Efficient video processing at scale using MSVP

H.M. Reddy, Y. Chen, J. Lan, I. Katsavounidis, B. Anandharengan, H.G. Lalgudi, S. Alaparthi, G. Hua, H.-C. Chuang, P.-H. Wu, Z. Lei, A. Mastro, C. Petersen, G. Chaudhari, P. Prakash, S. Regunathan, S. Reddy, P. Venkatapuram, V. Rao, K. Noru, A. Bjorlin, M. Zeile, A. Lewis, A. Singh, A. Sunil, C.-C. Chen, C.-F. Lin, C. Chen, D.P. Sundar, D. Jayaraman, H. Ucar, H. Li, J. Singh, J. C.-C. Liu, K. R. Rachamreddy, K. Sriadibhatla, K. Datla, L. V. D. Berg, L. Feng, P. Jampani, R. Moola, R. Mallya, S. Jha, S. Pan, S. Srinivasan, V. Vaduganathan, X. Zha, Z. Wang, A. K. Sengottuvel, B. Alluri, B. Oshin, C. Kanumetta, E. Sahin, J. M. Athaide, J. Wu, K. C. Kurapati, K. Manthati, K. Thottempudi, R. R. Chennamsetti, K. R. Jagannath, S. Arvapalli, T. Kala, T. Wang, P. Chopda, K. Gandhi, A. Ramesh, R. Gupta, S. Fadnavis, A. Qassoud, C. Friedt, F. Li, H. Gao, J. Lee, M. Dixit, S. Ugaji, T. Karuturi, X. Xie, A. Narasimha, B. Jakka, B. Dodds, J. Yang, K. Skandakumaran, M. Modi, P. Modi, C. Stejerean, D. Ronca, H. Wang, N. Pham, L. Lu, H. Shen, J. Ning, K. Narayanan, L. Chen, N. Avidan, W. Arnold, F. Xu, G. Patil, V. Balan, S. D. Grandhi

Meta Platforms Inc., 1 Hacker Way, Menlo Park, CA

ABSTRACT

Videos uploaded to Meta's Family-of-Apps are transcoded into multiple bitstreams of various coding formats, resolutions and qualities to provide the best quality of experience across a wide variety of devices and connection bandwidth constraints. On Facebook alone, there are more than 4 billion video views per day. To address video processing at this scale, we need a video processing solution that can deliver the best video quality possible, with the shortest amount of encoding time, while being scalable, programmable and energy efficient. In this paper, we present Meta Scalable Video Processor (MSVP), a custom ASIC developed by Meta, that can transcode videos at the similar quality as software encoders, but with much higher throughput and lower power consumption. Each MSVP ASIC can deliver a peak SIMO (Single Input Multiple Output) transcoding performance of 4K 15fps with the highest quality configuration, and can deliver up to 4K 60fps with the standard quality configuration. This performance is achieved at ~10W of PCIe module power. We achieved a throughput gain of ~8x for H.264 when compared against libx264 SW encoding at medium preset. For VP9, we achieved a throughput gain of ~50x when compared against libvpx speed 2 preset. Key components of MSVP transcoding include video decoding, frame rescaling, encoding and quality metric computation. In this paper, we will first provide a high-level overview of the MSVP architecture. Then, detailed design of individual components will be introduced. Finally, benchmarking results that compare the compression efficiency and performance per Watt against software encoders will be presented.

Keywords: Video processing, Video transcoding, H.264/AVC, VP9, AV1, video on demand

1. INTRODUCTION

Processing video for video on demand (VOD) and live streaming use cases are very compute intensive. It involves transcoding large video files into more manageable formats and delivering to a wide variety of audiences. Figure 1 shows an example of a set of encodings for an uploaded video. As soon as a video is uploaded, H.264 encodings are first produced at different resolutions to serve a wide variety of client devices. H.264 is chosen at first because the compute cost for encoding is much lower and device support is the most prevalent. Once the watch time of a video increases, the need for higher compression efficiency rises and more advanced codecs are used to produce encodings with higher compute cost to reduce egress bitrate. Today, the demand for video content is greater than ever. Emerging use cases, such as generative AI content, will require a higher video infrastructure compute capacity. With this, we believe that a dedicated ASIC is the best solution in terms of energy efficiency.

Applications of Digital Image Processing XLVI, edited by Andrew G. Tescher, Touradj Ebrahimi,
Proc. of SPIE Vol. 12674, 1267414 · © The Authors. Published under a Creative Commons Attribution
CC-By 3.0 License · 0277-786X · doi: 10.1117/12.2685875

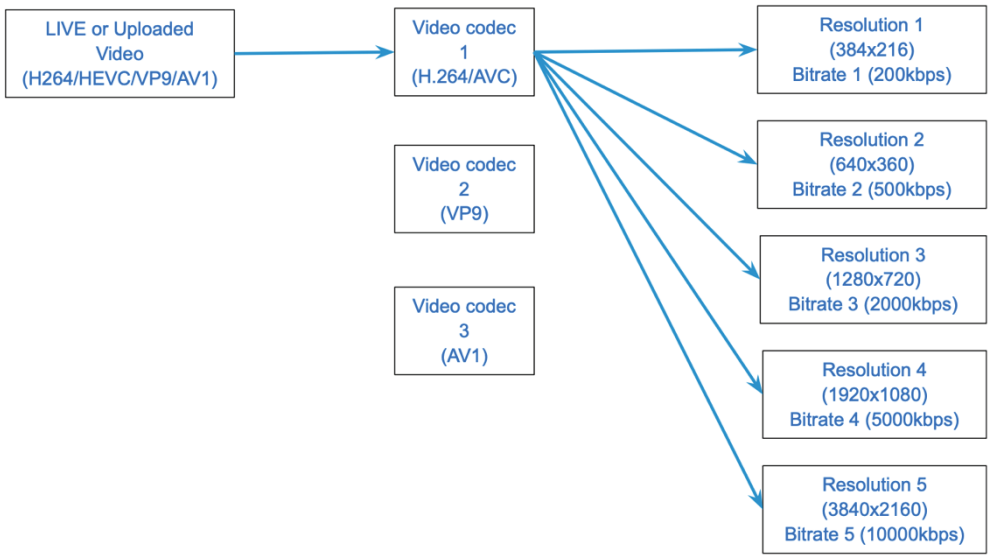

Figure 1. Video encodings at a streaming server

Many companies have engineered custom built accelerators that can perform video transcoding at scale. Alveo MA35D from AMD [3] offers 8 encoding engines per device and supports H.264, HEVC and AV1 with a total throughput of 32x 1080p60 streams. NVIDIA's latest GPU architecture Ada has 3 encoding accelerators NVENC [2] offering a total throughput of 107x AV1 1080p30 targeting cloud gaming applications. Intel's flex series GPU [1] has a performance specification of 36x 1080p60 streams of HEVC encoding. Google claims that its video accelerator, Video Coding Unit (VCU), offers 4-33x perf/TCO improvements over software encoding counterparts with quality improvements being continuously made through rate control optimizations [4].

In order to support Meta's video transcoding workload at scale, there are two major requirements. First, a very high-power efficiency. Second, a similar or better video quality than software encoders. There are certainly off-the-shelf video encoder IPs, but most of them are targeted at mobile devices with more tight area/power constraints. The result is that they cannot reach the same quality level as software encoders. With MSVP, we can improve the overall quality and compression efficiency across all codecs with much lower compute and power cost, and high stability, reliability, and near-infinite scaling.

In Section 2, we will go over the system level architecture of MSVP and its processing data flow. We will then present a high-level overview of key components of the transcoder IP. In addition, we will discuss block-level encoding algorithms that are accelerated within MSVP and frame-level algorithms that run on the control processor. Section 3 will deep dive into the details of the scaler and quality metrics module. In Section 4, we will compare quality and performance with software encoders. Section 5 will introduce how MSVP is used to enable Meta's advanced Adaptive BitRate streaming (ABR) encoding through the convex hull optimization scheme [7]. In Section 6, we present some of the challenges that engineering teams addressed when deploying MSVP in production.

2. MSVP ARCHITECTURE

2.1 System overview

The use cases for the video transcoding at Meta can be broadly classified into two categories: Video on Demand (VOD) and Live. Video on Demand use cases require transcoding with a blend of higher throughput and improved compression efficiency, while Live streaming is latency-critical, requiring faster processing time. MSVP can be configured to support both use cases. Using a multi-pipe, multi-pass, and scalable architecture, our solution provides high quality video transcoding with optimal energy efficiency.

Simplified view of MSVP System-on-Chip (SoC) is shown in Figure 2. Key components of MSVP SOC are:

- Multi CPUs Sub-system for resource management, security boot and transcoding FW Peripherals (SPI, UART, SMBUS and GPIOs), Debug & Trace connected to JTAG & USB, Power Management Unit.
- 4-lane PCIe (Gen4) PHY and controller connecting the ASIC to the Host CPU.
- LPDDR5 PHY and controller connecting to LPDDR5 memory (8GB).
- Transcoder IP for video transcoding operation.

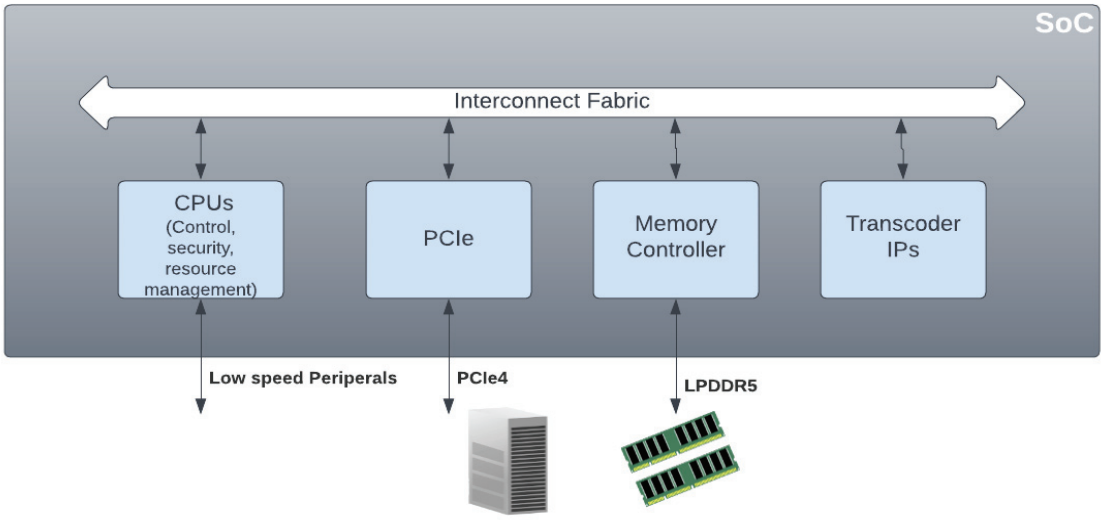

Figure 2. Key components of MSVP System-on-Chip

Figure 3 shows the data flow where the host (server) communicates with the on-chip CPU subsystem through the PCIe interface to schedule streams to be decoded, encoded, pre-processed or transcoded. Bitstreams will be pulled from the Host memory through the PCIE-DMA engine and stored in the local LPDDR. This operation can be triggered through the Host-firmware (FW) interface for every frame or a group of frames. When the final transcoded bitstreams are ready, FW will trigger the host to transfer the final transcoded bytes to host memory. Below figure shows the data flow. FFMPEG device driver is leveraged for the communication between the host and the device.

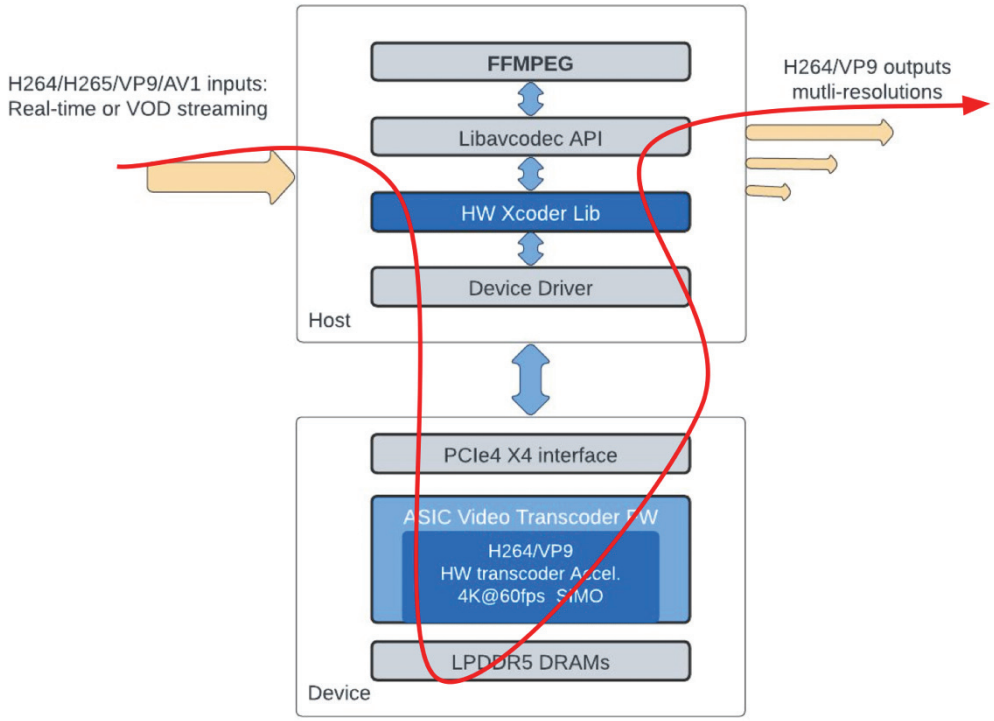

Figure 3. MSVP video transcoding data flow

The primary stages of the transcoding process are: Decoding, Preprocessing, Encoding, and Quality metrics calculation.

All these stages are implemented as memory-to-memory operations in MSVP, i.e., intermediate buffers are stored back to DRAM and re-fetched as needed by the downstream operation.

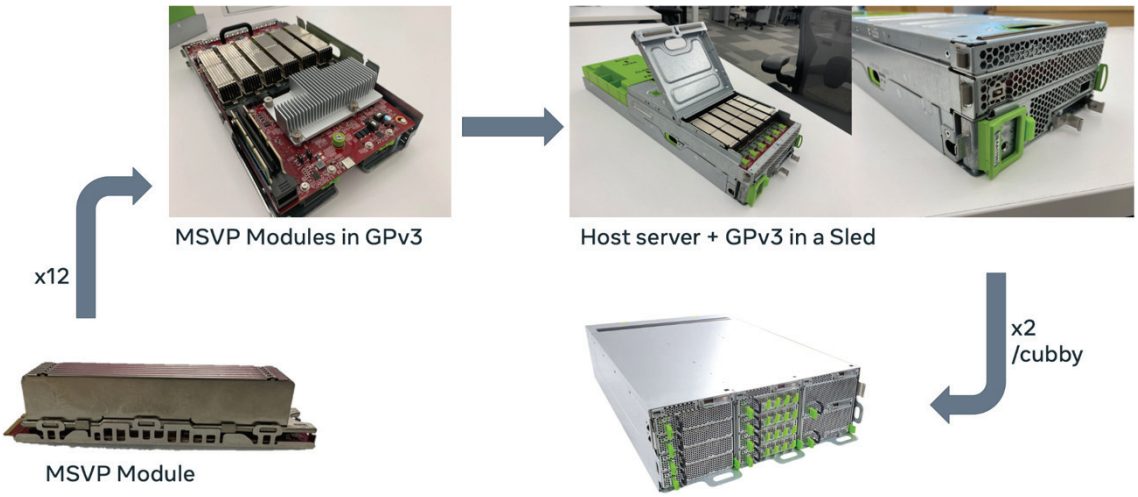

Figure 4. MSVP in the data center

Figure 4 shows MSVP on an M.2 module. The cards are housed in a Glacier Peak V3 (GPv3) carrier card and then paired with a Yosemite Version 3 (Yv3) host. For more details on the ASIC module, readers are referred to [22].

2.2 Transcoding pipeline

Figure 5 shows transcoding pipeline in a SIMO (Single-Input, Multiple-Output) use case where the ingest video is decoded once, scaled to different resolutions, encoded, and then followed by quality metrics, including Structure Similarity Index Map (SSIM) [16] and Visual Information Fidelity (VIF) [24] computation. Though MSVP can be run in a SISO (Single-Input, Single-Output) mode, it is less preferred for ABR encoding due to redundant decoding operations and data transfer. Below is the list of the key features of each stage in the transcoding pipeline.

- Decoding
    - A variety of elementary input video stream formats are supported, including H264, HEVC, VP9 and AV1
    - 8/10-bit pixel sample depths and YUV420 color format
- Preprocessing
    - Format Conversion
    - Scaling operation to multiple resolutions
    - Motion information calculation and Shot Change Detection.
- Encoding
    - Motion Estimation
    - Supports H264 and VP9 coding standards.
    - 8-bit pixel sample depth, YUV420 color format
- Quality Metrics (QM)
    - Full reference: SSIM, MS-SSIM, VIF, PSNR (Luma and Chroma)
    - No-Reference blurriness.

Meta opted to use an off-the-shelf multi-format decoder IP block, given the tight development schedule and the very high validation complexity of the normative decoder operations. In the next few sections, we will go over the algorithm and architecture of the remaining modules.

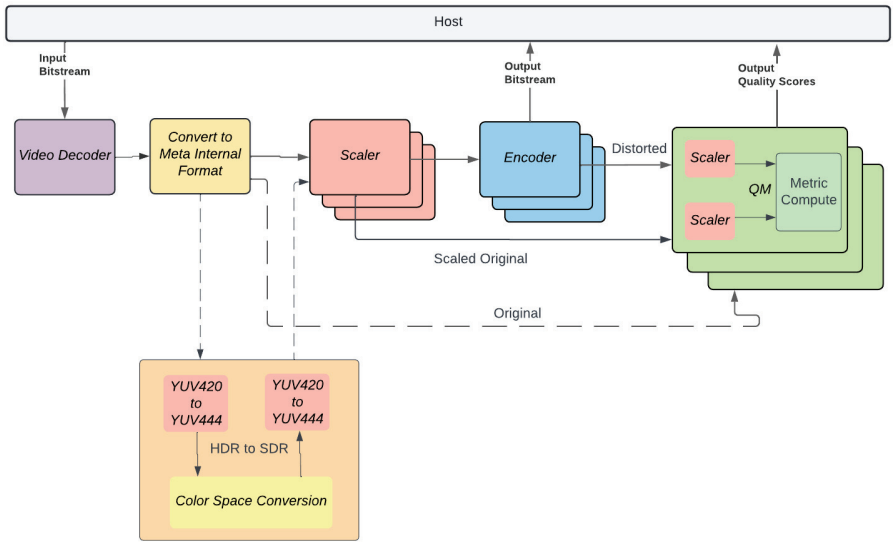

Figure 5. MSVP transcoding pipeline

3. PRE-PROCESSING

Scaling is used in many places in the transcoding pipeline. First, an uploaded video is to be encoded in multiple resolutions, so before encoding, an uploaded original video needs to be scaled to multiple - typically, 4 to 6 - encoding resolutions. Second, an encoded video will be evaluated for its quality not just at the encoding resolution, but also at multiple viewport resolutions. So, scaling is needed before quality metric computation. Another scenario where scaling is needed is when converting High Dynamic Range (HDR) video to Standard Dynamic Range (SDR) video, Chroma upscaling and downscaling is needed for conversion between YUV420 and YUV444 formats. Given how often scaling is used, it is essential to maximize its quality. Moreover, given the very wide distribution of input video resolutions, which includes very atypical values, for example frame heights of 1070 pixels, rather than the more typical 1080 pixels, it is of utmost importance to support arbitrary input and output resolutions in the scaler module.

The MSVP scaler was also designed to provide high quality and performance in mind. It supports programmable and wider filter kernels, higher precision filter coefficients, and wider data paths. The MSVP scaler contains a generic horizontal scaler followed by a vertical scaler, with a final output/post-processing stage. Key characteristics of the scaler are as follows:

1. Supports arbitrary scaling ratios defined by any input/output frame resolution up to 4K, in the range [0.25, 7.50]
2. High precision and up to 25-tap wide filters, enabling multiple interpolation methods, such as linear, bilinear, Lanczos (alpha<=5) and any other custom filters.
3. Support input-output pixel phase shifts, independently for Luma and Chroma planes.
4. Supports multiple frame boundary extension types such as pixel copying and mirroring.
5. Output post-processing: dithering, output range limiting.

Compared to some 3rd party scalers, the MSVP scaler can achieve PSNR gains of 9dB, achieving 47.5dB vs 38.5dB in a downscale-by-2 then upscale-by-2 experiment. Performance-wise, MSVP scaler can comfortably support 4K HDR to SDR conversion as well as 5 sequential downscaling operations at 60 frames/second.

Pre-processing module also supports HDR to SDR conversion. HDR frames in YUV420 are first converted to YUV444 by upsampling the chroma channels by a factor of 2, then the YUV444 frames are converted to RGB and go through various color space remappings. In the final stage of converting YUV444 back to YUV420 space, we simply leverage the generic scaler to do the chroma downsampling by a factor of 2, to output in YUV420 color format. All the remapping operations are done through programmable lookup tables and matrix operations so it can support different color spaces (such as PQ or HLG).

4. ENCODER ARCHITECTURE AND ALGORITHMS

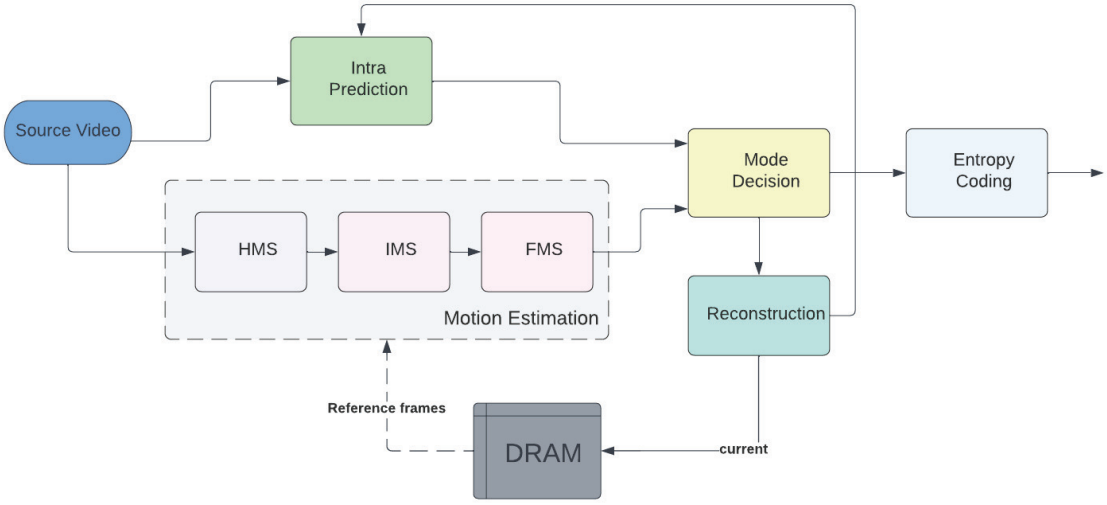

Figure 6. Simplified video encoding pipeline

Figure 6 shows a simplified system diagram of modern hybrid video encoders. Intra-coding tries to reduce spatial redundancy, and inter-coding is utilized to remove temporal redundancy in source video frames. Different stages of motion estimation are applied in inter-coding to find the best prediction among all possible block positions in previously reconstructed reference frames. Entropy coding is the lossless compression part that encodes all syntax elements, including encoding modes, motion vectors and quantized residual coefficients. In MSVP, we designed HW algorithms and chose coding tools that can get the highest return-on-investment (ROI) in terms of silicon area and power. A few highlights of the MSVP encoding algorithms are described in the following sections at different levels.

4.1 Motion Estimation

To find accurate motion vectors that closely match the block currently being encoded, a full motion estimation pipeline often includes a multistage search to balance among large search range, computing complexity, and accuracy. Although MSVP lacks the flexibility of iterative software motion search algorithms, such as diamond or hexagon searches, hardware motion estimation can search multiple blocks in parallel. Thus, it allows us to search more candidates, cover a larger search set and more reference frames in both single direction and bidirectional mode, and search all supported block partition shapes in parallel.

4.2 Mode Decision

There is a very large number of decisions to make in video encoding: intra/inter modes, partition block size, transform block types/sizes, and more. To ensure highest quality, full Rate Distortion Optimization (RDO)-based mode decision algorithm is used. The goal of the RDO algorithm is to find the best coding mode that minimizes the joint RD cost, $J = D + \lambda R$, where D is the distortion of a certain decision (typically, MSE) and R is the number of bits associated with that decision, while λ is the Lagrange multiplier that weighs these two factors appropriately and is typically derived from the quantization parameter (QP). In software encoders, various heuristics have been developed and used in many software video encoders that try to limit the number of candidates considered for this minimization process. MSVP supports exhaustive RDO for multiple coding standards [11] and at almost all mode decision stages. The high-level architecture diagram of the Mode Decision module is shown in Figure 7. Partition levels are parallelized to meet throughput requirements without sacrificing quality. Distortion calculation in the RDO engine is straightforward. However, accurate rate estimation involves knowing neighboring context, since modern video coding standards adopt sophisticated context-adaptive entropy coding techniques. MSVP has a HW-friendly rate estimation [12] that simplifies context with minimal impact in quality.

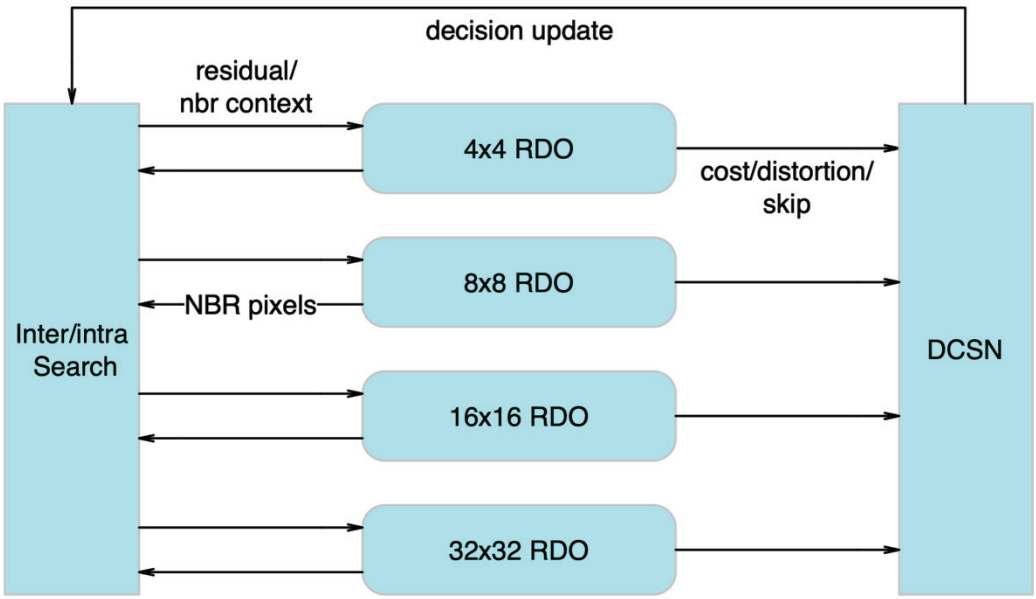

Figure 7. Mode Decision Architecture

4.3 Adaptive Quantization

Quantization of residues after subtracting intra or inter predictions from current block pixel values is parametrized by a quantization parameter (QP) that controls the rate-distortion tradeoff. Changing the QP value within a frame to:

1. Meet a target bit budget for a frame,
2. Distribute the bits in a perceptual optimized fashion and
3. Distribute more bits to blocks that are referenced more in the temporal prediction.

In MSVP's hardware encoder, QP is determined using both spatial and temporal characteristics. The human visual system is less sensitive to quality loss at high texture or high motion areas. This characteristic is exploited spatially wherein a higher QP value can be applied to coding blocks with high texture and lower QP value for smooth area. For temporal QP adjustments, coding blocks that are referenced more in the future can be quantized with a lower QP to get higher quality, such that future coding blocks that reference these blocks will benefit from it.

4.4 Frame level algorithms and rate control

MSVP provides frame-level buffer management, QP selection and rate control through the control CPU in the ASIC. This enables flexible algorithm tuning to meet the production needs in a short time frame. Frame-level algorithms for MSVP encoder can be configured to be either 2-pass or 1-pass depending on whether it is VoD or Live use cases. In the high quality (longer latency) VoD 2-pass mode, MSVP looks ahead at N frames and collects statistics, such as intra/inter cost and motion vectors, from these frames.

Figure 8 shows the encoder frame-level control flow. A fast look-ahead encoding (first pass encoder) is done to collect video statistics. These statistics [23] are then used for:

1. Intra/Key frame decision
2. Frame type / sub-GOP structure decision
3. Frame QP decisions

4. Spatial adaptation of QP
5. Temporal filtering strength selection (for VP9)

Key frame and sub-GOP decisions are decided based on the complexity of the video. For example, it is more efficient to encode a lesser number of B-frames in between 2 P frames in high motion scenes. In addition to frame type, frame QP is derived based on the correlation strength in the referencing structure. On top of the frame QP variations, spatial and temporal changes to QP are applied. In essence, back-propagation on the reference tree is used to identify important blocks in each of the reference frames in the lookahead buffer. The accumulated reference importance of the to-be coded frame is modulated to control the AQP of each block.

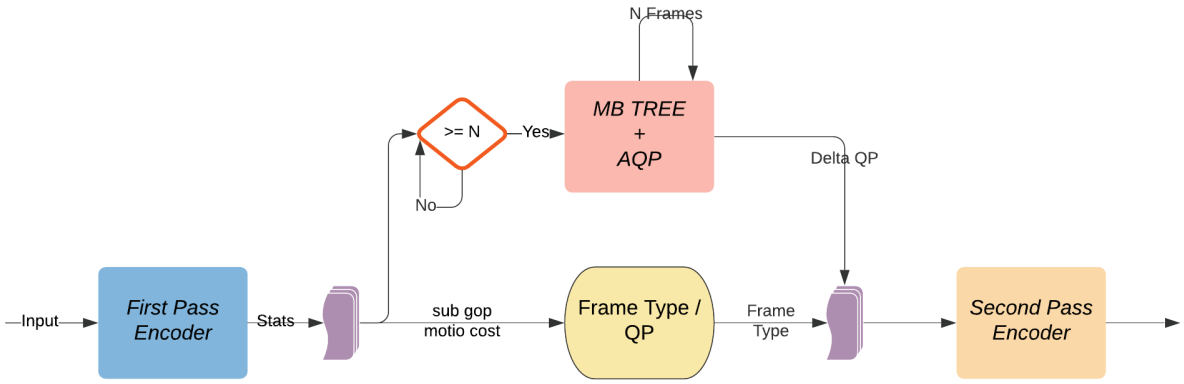

Figure 8. Encoder frame level control

MSVP VP9 uses a similar strategy, though statistics used and control decisions [9] are adapted to VP9 codec. In addition, VP9 also supports temporal filtering of the anchor frames in a sub-GOP (Alt-ref) as an extra step in the lookahead buffer. This is proven to be very useful for improving the coding efficiency as ARF is used as reference for many frames. Each of the passes (first pass, ARNR, encoding) is done by time multiplexing the HW. One of the key challenges in MSVP is to get close to full utilization of HW by efficient design of pipelining and processing in control CPU.

5. QUALITY METRIC

Video quality metrics need to be computed many times as basis/hints to select the best encoding resolution/quality level for playback to a given client. MSVP transcoding pipeline creates N encodings at different resolutions for each uploaded video. Quality metrics are then computed for each encoded video at M viewport resolutions with respect to the original video. This means that quality scores are computed N by M times for each video which makes it compute-intensive and sometimes could become the bottleneck of the whole pipeline. MSVP contains a dedicated IP block called QM, which accelerates multiple Human-Visual-System (HVS) based quality metrics. A top-level architecture diagram of QM is shown in Figure 9, and it has the following important features and characteristics:

- Accelerates the following 5 metrics: SSIM, Multi-Scale SSIM, VIF, PSNR (luma/chroma) and No-ref blurriness;
- Includes inline scalers for viewport scaling which avoids extra memory IO if external scalers are used;
- Performs unified low-pass filtering and image pyramid processing for SSIM, MS-SSIM and VIF;
- Contains HW-friendly implementation of Log function;
- Uses high precision fixed point representation instead of float;
- Includes additional numerical stability guards.

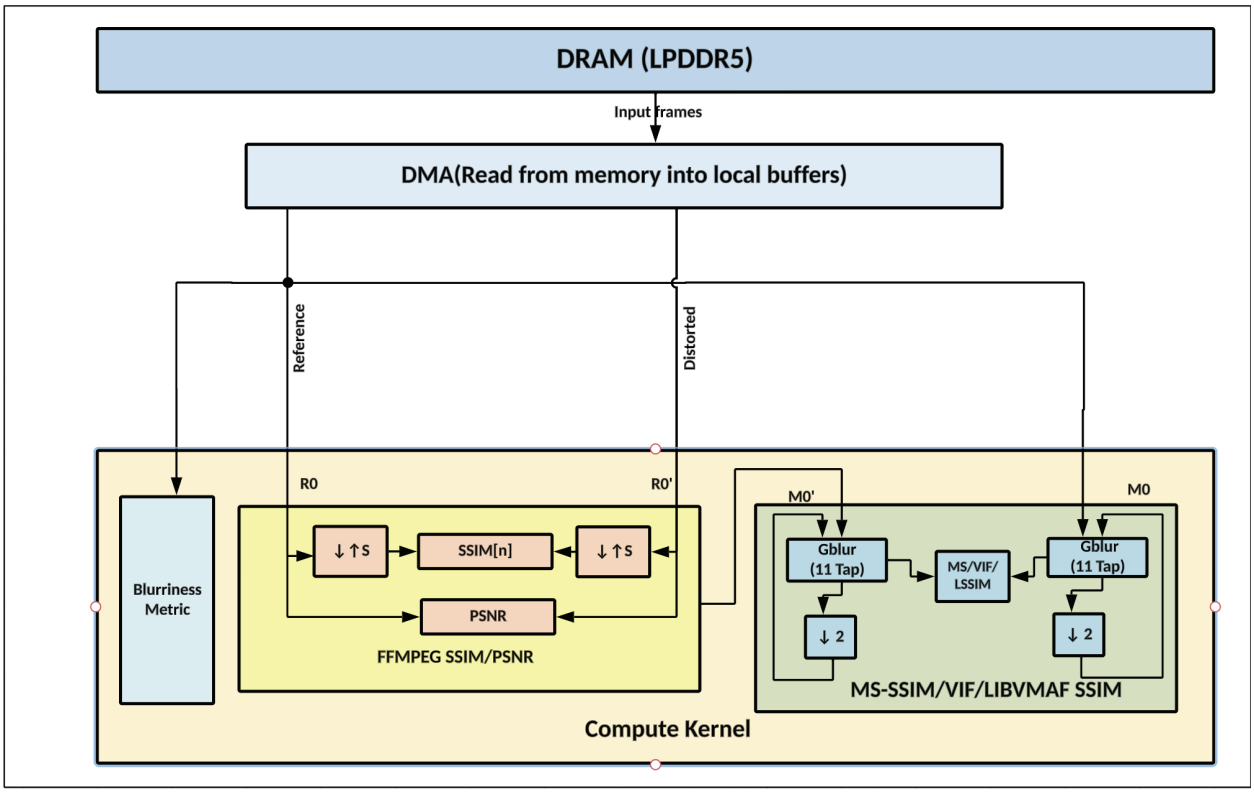

Figure 9. Quality Metric module in MSVP

To validate that all HW-friendly adaptations in the MSVP QM still yield valid and accurate quality scores, we used images and videos in 4 commonly used subjectively evaluated datasets to compare the SSIM scores produced by MSVP and SW implementations in the popular ffmpeg package. We then compute the Spearman Rank Order Correlation Coefficient (SROCC) values between the computed SSIM scores and the human subjective evaluation score. The higher the correlation, the better or more accurate the metric is. It can be seen from Table 1 that the SSIM scores produced by MSVP QM HW have comparable correlation to subjective scores as the SW ffmpeg version.

Table 1. SROCC of SSIM (Produced by SW or HW) compared with Subjective Scores

	LIVE IQA	TID 2013	LIVE VQA	Netflix Public
ffmpeg	0.9305	0.6572	0.5991	0.6567
MSVP QM	0.9315	0.6674	0.6237	0.6628

To further show that HW scores approximate SW scores, Table 2 shows the difference between HW and SW scores as measured by other statistics such as RMSE, difference of means, difference in standard deviations, and 99th percentile difference. The differences are very small as measured by all these statistics. These data showed that we can safely replace the SW quality scores with HW scores in the pipeline, saving many cycles of CPU time.

Table 2. Difference between SW and HW QM Scores

	LIVE IQA	TID 2013	LIVE VQA	Netflix Public
RMSE	0.0073	0.0072	0.0055	0.0047
Diff. of Means	0.0045	0.0041	0.0046	0.0035
Diff. of Stds	0.0018	0.0024	0.0020	0.0013
99th %ile Diff.	0.0216	0.0208	0.0129	0.0103

6. QUALITY AND PERFORMANCE BENCHMARKING

In this section we present the quality and performance of MSVP against SW encoders. We used User Generated Contents (UGC) consisting of 83 1080p videos that are representative of user uploaded contents [8]. Each video is encoded in 6 resolutions and 8 QP/crfs as listed below:

Resolutions: 1920x1080, 1280x720, 854x480, 640x360, 426x240, 256x144

H.264 crf: (MSVP and libx264): 26, 28, 30, 32, 34, 36, 38, 40

VP9 (MSVP and libvpx) crf: 28, 32, 36, 40, 44, 48, 54, 58

Sweeping a wide range of QPs and resolutions is essential to take into account wide coverage of bit-rates and clients where videos get served.

6.1 Quality measurement

To follow the similar encoding operation as in Meta's production system, each input video is downscaled into 6 resolutions and encoded with 8 QP/CRF values. Then each decoded video is upscaled to the original resolution (1080p) and the quality metrics, including SSIM and VMAF [25], against the original video are computed. For each video, from a total of 48 R-D points (8 crf x 6 resolutions), a R-D curve is constructed using a convex hull [7] method explained in the next section. To measure compression efficiency difference between two encoders, BD-Rate [10] between the two convex hulls are calculated.

Similar to software encoders, there are multiple encoder presets defined in the MSVP H.264 and VP9 encoders to tradeoff encoding quality vs compute complexity. In our tests, the following presets of different encoders are tested. For MSVP H.264 and VP9 encoders, lower presets indicate higher compression efficiency and higher complexity.

Libx264: slow, medium, fast, very fast

libVP9(alt-ref 6): Speed 1, 2

MSVP H.264: Speed 1-4

MSVP VP9: Speed 1-6

SVT AV1 speed 3-13

6.2 Performance measurement

Comparing performance in production settings involves many other factors such as host allocations, SW stack optimization etc. Hence, we do throughput evaluation of MSVP against best SW encoders in a controlled environment but matching production scenario. In the experiment set up here, we run the same encoding test on the following two different platforms:

- T1 system: A dedicated CPU only platform with Intel® Xeon® Platinum 8321HC 8321HC CPU @ 1.40GHz, CPU(s): 52, Model: 85

- T15 system: MSVP platform with the same CPU as T1 system and paired with MSVP.

Total number of transcode jobs submitted are: 83 videos x 8 (crfs) x 3 (duplications) = 1992 SIMO transcodes or 11,952 (1192x6 resolutions) encoded bitstreams. Duplications of jobs are done to have good load balancing throughout the time, there by, have high average CPU utilization. The total compute time (elapsed time) of all these jobs is measured and used as the performance metric.

Figure 10 shows performance measurement set up. To fully utilize the T15 system, we use simple xargs utility to parallelize the transcode jobs to ensure a fully loaded system in both cases. In the case of MSVP, we set the number of concurrent jobs per accelerator card to be 4 and run on all the 12 accelerators attached to the CPU. This gives a total of 48 parallel transcode jobs. With a trial run, we can observe that the HW encoder core utilization is 80%-96% depending on the preset. However, utilization of the host CPU itself in the T15 system is found to be quite low (~10-15%) as much of the load is taken care of by the ASIC. Therefore, the host CPU in the T15 system can be used for other workloads, such as audio encoding, etc.

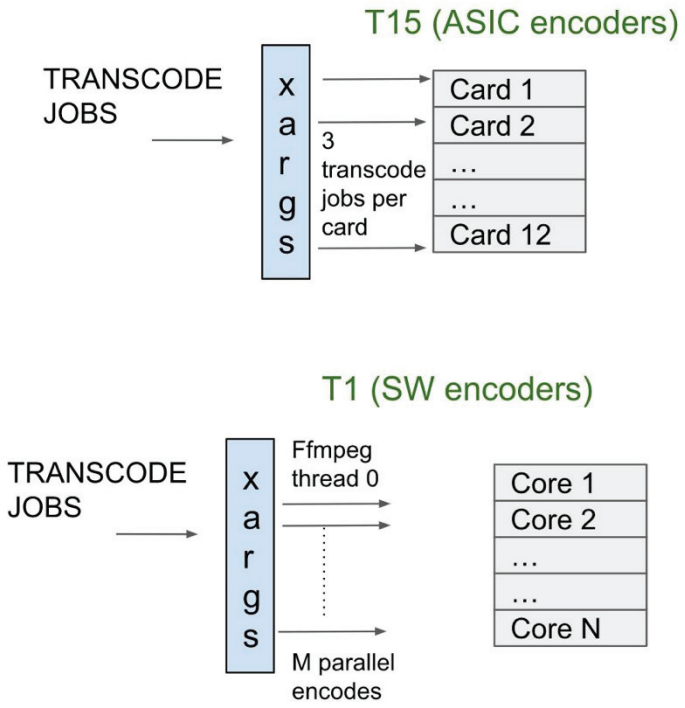

Figure 10. Performance evaluation set up of T15 and T1 systems.

As shown in Figure 10, for the T1 system, each transcode job was run in a multi-threaded execution mode, and we spawned M parallel jobs to make sure the system is fully utilized. The optimal number of parallel jobs to run (M) is selected by iterating through a few runs with values from 14 to 32 and selecting the configuration with the lowest run time. We found that the best configuration is M=16 for libx264, M=22 for libvpx. For SVT-AV1, we selected M to be between 14 and 16 depending on the speed preset given the high number of presets and wide extent of complexity. We logged the average CPU utilization, and they were all 90%+ for the best performance number points mentioned above.

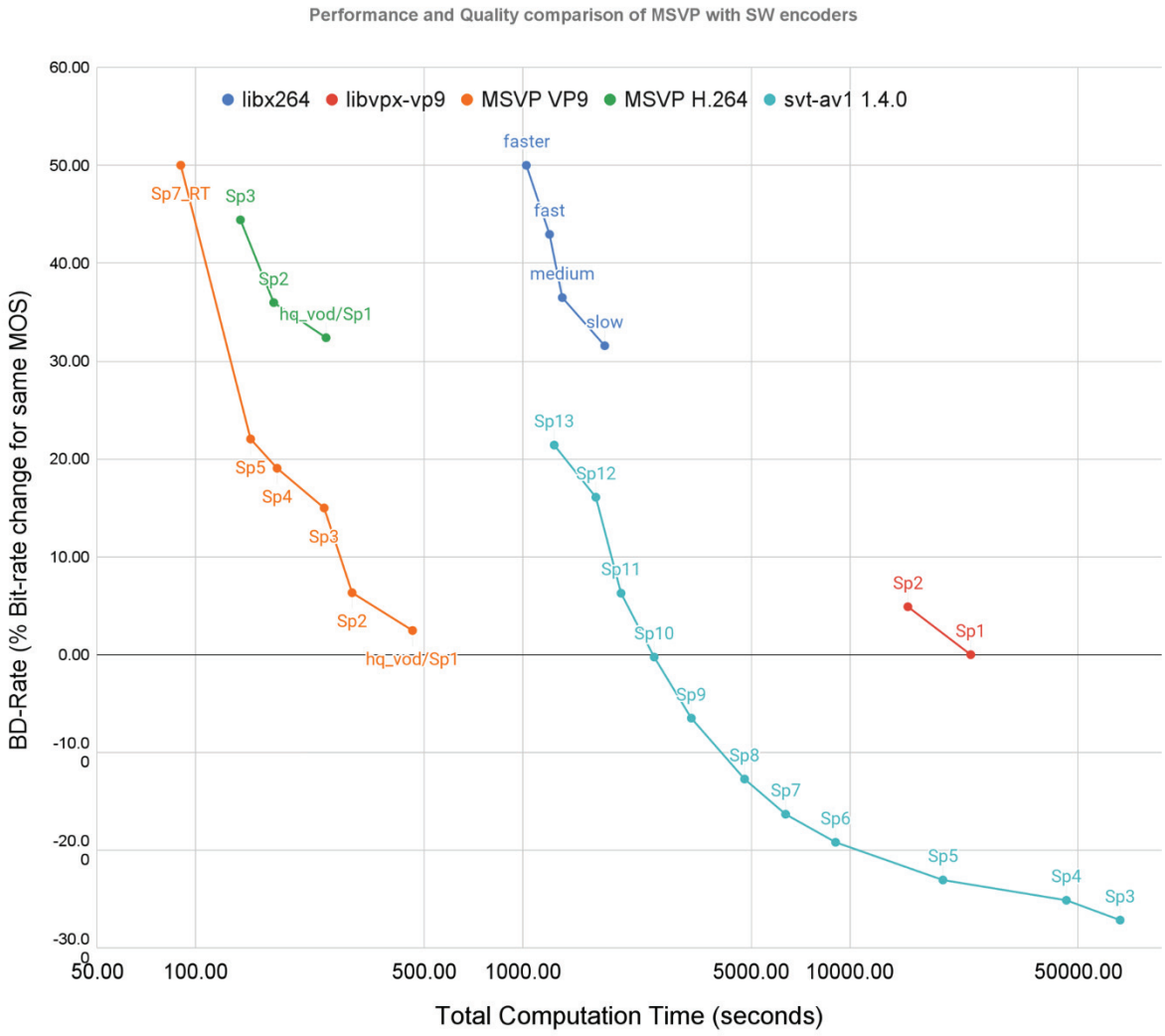

Figure 11. Performance - quality comparison

Figure 11 shows compute time vs compression efficiency comparison of few encoders used in production. X-axis is the total compute time (in log scale) of the transcode jobs, and the y-axis is the BD-rate calculated with libvp9 speed 1 as the baseline. When compared against x264 medium preset, MSVP H.264 speed 2 can achieve similar quality, but with 7.8x speed up. MSVP VP9 Speed 2 can achieve similar quality as libvpx speed 2 with 50x speed up. When compared against SVT-AV1 v1.4 release, MSVP VP9 encoder can reach the similar quality as SVT-AV1 preset 11 with 3.8x speed up. Overall, MSVP H.264 and VP9 encoders can achieve a similar quality level as software encoders with much higher throughput.

7. MSVP IN PRODUCTION AT SCALE

In Meta's video production and delivery system, only a relatively small percentage of uploaded videos account for the majority of the watch time. Therefore, we have to use different encoding configurations to process videos with different popularity to optimize the compression efficiency and compute efficiency at scale. At the system level, there is a benefit cost model that predicts the watch time for every uploaded video and then controls what encoding configuration will be triggered based on that [17]. Once a video is uploaded, it would be processed by the backend production and delivery

system. Typically, multiple Adaptive Bit Rate (ABR) encoding families are produced with different codecs and configurations. For every uploaded video, a basic ABR encoding family would be produced first, usually with H.264 encoder at high-speed preset. The goal is to quickly encode the video with good quality so the users can access and share the videos quickly. Therefore, the latency and overall efficiency are the key metrics we need to optimize for, as it is applied for all uploaded videos. Once the video gets sufficiently high watch time, more advanced ABR encodings with a dynamic ABR ladder using advanced codecs such as VP9 or AV1 encoders to further improve video quality [18].

MVSP has been deployed in Meta's production system to first replace the basic ABR encoding family that previously used software encoder. For producing encoding at upload time, latency, reliability, and efficiency are very important metrics we need to optimize. With software encoders, if we want to provide better compression efficiency, we would have to sacrifice compute efficiency as well as latency. This is not the case with MSVP. In terms of latency, we observed that MSVP is consistently better compared to software encoders across different types of products. The reduction in publishing latency enabled our users to share their videos much quicker. Quality wise, with MSVP, we are able to produce 1080p encoding for all videos at the upload time, without sacrificing quality or efficiency. Reliability wise, MSVP met or exceeded our requirements for system reliability, especially for handling large file uploads.

MSVP is also used in the Advanced ABR encoding. At a high level, the Advanced ABR family is a convex hull based dynamic optimization approach. Let's use the example in Figure 12 to go through the basic process. For an uploaded video, we will first produce multiple down-scaled versions and then encode each version with multiple QP/CRFs. For example, for a 1080p video, we could have 7 resolutions and 5 CRFs, for a total of 35 encodings. After these 35 encodings are completed, we could produce a set of Rate-Distortion (RD) curves (dash curve in the chart) as shown in the following graph. In this graph, the x-axis is the encoding bitrate and the y-axis is the quality score, expressed in MOS [5] units on a scale [0-100]. After encoding, decoded videos will be first upscaled to the original resolution and quality score will be calculated.

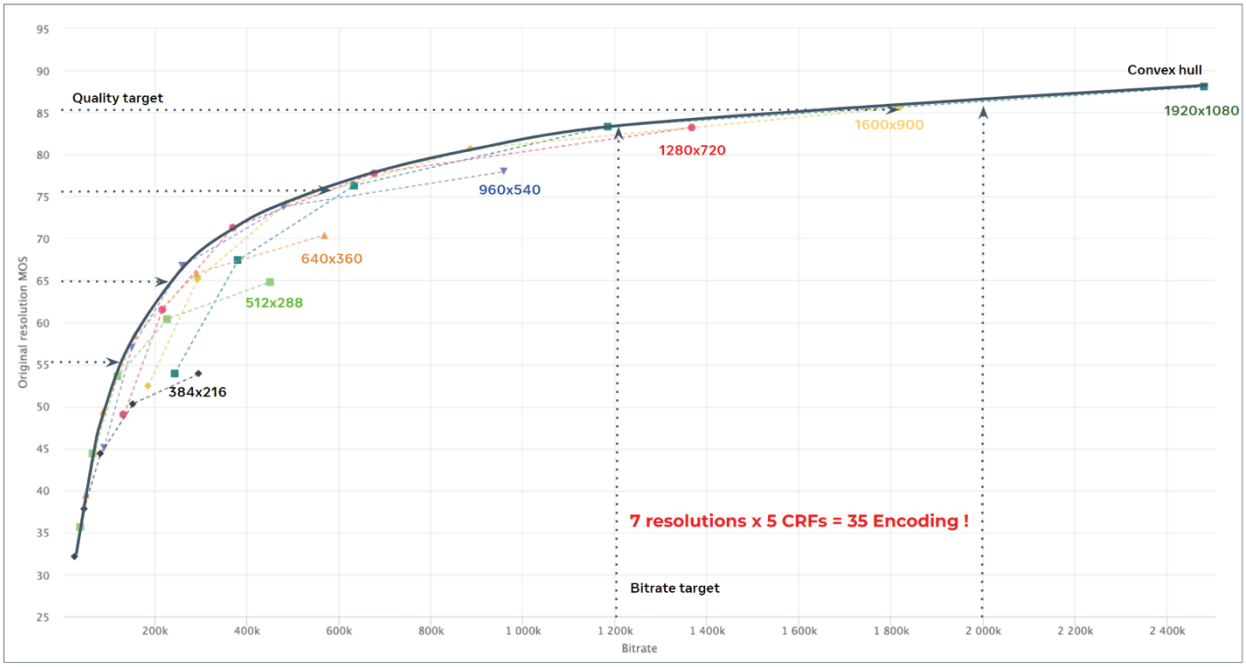

Figure 12. Convex Hull Optimization

From these 35 RD points, we could calculate the convex hull, which is a curve that connects the RD points on the upper left boundary (solid black curve). Once the convex hull is generated, we can then select the best encoding result for delivery. This selection process can be based on either the quality target or bit rate target for delivery as illustrated in Figure 12.

The complexity of this process is very high, and we have to find options to significantly reduce it. Based on previous studies, we found that we could use a high speed preset to perform the encoding (Level 1/L1) and produce the convex

hull, and then use a high quality preset to encode at the selected (resolution, CRF) points in the second pass as final delivery (Level 2/L2). Even though we have to do some additional encodings (L1 & L2) with this approach, because the L1 can be done much faster, the total encoding time for each video is actually reduced significantly. The coding efficiency drop with this approach is very small. With some further study, we found that this approach can also be applied, when the L1 and L2 use different encoders [19]. For example, we can use AVC or VP9 for the L1 encoding, and use AV1 in the L2 encoding. Furthermore, we can offload the L1 encoding to the MSVP HW encoder completely to further speed up the Advanced ABR encoding process.

We use a two-stage hybrid HW/SW approach for the advanced ABR encoding family. In this approach, MSVP fast preset encoding is used as L1 to compute the quality and bit rate information. This information is then used to calculate the convex hull curve and find the CRF and resolution for L2 encoding according to target bitrate or quality.

When we start to use MSVP for the L1 encoding in the Advanced ABR family, with actual production videos, we have observed significant compression efficiency improvement over the hardware H.264 encoder in the previous generation of ASIC. For MSVP H.264 encoder, about 20% BD-RATE improvement is achieved. For MSVP VP9 encoder, about 40% BD-RATE improvement is achieved. With improved compression efficiency, we can leverage the MSVP VP9 encoder for the L1 stage encoding and replace the previously used H.264 encoder. Moreover, we can use a faster preset of the software SVT-AV1 encoder at L2 encoding stage to reduce overall complexity of the advanced ABR encoding family at similar quality, so it can be applied to more videos.

Besides VOD usage, we are also exploring the possibility of using MSVP for Live and Broadcasting usages.

8. FUTURE WORK

While we are ramping up the MVSP usage for existing production use cases, we're also continuing to work on further improving video quality with MSVP by adding more codecs, using preprocessing methods such as noise reduction, compression artifact removal, and super resolution, etc. In the future, MSVP will allow us to support more Meta's most important use cases and needs, including short-form videos, live broadcasting, generative AI, AR/VR, and other metaverse content.

9. ACKNOWLEDGMENTS

This project has been the culmination of dedicated and enthusiastic work of many talented teams and individuals. While it is impossible to mention every team and every person here, the authors would like to specially thank the following teams: Infra Silicon, Video Infra Systems, Platform software, Emulation, Release to Production (RTP), Sourcing and Operations Engineering (SOE) and Hardware Platform. Authors would also like to thank engineers who have contributed while they were with Meta: Y. He, Y. Wang, E. Song, S. Somasundaram, S. Zhao, C. Zhao, S. Kumar, S. Venkatesan, Z. Shahid.

REFERENCES

[1] https://www.intel.com/content/www/us/en/newsroom/news/introducing-intel-data-center-gpu-flex-series.html
[2] https://developer.nvidia.com/video-codec-sdk
[3] https://www.amd.com/en/newsroom/press-releases/2023-4-6--amd-launches-first-5nm-asic-based-media-accelerat.html
[4] Ranganathan, P., Stodolsky, D., Calow, J., Dorfman , J., Guevara, M., Smullen IV, C.W., Kuusela, A., Balasubramanian, R., Bhatia, S., Chauhan, P. et al., "Warehouse-scale video acceleration: co-design and deployment in the wild," in Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 600–615 (2021).
[5] Regunathan, S. L., Wang, H., Zhang, Y., Liu, Y., Wolstencroft, D., Reddy, S., Stejerean, C., Gandhi, S., Chen, M., Sethi, P., Puntambekar, A., Coward, M., Katsavounidis, I., "Efficient Measurement of Quality at Scale in Facebook Video Ecosystem," Proc. SPIE 11510, 69-80 (2020).

[6] Mukherjee, D. et. al., "A Technical Overview of VP9 – The Latest Open-Source Video Codec," SMPTE Annual Technical Conference & Exhibition, 1-17 (2013).
[7] https://netflixtechblog.com/per-title-encode-optimization-7e99442b62a2
[8] Ling, S., Baveye, Y., Callet, P., Skinner, J., and Katsavounidis, I., "Towards Perceptually-Optimized Compression Of User Generated Content (UGC): Prediction Of UGC Rate-Distortion Category," in IEEE International Conference on Multimedia and Expo (ICME), London, United Kingdom, 1-6 (2020).
[9] Chaudhari, G., Chuang, H.-C., Koba, I., and Lalgudi, H., "Rate Estimation Techniques for Encoder Parallelization," IEEE International Conference on Image Processing (ICIP), Anchorage, AK, USA, 2059-2063 (2021).
[10] Bjøntegaard, G., "Calculation of average PSNR differences between RD-curves (VCEG-M33)," in VCEG Meeting (ITU-T SG16 Q. 6) (2001).
[11] Wang et al.. "Hardware pipelines for rate–distortion optimization (RDO) that support multiple codecs," U.S. Patent No. 11,683,498 B2 (2023).
[12] Chaudhari, G. et. al., "Architecture for rate estimation in video coding," U.S. Patent No. 11,368,694 B1 (2022).
[13] VP9 Bitstream and Decoding Process Specification, https://www.webmproject.org/vp9/
[14] "https://github.com/Netflix/vmaf/blob/master/resource/doc/datasets.md" - Netflix Public Dataset
[15] 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).
[16] Wang, Z., Bovik, A.C., Sheikh, H.R., and Simoncelli, E.P., "Image quality assessment: from error visibility to structural similarity," in IEEE Transactions on Image Processing, vol. 13, no. 4, 600-612, (2004).
[17] https://engineering.fb.com/2021/04/05/video-engineering/how-facebook-encodes-your-videos/
[18] https://engineering.fb.com/2023/02/21/video-engineering/av1-codec-facebook-instagram-reels/
[19] Wu, P.-H., Kondratenko, V., Chaudhari, G., and Katsavounidis, I., "Encoding Parameters Prediction for Convex Hull Video Encoding," Picture Coding Symposium (PCS), Bristol, United Kingdom, 1-5, (2021).
[20] libvpx, code repository - open-source VP9 encoder/decoder software," link: https://github.com/webmproject/libvpx.
[21] Mukherjee, D. et. al., "A Technical Overview of VP9--the Latest Open-Source Video Codec," SMPTE Motion Imaging Journal, Vol. 124, 44-54 (2015).
[22] M.2 Accelerator Module Hardware Specification V1.0
[23] Chaudhari, G., Koba, I., Katsavounidis, I., and Reddy, H., "Machine-learning-based tuning of encoding parameters for UGC video coding optimizations," Proc. SPIE 11842, 43-51 (2021).
[24] Sheikh, H. R., & Bovik, A. C. "Image information and visual quality," IEEE Transactions on image processing, 15(2), 430-444 (2006).
[25] Blog, Netflix Technology (2016-06-06). "Toward A Practical Perceptual Video Quality Metric," Netflix TechBlog. Retrieved 2017-07-15