, C_2 and C_3 are constants added to avoid instability when the denominator is very close to zero. Typically, α, β, γ parameters are set to 1 and $C_3 = C_2/2$. This results in a specific form of SSIM index

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C1)(2\sigma_{xy} + C2)}{(\mu_x^2 + \mu_y^2 + C1)(\sigma_x^2 + \sigma_y^2 + C2)} \tag{5}$$

The SSIM index is computed locally over a window size M. To obtain the global SSIM index for an image, local SSIM values are averaged across all pixel locations. Similarly, for a video sequence, SSIM values are calculated per frame and then averaged temporally.

Several adaptations of SSIM algorithm are available. In this paper, we support the computation of two such adaptations.

(1) FFMPEG SSIM - This uses an overlapped 8x8 approximation algorithm [6]

(2) LIBVMAF SSIM – This implements the algorithm as described in [1][7]

2.1.2 VIF

Visual Information Fidelity is yet another full reference metric to compute image quality. It determines how much information about the reference frame can be extracted from the distorted frame [8]. We propose a mechanism to extend the current hardware by reusing the SSIM/MS-SSIM pipeline to compute VIF with minor modifications.

2.1.3 No-reference blurriness metric

No-reference metric is an approach to compute quality when the reference frame is not available. This is used in applications where the upload (source) quality of a video needs to be established. A number of no-reference metric algorithms have been developed. In this paper, we provide a mechanism to compute the blurriness metric in hardware. Similar to full reference metrics, the fixed-point equivalent of the algorithm is implemented in hardware.

3. ACCELERATOR ARCHITECTURE

The proposed HW accelerator architecture is shown in Figure 1. The reference and distorted frames are read from memory and sent to the compute kernel for calculating the quality score. The accelerator can be programmed to compute any one of the supported full reference metrics. An operational mode to compute no-reference blurriness metric along with the full reference metric is provided in the accelerator.

Each of these components are described in detail below.

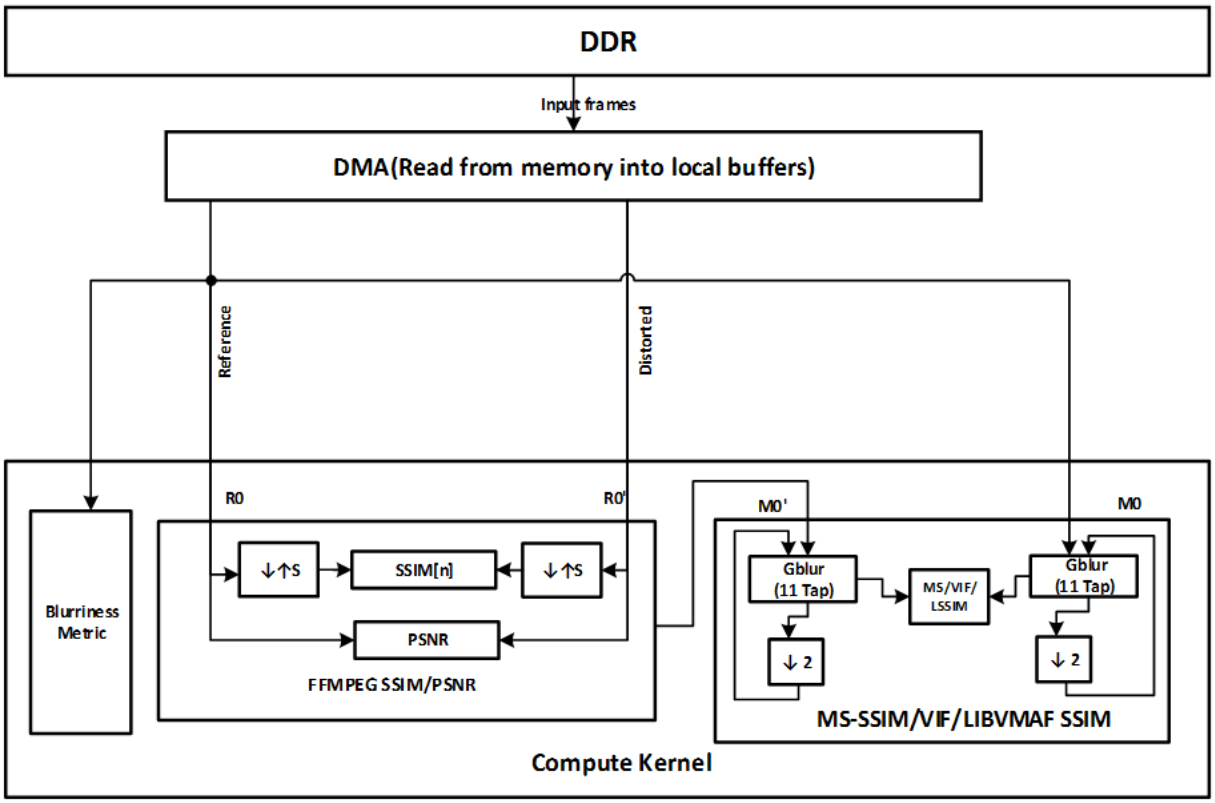

Fig 1. Architecture of Quality Metrics accelerator

3.1.1 DMA controller

The DMA (Direct Memory Access) controller block is responsible for reading video input frames from memory. It reads both the reference and distorted frames from DDR and loads them into local input buffers. A ping/pong buffer to hide memory read latency is used at the input. The data is read from memory in units of a predefined input block size. The block size is a factor of both the input buffer size and the bandwidth supported by the on-chip network chosen.

All these metrics compare reference and distorted frames pixel by pixel. This requires input blocks of both frames be spatially aligned while feeding into the kernel. The controller ensures that the input reads of both frames are synchronized and reads it in a raster scan order from memory.

Memory bandwidth is an important resource that can be both limited and expensive in terms of power. By performing computation of multiple metrics simultaneously (full and no-reference) we are avoiding the need to re-read frames multiple times saving bandwidth and power.

3.1.2 Compute kernel

The kernel is the heart of the accelerator and computes the programmed quality metric. Once the reference and distorted frames are read from memory and scaled to the desired resolution, the kernel starts processing pixels.

Three kernels are used to implement the different algorithms.

(1) FFMPEG kernel - This computes PSNR and the SSIM index as per the FFMPEG [6] implementation

(2) SSIM kernel - This is a unified kernel to compute single-scale and multi-scale SSIM and VIF

(3) Blur kernel - This computes the blurriness score used in no-reference metrics.

Often times, while determining video encoding quality, there is a need to compute the quality loss when it is viewed at different viewport resolutions. This means that the reference and distorted frames have to be scaled to the chosen viewport resolution.

Our proposed architecture offers scaler support to upscale or downscale both the reference and distorted frames. This not only allows inline processing but further optimizes memory bandwidth, thereby avoiding the need to write/read the scaled output to/from memory. The choice of scaler also plays an important role and should not introduce more quality loss during the scaling process. The accelerator uses a set of programmable filters with 10 taps for scaling input frames.

All filter operations involved in these kernels are separable. The hardware implements 1D filters performing filtering in horizontal followed by vertical direction. This significantly reduces the number of compute operations thereby reducing area and power.

For any of these metrics, block level scores can also be easily calculated. Block level information is very useful in identifying the regions that have higher impact on quality in a frame. Support is also added to compute PSNR in parallel with other metrics.

3.1.3 FFMPEG Kernel

As the name suggests, this kernel is used to compute SSIM index using 8x8 overlapped approximation algorithm. Figure 2 shows the compute operations to obtain LCS (Luma, Contrast, Structurer) components and combine them to get the SSIM Index.

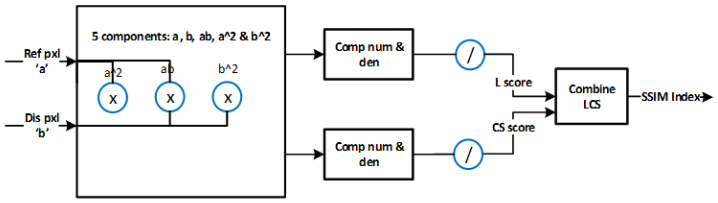

Fig 2. LCS compute logic

The cost (area and power) of the kernel is directly proportional to the number of multipliers and dividers used. This dictates how many pixels can be processed per cycle based on the area/power budget available. The hardware is scalable, that is, the number of kernels that run in parallel enhances the compute speed. The number of paths that can be added is again a tradeoff between performance and area/power consumption.

3.1.4 SSIM Kernel

The SSIM kernel is a unified kernel that computes the SSIM index for both single scale as well as multi-scale operation. Figure 3 shows the internal details of the kernel.

The first step is to perform a Gaussian blur operation to smoothen the input frames. Five components similar to FFMPEG SSIM are computed. Each of these components are sent through a 11 tap gaussian blur filter before LCS values are calculated. For single scale (LIBVMAF SSIM), input pixels are sent once through the kernel. For MS-SSIM, the gaussian blurred output of both frames (the 'a' and 'b' component) are sent through a dyadic downsampler and looped back to the same kernel to process the higher scales. This process is repeated 'M' times where M is the number of scales supported. This feedback path helps us reuse the same hardware to compute all scales.

For a given input block, all scales are computed before processing the next input block read from memory. If the frame width or height is not a multiple of input block size, at the frame boundaries the kernel operates only on those pixels that lie within the frame.

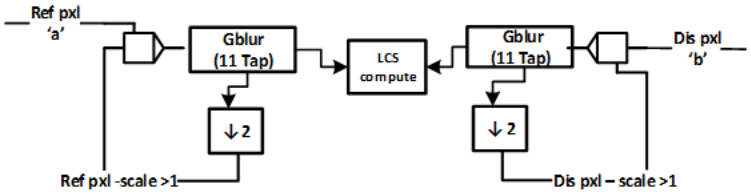

Fig 3. SSIM Kernel

The use of a dyadic down sampler on the gaussian blurred output is to ease hardware implementation. This is a simplification from the algorithm which uses an averaging filter iteratively to obtain the input for higher scales.

VIF metric is computed by reusing the same kernel, since it relies on the same fundamentals natural-scene statistics framework that SSIM is based upon. The kernel is enhanced to compute the logarithm operation as detailed in [8] on the variance/covariance (σ) components, which are calculated as part of LCS compute, and thus provide VIF scores as well. The VIF algorithm dictates varied filter taps for the different levels. The hardware accelerator simplifies this to reuse the same 11 tap gaussian filter across all levels.

3.2.3 Blur kernel

The blur kernel computes the blurriness metric that is used in the no-reference metric algorithms. Figure 4 shows the various stages involved in computing the metric.

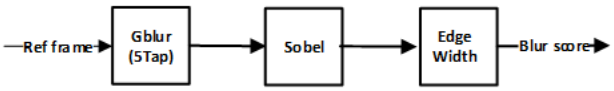

Fig 4. Blur kernel

The first step is to smoothen the input using a 5-tap gaussian blur filter. The smoothened output is then sent through a Sobel filter [9] [10], to compute the pixel gradients (Gx, and Gy), as follows.

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

The edge width block then determines the spread of the edge for each pixel, which are then used to compute the final blur scores.

To compute the edge width (spread), for each pixel we search in the direction computed by Sobel operator for a certain search window size (NXN). This also involves searching along diagonal directions as shown in Figure 5, which are natively not hardware friendly. We use a preload mechanism to load the diagonal neighbors upfront and reuse the memory read data during the search process. This saves multiple reads from memory for the same neighbor block.

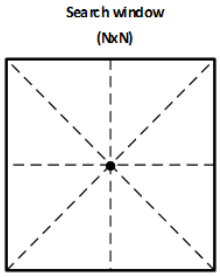

Fig 5. Edge width search

4. RESULTS

As described in earlier sections, fixed point version of the quality metrics algorithm is what the hardware implements. To validate our HW algorithm and implementation, we created a floating-point SW reference model which we call A-model to compare with the bit accurate C-model that models our HW implementation. The A-model was created by directly taking the core metric computation functions from the open-source software packages ffmpeg [6] (for FFMPEG ssim) and Libvmaf [7] (for SSIM_libvmaf, MS-SSIM and VIF), and putting them in the same test harness as our C-model so both A-model and C-model will take the same input images.

The main difference between A-model and C-model is that wherever A-model uses floating point representation which uses many more bits per value, C-model (our HW) will be using fixed point representation with fewer bits due to HW resource limitations, such as for Gaussian filtering coefficients or intermediate results inside score calculation. For VIF score calculation which involves log2() computation, C-model will need to also use some hardware viable approximations. We then feed images from about 400 real input streams of 4 input resolutions to both models. We report the average absolute difference between the A-model scores and C-model scores in Table 1, which is a measure of the effect of HW approximation from SW reference implementation.

Table 1. Average absolute error/difference between floating- and fixed-point model for different metrics

QP VALUE	RESOLUTION	SSIM FFMPEG	SSIM LIBVMAF	MS_SSIM	VIF
23	360p	0.00004	0.00023	0.00079	0.01352
	480p	0.00004	0.00062	0.00102	0.01358
	720p	0.00004	0.00032	0.00081	0.01387
	1080p	0.00004	0.00050	0.00087	0.01258
31	360p	0.00005	0.00024	0.00084	0.01409
	480p	0.00004	0.00063	0.00108	0.01436
	720p	0.00004	0.00035	0.00092	0.01451
	1080p	0.00004	0.00052	0.00093	0.01307

The memory bandwidth consumed while computing PSNR, no-reference blurriness and FFMPEG SSIM was also compared for different encoded resolutions. Figure 6 shows the comparison between what we see on traditional CPU vs proposed accelerator.

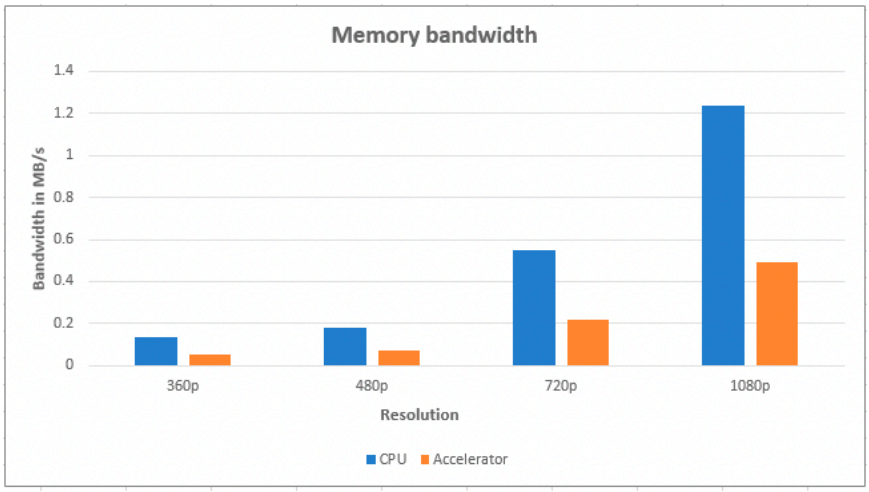

Fig 6. Memory bandwidth – CPU vs Accelerator

Some of the metrics, like VIF and MSSSIM, are very compute intensive and often consume significant power when run on servers. However, the HW Implementation can improve performance per unit of power (perf/W) by 100x magnitude.

5. FUTURE ENHANCEMENTS

In this paper, the proposed architecture implements all image/video quality algorithms for the luminance (luma) component. The hardware itself is agnostic to the component being processed and it can process either luma or chroma planes. We can do this either as separate passes where each plane is read and processed separately or have some additional pre-processing hardware to split these planes and feed it to the kernel in a specific order.

We intend to experiment with the usefulness of such color metrics and balance the benefits against the resulting increase in power usage to establish a good tradeoff, perhaps coupled with subjective testing for validation.

In our current approach, we are using the VIF scale features to approximate VMAF scores; yet, the architecture is also extensible with programmable support on filter and scaler coefficients. We can also enhance the accelerator to extend the VIF computation by making the gaussian blur coefficients programmable per level and include additional metrics like DLM [11] to eventually provide full VMAF scores.

6. CONCLUSION

In this paper, we proposed an architecture to accelerate video quality metrics. This can tremendously improve the performance compared to CPUs and help quality metrics serve as a tool in improving parameter selection in transcoding process. This being a first step in enhancing the quality compute operations, more complex algorithms can be investigated to explore the possibility of offloading them to hardware processing.

REFERENCES

[1] Z.Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” lEEE Transactions on Image Processing, vol.13, April (2004)

[2] Z.Wang, E. P. Simoncelli and A. C. Bovik, “Multi-Scale Structural Similarity For Image Quality Assessment”, 37th IEEE Asilomar Conference on Signals, Systems and Computers, Nov. (2003)

[3] Li, Z., Aaron, A., Katsavounidis, I., Moorthy, A., and Manohara, M., “Toward a practical perceptual video quality metric,” https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652 (2016)

[4] Ioannis Katsavounidis, “Video quality keynote”, https://atscaleconference.com/videos/video-scale-2019-video-quality-keynote/, Oct. 16, (2019)

[5] Lee, K. and Rao, V., “Accelerating facebook’s infrastructure with application-specific hardware.” https://engineering.fb.com/data-center-engineering/accelerating-infrastructure/ (2019)

[6] FFmpeg Developers. (2020). ffmpeg tool [Software]. Available from http://ffmpeg.org

[7] LIBVMAF, < https://github.com/Netflix/vmaf/tree/master/libvmaf>

[8] H. R. Sheikh, A. C. Bovik, “Image Information and Visual Quality”, lEEE Transactions on Image Processing, vol.15, February (2006)

[9] Kanopoulos, N., Vasanthavada, N., & Baker, R. L. (1988). Design of an image edge detection filter using the Sobel operator. IEEE Journal of Solid-State Circuits, 23(2), 358–367

[10] Sobel, I., Feldman, G., "A 3x3 Isotropic Gradient Operator for Image Processing", presented at the Stanford Artificial Intelligence Project (SAIL) in (1968)

[11] S. Li, F. Zhang, L. Ma, and K. Ngan, “Image Quality Assessment by Separately Evaluating Detail Losses and Additive Impairments,” IEEE Transactions on Multimedia, vol. 13, no. 5, pp. 935–949, Oct. (2011)

[12] G. Chaudhari, H. Lalgudi, H. Reddy, “Cross-codec encoding optimizations for video transcoding”, SPIE, Applications of Digital Image Processing XLIII, volume 11510, August (2020)