Detailed Comparative Analysis of VP8 and H.264

Yousef O. Sharrab and Nabil J. Sarhan
Electrical and Computer Engineering Department & Wayne State Media Research Lab
Wayne State University
Detroit, MI 48202
Email:{yousef.sharrab, nabil}@wayne.edu

***Abstract*—VP8 has recently been offered by Google as an open video compression format in attempt to compete with the widely used H.264 video compression standard. This paper describes the major differences between VP8 and H.264 and provides detailed comparative evaluations through extensive experiments. We use 29 raw video sequences, offering a wide spectrum of resolutions and content characteristics, with the resolution ranging from 176x144 (QCIF) to 3840x2160 (2160p). To ensure a fair study, we use 3 coding presets in H.264, each with three types of tuning, and 7 presets in VP8. The presets cover a variety of achieved quality or complexity levels. The performance metrics include accuracy of bitrate handling, encoding speed, decoding speed, and perceptual video quality.**

***Keywords*-Comparative Analysis, Decoding Speed, Encoding Speed, H.264, Perceptual Video Quality, Video Codecs, Video Compression, VP8.**

I. Introduction

The H.264 video compression standard currently enjoys a great support by video websites, applications, and hardware platforms and devices, including TVs, smart phones, and digital cameras. In addition, it has numerous popular implementations, including JM, X264, and FFmpeg. The widespread of H.264 has recently faced a great challenge with Google releasing VP8 as royalty-free open video compression format in an attempt to compete with H.264 and gradually replace its usage on YouTube [1]. The effectiveness of VP8 compared with H.264 will be one of the deciding factors that will determine whether VP8 will be the video compression of choice in the future. With the great overreach of video compression, it is imperative to rigorously evaluate the effectiveness of VP8 and compare it with H.264.

Unfortunately, only little work compared the effectiveness of H.264 and VP8 [2], [3]. That work is also highly limited. A commercial report [2] has recently compared the performance of VP8 and H.264. Since most of the used video sequences were previously compressed using other codecs, they cannot be used reliably to draw any conclusions because of the inevitable bias introduced in re-compression tests. Study [3] has compared VP8 and H.264 in terms of only video quality, using only one metric, only three CIF sequences (with 352x288 resolution), and basic encoding settings.

This paper describes the major differences between VP8 and H.264 in features and operation. It also provides detailed comparative evaluations by more than $1,300$ experiments, so as to reflect real-life situations by carefully choosing the encoding parameters, the video test sequences, and the proper metrics. We use 29 raw video sequences, offering a wide spectrum of resolutions and content characteristics, with the resolution ranging from 176x144 (QCIF) to 3840x2160 (2160p) and the content varying greatly in the level of detail and motion speeds. To ensure a fair study, we use 3 coding presets in H.264, each with three types of metric tuning, and 7 presets in VP8. These presets cover a variety of achieved quality or complexity levels. The bitrate for each sequence is varied in a wide range that is suitable for that sequence. For H.264, we use X264 , which is the best implementation, according to the results in [2]. The performance metrics include perceptual video quality, encoding speed, decoding speed, and accuracy of bitrate handling. For perceptual video quality, we use two metrics: *Peak Signal-to-Noise Ratio* (PSNR) and the *Structural SIMilarity index* (SSIM) [4]. The experiments with metric tuning in H.264 are "No Tuning", "SSIM Tuning", and "PSNR Tuning". Although we discuss all the results, the conclusions are based on "No Tuning" as the other two options may be unfair to VP8.

The rest of the paper is organized as follows. Section II provides preliminary analysis of H.264 and VP8. Subsequently, Section III discusses the performance evaluation methodology. Finally, Section IV presents and analyzes the main results.

II. Preliminary Analysis

Video data contains spatial and temporal redundancy. Therefore, similarities can be encoded by just considering differences (residuals) within a frame. The first frame of a sequence or a random access point is typically intra-coded. Each block of pixels in an intra-frame is predicted using previously-encoded neighboring blocks. For all remaining frames of a sequence or between random access points, inter-coding is usually used, employing block motion compensation to predict blocks from other previously decoded frames. The residuals of the intra and inter-prediction are then transformed to the frequency domain using the Integer Discrete Cosine Transform (Integer DCT). Subsequently, the transform coefficients are quantized, thereby reducing the overall precision of the coefficients and possibly eliminating high frequency coefficients. The quantized transform coefficients are entropy coded and transmitted together with any possible motion vectors.

In the *YUV* colorspace, each pixel is represented by three components: *Y*, *U*, and *V*. The Y component determines

This work was supported in part by U.S. NSF grant CNS-0834537.

978-0-7695-4875-3/12 $26.00 © 2012 IEEE
DOI 10.1109/ISM.2012.33

TABLE I
COMPARING H.264 AND VP8 CHARACTERISTICS

Encoding Step	H.264	VP8
Intra-Prediction	Uses 9 prediction modes for each 4×4 Luma block and for each 8×8 Luma block in high profiles prediction. Uses 4 prediction modes for a 16×16 Luma block and 8×8 Chroma prediction modes	Uses 4 common intra-prediction modes which shared by these macroblocks: 4×4 Luma, 16×16 Luma, and 8×8 Chroma.
Inter-Prediction	Supports up to 16 reference frames	Supports up to three reference frames
	Partition types are 16×16, 16×8, 8×16 and each 8×8 can be 8×8, 8×4, 4×8, or 4×4	Partition types are 16×16, 16×8, 8×16, 8×8, 4×4
	Chroma motion vector are calculated directly from Luma motion vectors	Chroma motion vectors are calculated from Luma vectors by averaging the motion vectors within a macroblock
	Interpolation filter uses qpel, six-tap Luma and bilinear Chroma	Interpolation filter uses qpel, six-tap Luma, and mixed four/six-tap Chroma
	Supports B-frames	Does not supports B-frames
Transform	Uses integer DCT	Uses DCT
Quantization	Has a built-in adaptive quantization	Adaptive quantization is not a core feature
Entropy Coding	Uses an adaptive arithmetic coder	Uses non-adaptive arithmetic coder
Loop Filter	Its complexity is between VP8's fast and normal modes	Has two modes: fast and normal

the brightness, which is referred to as *Luminance* or *Luma*, whereas the U and V components determine the color itself, referred to as *Chrominance* or *Chroma*. Since the human eye is more sensitive to luminance, the Chroma component can be sampled at a much lower rate.

An H.264 encoder can choose from many different intra and inter modes when coding a macroblock. The *Rate-Distortion Optimization* (RDO) mode selection is an algorithm for choosing the best coding mode for each macroblock, based on the bitrate and distortion cost. To select the best encoding mode for a macroblock, RDO algorithm examines all possible combinations.

The rest of the section compares the characteristics of H.264 and VP8.

Intra-Prediction

Intra-prediction is used to predict the content of a block from its surroundings without referring to other frames. With H.264, 9 modes are possible for intra-prediction of each 4×4 Luma block and for each 8×8 Luma block: DC mode and 8 directional modes. In DC mode, each block is predicted by the mean of the pixel samples in the upper row and left column. Directional modes include Vertical, Horizontal, and 6 other angular modes, whereby the samples are extrapolated vertically, horizontally or with a specified angle, respectively. (The Luma component with an 8×8 block size is only available in the high profile). In addition, 4 prediction modes are available for each 16×16 Luma block (used for regions in the frame with less spatial details), and for each 8×8 Chroma block. These modes are DC, Horizontal, Vertical, and Planar. Planar is a linear plane function fitted to the upper and left-hand samples.

Four intra-prediction modes of VP8 are used for each one of the three types of macroblocks: 4×4 Luma, 16×16 Luma, and 8×8 Chroma. These modes are DC (DC_PRED), Horizontal (H_PRED), Vertical (V_PRED), and TrueMotion (TM_PRED) prediction. TM_PRED prediction is similar to Planar in H.264. In intra-mode selection in H.264, the number of mode combinations for one 16×16 pixel MacroBlock (MB) without considering the high profile is $N8 \times (16 \times N4 + N16)$, where $N8$, $N4$, and $N16$ represent the number of modes of an 8×8 Chroma block, a 4×4 Luma block, and a 16×16 Luma block, respectively. To select the best mode for one MB in intra-prediction, the encoder performs $4 \times (16 \times 9 + 4) = 592$ RDO calculations [5]. For VP8, applying the above equation yields $4 \times (16 \times 4 + 4) = 272$ RDO calculations. Therefore, the complexity of VP8 intra-prediction is less than half that of H.264 without optimization.

Inter-Prediction

Inter-prediction is the process of estimating the content of a block by referring to encoded frames. The main components of inter-prediction are reference frames and motion vectors. The reference frame is a previously encoded frame used to predict blocks in another frame, whereas the motion vectors indicate the distance between the position of the current block in a frame and the corresponding prediction block in the reference frame [6].

The distortion measure *sum of absolute differences* (SAD) is commonly used to find the best matching block for the current block in the frame to be encoded from the blocks of previously encoded reference frames. $SAD(V_x, V_y)$ is defined as the SAD for block A of size $P \times Q$ located at (x, y) inside the current frame compared to block B located at a displacement of (V_x,V_y) relative to block A in the reference frame. It can be found by summing the absolute differences between each pixel in block A and the corresponding pixel in block B. In the Full Search (FS) algorithm, if a maximum displacement of S pixels in a frame is allowed, $(2.S + 1)^2$ locations have to be searched to find the best match for the current block in the reference frame.

VP8 supports up to 3 reference frames: previous frame, alternative reference frame, and golden frame. In contrast, H.264 support up to 16 reference frames. Study [7] demonstrate by experiments that using more than three reference frames is exceedingly unlikely to provide a significant benefit in perceptual quality while significantly increasing the implementation complexity and power consumption.

VP8 and H.264 have similar partitioning structures. VP8 uses the following pixel partition types: 16×16, 16×8, 8×16, 8×8, and 4×4. By contrast, H.264 uses 16×16, 16×8, and 8×16, and each 8×8 can be further divided into 8×8, 8×4, 4×8, or 4×4. Dropping 8×4 and 4×8 partitions in VP8 is unlikely to be of a significant consequence [7], [6].

For motion vectors, both VP8 and H.264 support variable-size partitions. Both VP8 and H.264 use quarter-pixel precision

TABLE III
CHARACTERISTICS OF STANDARD SEQUENCES CATEGORIES

Format	Resolutions	Category	bitrate (Kbps)
QCIF	176x144	Video Conferences	20-800
CIF	352x288		
4CIF	704x576	SDTV	100-2000
720p	1280x720	HDTV	500-3000
1080p	1920x1080		
2160p	3840x2160	Quad HDTV	2000-8000

(Qpel) motion vectors with a six-tap interpolation filter for Luma pixels. Qpel uses interpolation methods to detect a motion more accurately in a video frame by increasing the precision to $1/4$ of a pixel. H.264 predicts the Chroma motion vectors directly from the Luma vectors, whereas VP8 predicts them from Luma vectors by averaging the motion vectors within a macroblock. A further distinction is that H.264 uses the simpler bilinear prediction for Chroma, whereas VP8 uses a mixed four/six-tap interpolation.

H.264 supports three types of frames: I-frames (intra-coded frames), P-frames (predictive inter-frames), and B-frames (bi-predictive inter-frames). P-frames are predicted from previous frames, whereas B-frames are predicted from both previous and future frames, and I-frames are predicted only using intra-prediction. B-frames can achieve up to 20% compression benefit at the expense of implementation complexity [7], [6]. VP8 does not support B-frames, but tries to compensate for that by the intelligent use of both the golden and the alternate reference frames [7].

Transformation and Quantization

H.264 uses integer DCT. This transform results in lower compression rates, but greatly simplifies the transform process, which can be implemented by using only additions, subtractions, and right shifts. VP8 uses an accurate version of DCT, which requires many multiplication operations.

For quantization, H.264 use a built-in adaptive quantization algorithm that performs macroblock-level quantization to improve video quality. In contrast, VP8 does not use macroblock-level quantization [6].

Entropy Coding

Entropy coding is the process of lossless compression of all the information from all the other encoding processes, such as DCT coefficients, prediction modes, and motion vectors. The arithmetic coder are similar for both VP8 and H.264, but H.264 uses an adaptive arithmetic coder, whereas VP8 uses a non-adaptive arithmetic coder [6].

Loop Filter

The loop filter is used to smooth the block edges. VP8 and H.264 use similar loop filters, but the filter in VP8 can be configured in fast mode or a normal mode. The complexity of the filter in H.264 is between the fast and normal modes of VP8.

III. PERFORMANCE EVALUATION METHODOLOGY

For VP8, we use vpx codec version v0.9.5 for both encoding (using vpxenc), and decoding (using vpxdec). v0.9.0 (ivfenc for encoding and ivfdec for decoding). For H.264, we use X264 (r1688) for encoding and FFmpeg (SUN-r24758) for decoding.

TABLE V
VP8 PRESET PARAMETERS

Best	-p 2 –i420 -w 176 -h 144 –target-bitrate=18 –fpf=tmp.fpf –threads=4 –best –end-usage=0 –auto-alt-ref=1 -v –minsection-pct=5 –maxsection-pct=800 –lag-in-frames=16 –kf-min-dist=0 –kf-max-dist=999999 –token-parts=2 –static-thresh=0 –min-q=0 –max-q=63
Good0	same as best, but –good –cpu-used=0 instead of –best
Good1	same as good0, but –cpu-used=1
Good2	–cpu-used=2
Good3	–cpu-used=3
Good4	–cpu-used=4
Good5	–cpu-used=5

We divide the test sequences into four groups based on the resolution as shown in Table III. A description of each of the used sequences is shown in Table II.

The performance metrics include accuracy of perceptual video quality, encoding speed, decoding speed, and bitrate handling. Bitrate handling captures the accuracy of the encoder in achieving the desire bitrate. For perceptual video quality, we use two metrics: *Peak Signal-to-Noise Ratio* (PSNR) and the *Structural SIMilarity index* (SSIM) [4]. The $PSNR$ between an original frame A and the corresponding encoded frame B can be given as follows:

$$PSNR(dB) = 10 \times \log \frac{MAX^2}{MSE}, \quad (1)$$

where MSE and MAX represent the Mean-Square Error, and the maximum possible pixel value of the image, respectively. When each pixel is represented as 8 bits, $MAX = 255$. MSE can be given by

$$MSE = \sum_{i=1}^{n} \sum_{i=1}^{m} \frac{(A_{ij} - B_{ij})^2}{n \times m}, \quad (2)$$

where m and n represent the width of the image in pixels and the height of the image in pixels, respectively. Despite the widespread usage of PSNR, the non-linear behavior of the human visual system causes it to not capture accurately the perceptual video quality. SSIM, however, is more correlated to the quality. SSIM compares the distortion in three image components: luminance, contrast, and structure.

$$SSIM(x, y) = \frac{(2M_A M_B + C_1)(2S_{AB} + C_2)}{(M_A^2 + M_B^2 + C_1)(S_A^2 + S_B^2 + C_2)}, \quad (3)$$

where M_A, M_B, S_A, S_B, and S_{AB} are the average of A, the average of B, the variance of A, the variance of B, and the covariance of A and B, respectively. C_1, C_2 are two constants to stabilize the division with weak denominator.

Since the human eye is more responsive to brightness than to color, we only use the Luma (Y) components in the YUV colorspace to determine both PSNR and SSIM.

The raw video sequences and decoded videos are all made in the YV12 format. Videos with the y4m format are converted to the YV12 format using FFmpeg. We develop Java scripts to automate the measurements of the encoding and decoding times and the bitrate of the encoded video.

TABLE II
CHARACTERISTICS OF THE USED STANDARD VIDEO SEQUENCES

Sequence Name	Dur. (sec.)	# Frames	Resolution	Description
Forman	12	300	QCIF, CIF	A talking man face with very rich details, followed by construction
Salesman	17	449	QCIF	A sitting man engaged in moderate gestures without much movements
News	12	300	QCIF, CIF	A sitting two television announcers, with a television in the back with moving pictures
Mobile	12	300	QCIF	A moving calendar with text and a detailed photo of a ship
Highway	80	2000	QCIF	A high motion video of a highway
Stefan	3	90	CIF	A high movement player, with many viewers and much details
Paris	42	1065	CIF	A man and a woman sitting on a table and talking
MotherDaughter	12	300	CIF	A mother and her daughter speaking (similar to video conferences)
City	24	600	4CIF	A part of a city that has many crowded buildings, which move due to moving camera
Crew	24	600	4CIF	10's of men walking and waiving by their hands, most of them wear the same color
Harbour	24	600	4CIF	Moving ships in two directions on a harbour
Soccer	24	600	4CIF	A soccer game with players running in different directions
Ice	19	480	4CIF	A people skiing on the ice without much details in the background
DucksTakeOff	20	500	720p, 2160p	Ducks take off from the sea, without much details on the backgrounds or the flying ducks
InToTree	5	500	720p, 2160p	A building, a lake, and many trees, with all the scene moving due to moving camera
OldTownCross	5	500	720p, 2160p	An Old City Building with much details moving due to moving camera
ParkJoy	20	500	720p, 2160p	Seven people running on a park with much moving details
Mobcal	20	504	720p	A moving calendar with text and a detailed photo of a ship
BlueSky	8	645	1080p	A moving picture of a big tree with sky as background
PedestrianArea	15	375	1080p	Many people crossing a street in two directions
Riverbed	10	250	1080p	A river which shows the movement of a clean water where we can see the stones
RushHour	20	500	1080p	Many cars in both directions of a street, and walking people in a middle of a city
Station2	12	313	1080p	A train station with many moving trains

TABLE IV
X264 PRESET PARAMETERS, (–KEYINT 500 IS COMMON FOR ALL PRESETS)

HD Preset Parameters	QCIF, CIF, 4CIF Preset Parameters	Tuning
High Quality Parameters		
–preset slow –ref 4 –pass 1 –preset slow –ref 4 –pass 2	–preset slow –pass 1 –preset slow –pass 2	None
–preset slow –ref 4 –tune PSNR –pass 1 –preset slow –ref 4 –tune PSNR –pass 2	–preset slow –tune PSNR –pass 1 –preset slow –tune PSNR –pass 2	PSNR
–preset slow –ref 4 –tune SSIM –pass 1 –preset slow –ref 4 –tune SSIM –pass 2	–preset slow –tune SSIM –pass 1 –preset slow –tune SSIM –pass 2	SSIM
High Speed Parameters		
–preset faster –subme 3 –trellis 0 –weightp 0 –ref 1	–preset fast –ref 1	None
–preset faster –subme 3 –trellis 0 –weightp 0 –ref 1 –tune PSNR	–preset fast –tune PSNR –ref 1	PSNR
–preset faster –subme 3 –trellis 0 –weightp 0 –ref 1 –tune SSIM	–preset fast –tune SSIM –ref 1	SSIM
Normal Parameters		
–preset faster –weightp 0 –subme 3 –pass 1 –preset faster –weightp 0 –subme 3 –pass 2	–preset fast –pass 1 –weightp 0 –subme 5 –preset fast –pass 2 –weightp 0 –subme 5	None
–preset faster –weightp 0 –subme 3 –tune PSNR –pass 1 –preset faster –weightp 0 –subme 3 –tune PSNR –pass 2	–preset fast –tune PSNR –pass 1 –weightp 0 –subme 5 –preset fast –tune PSNR –pass 2 –weightp 0 –subme 5	PSNR
–preset faster –tune SSIM –weightp 0 –subme 3 –pass 1 –preset faster –tune SSIM –weightp 0 –subme 3 –pass 2	–preset fast –tune SSIM –pass 1 –weightp 0 –subme 5 –preset fast –tune SSIM –pass 2 –weightp 0 –subme 5	SSIM

The encoding parameters for both X264 and VP8 are chosen to cover high speed, normal quality, and high quality, thereby considering the tradeoff between the speed and quality. The encoding parameters also cover a range of the bitrate suitable for each sequence. The encoding parameters of X264 are chosen based on the recommendation of the developers, as described in [2]. Similarly, the VP8 parameters are chosen from the WebM website. Tables IV and V detail the values of the used parameters. The encoded videos have the extensions mp4 and WebM for H.264 and VP8, respectably. The encoding and decoding speeds are measured in the number of frames per second. The bitrate handling is measured by dividing the target bitrate (which is an input to the encoder) by the accomplished bitrate (which is the bitrate of the encoded video). The accomplished bitrate is assessed by dividing the file size in Kbits by the sequence display time in seconds. To assist in determining the PSNR and SSIM metrics, we use Matlab scripts. The following computer configuration is used for the main tests: 64-bit Windows 7 Professional, Intel Core 2 Duo CPU E7500 at 2.93 GHz and 4 GB RAM.

IV. RESULT PRESENTATION AND ANALYSIS

Let us now analyze the results of comparing VP8 and H.264. To keep the figures less crowded, not all preset results are shown in all figures. Specifically, the results of VP8 presets "Good0", "Good1", "Good3", and "Good4" are shown only in Figure 6. Furthermore, H.264 presets "High Quality PSNR Tuning", "High Speed PSNR Tuning", and "Normal PSNR Tuning" are shown only in Figure 7 and "High Quality SSIM Tuning", "High Speed SSIM Tuning", and "Normal SSIM

Tuning" are shown only in Figures 6 and 7.

Perceptual Video Quality

Figures 1 and 2 compare VP8 and H.264 in terms of PSNR and SSIM, respectively. H.264 performs better than VP8 in all resolutions up to and including 720P, but VP8 achieves better results for 1080P and 2160P. Interestingly, H.264 preset "High Quality" and VP8 'preset 'Best" (higher computation complexity) perform poorly in terms of both perceptual video quality and encoding speed (discussed next) in HD resolutions.

Encoding Speed

Figure 3 compare H.264 and VP8 in terms of encoding speed. H.264 is superior to VP8 in most resolutions. This behavior is despite the reduced feature complexity of VP8 (discussed in section II) and can be attributed to the implementation.

Decoding Speed

Figure 4 compares the decoding speeds of VP8 and H.264. VP8 and H.264 either have close performance or VP8 performs better, especially with preset "Best" at lower bitrates.

Bitrate Handling

Figure 5 compares H.264 and VP8 in terms of bitrate handling, as measured by the ratio of the target to accomplished bitrate. H.264 achieves significantly better than VP8 in bitrate handling in all resolutions higher than 4CIF. It can achieve the desired bitrates much more accurately for this wide range of resolutions. VP8 performs slightly better only in 4CIF and lower resolutions.

Encoding Speed and Bitrate Tradeoff at Fixed Perceptual Quality

Figure 6 compares the VP8 and H.264 encoding speeds and accomplished bitrates at fixed perceptual video quality. In all resolutions excluding HD, H.264 achieves higher encoding and lower bitrates than VP8 at the same quality. In HD resolutions, VP8 yields lower bitrates but lower encoding speeds than H.264 at the same quality. Comparing the presets of VP8 under QCIF, CIF, and 4CIF, preset "Best" (the highest computation complexity) achieves the best quality and the worst encoding speed, followed by presets "Good0", "Good1", "Good2", "Good3", "Good4", and "Good5". As expected, preset "Good5" (the lowest computation complexity) produces the worst quality and the highest encoding speed among all the VP8 presets. For HD resolutions, presets "Good1", "Good2", and "Good3" perform better in both the quality and encoding speed. For H.264, generally speaking, preset "Higher Quality" achieves the highest quality and the lowest encoding speed, especially for lower resolutions, whereas preset "Higher Speed" performs the worst quality and the best encoding speed, and "Normal Quality" performs in the middle for lower resolutions but the best for HD resolutions.

Impact of SSIM and PSNR Tunings in H.264

Figure 7 demonstrates the impact of metric tuning in H.264 encoding. As expected, H.264 performs better in SSIM metric when it is configured to perform SSIM tuning and performs better in the PSNR metric when it is configured with PSNR tuning. The difference is great and makes the comparisons of H.264 with VP8 unfair if H.264 is tuned. Thus, we generally assumed no such tunings in our conclusions.

V. Conclusions

We have compared the implementation complexities of VP8 and H.264, and have showed that VP8 has simpler intra and inter prediction algorithms. Encoding simplicity leads to faster encoding and lower power consumption at the encoder. In addition, we have analyzed and compared the performance of H.264 and VP8 through extensive experiments. The main results can be summarized as follows.

- In terms of perceptual video quality, H.264 performs better than VP8 in all resolutions up to and including 720P, but VP8 achieves better results for 1080P and 2160P.
- H.264 is superior to VP8 for most resolutions in terms of the encoding speed. This behavior is despite that reduced complexity of VP8 and can be attributed to the implementation.
- VP8 performs better than H.264 in decoding speed for certain resolutions.
- H.264 achieves significantly better than VP8 in bitrate handling in all resolutions higher than 4CIF. It can achieve the desired bitrates much more accurately for this wide range of resolutions.
- Surprisingly, faster encoding presets perform better in terms of both perceptual quality and encoding speed with both H.264 and VP8 under HD resolutions.

These results demonstrate that H.264 generally achieves better than VP8 in terms of perceptual video quality, encoding speed, and bitrate handling. The results in terms of encoding speed are surprising considering that VP8 seeks to reduce the implementation complexity by providing more limited features. The gap between the two in terms of encoding speed is being reduced over times due to improvements in the implementation of VP8.

References

[1] H. Wilson, "Open letter to google: free vp8, and use it on youtube," 2010-03-12.

[2] D. D. Vatolin, D. D. Kulikov, and A. Parshin, "Sixth MPEG-4 AVC/ H.264 video codecs comparison - short version."

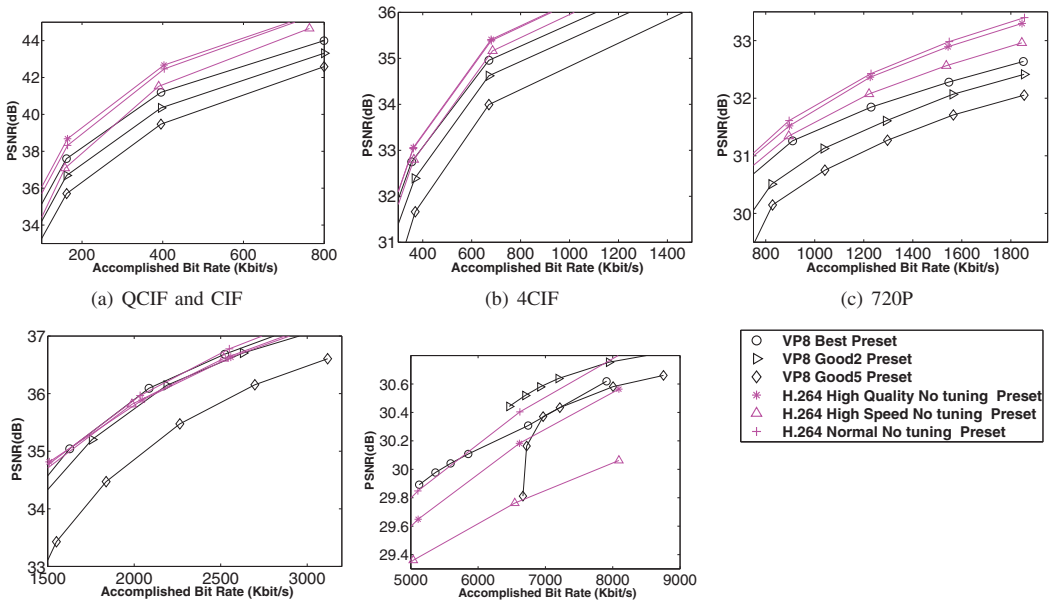
[3] P. Seeling, F. H. Fitzek, G. Ertli, A. Pulipaka, and M. Reisslein, "Video network traffic and quality comparison of VP8 and H.264 SVC," in *Proceedings of the 3rd workshop on Mobile video delivery. Conference on Human Factors in Computing Systems*, 2010.

[4] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 13, no. 4, pp. 600–612, 2004.

[5] J. Kim, D. Kim, and J. Jeong, "Complexity reduction algorithm for intra mode selection in H.264/AVC video coding." *Advanced Concepts for Intelligent Vision Systems (ACIVS'06)*, vol. 4179, pp. 454–465, 2006.

[6] J. Garrett-Glaser, "The first in-depth technical analysis of VP8, http://x264dev.multimedia.cx/archives/377."

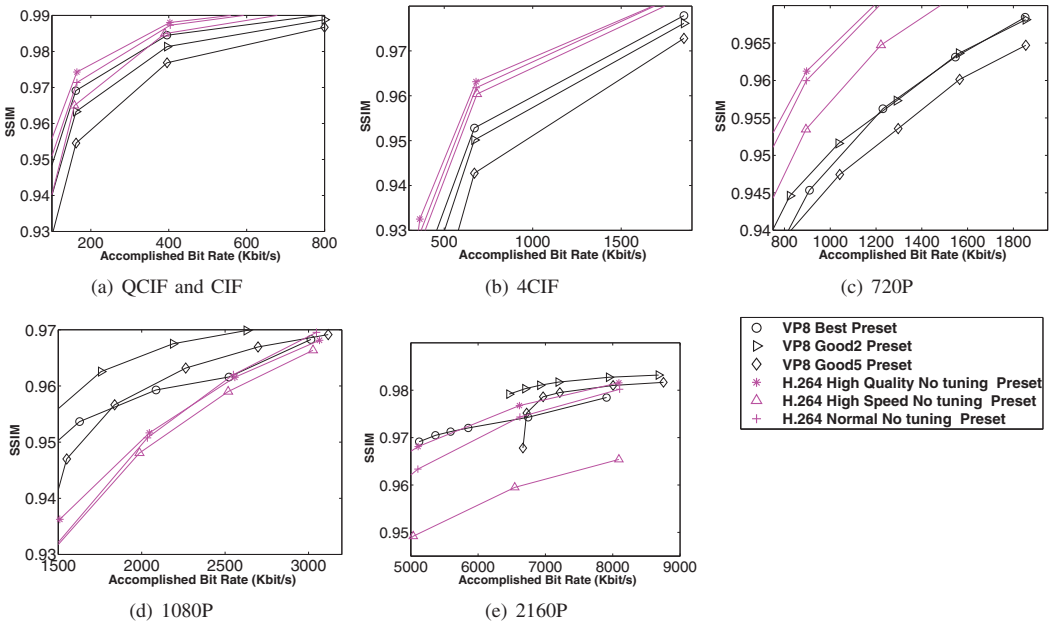
[7] J. Bankoski, P. Wilkins, and Y. Xu, "Technical overview of VP8, an open source video codec for the web."



(a) QCIF and CIF (b) 4CIF (c) 720P

(d) 1080P (e) 2160P

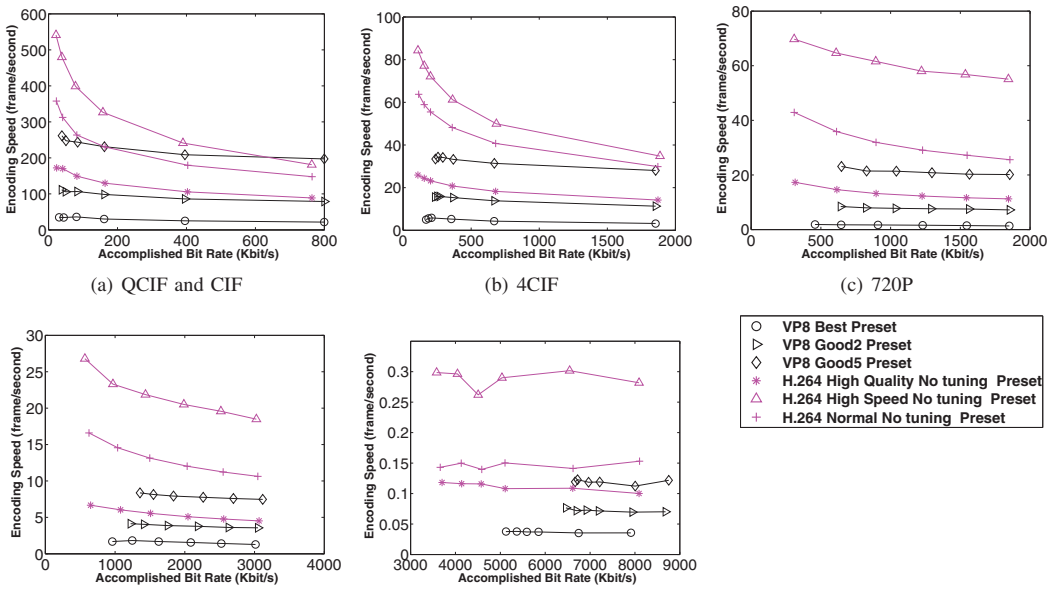
Fig. 1. Comparing VP8 and H.264 in Perception Video Quality Using the PSNR Metric



(a) QCIF and CIF (b) 4CIF (c) 720P

(d) 1080P (e) 2160P

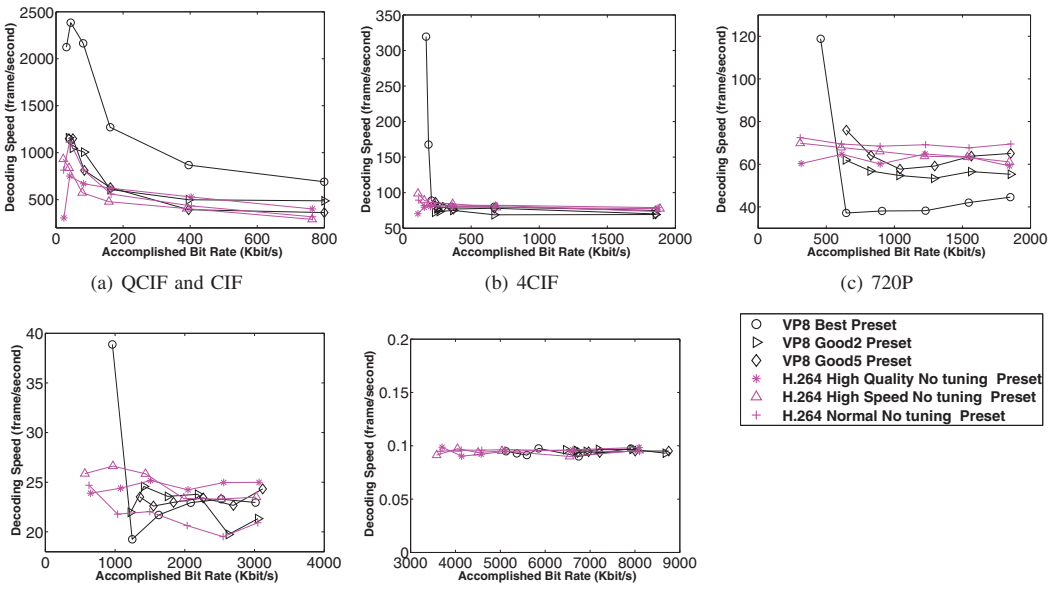
Fig. 2. Comparing VP8 and H.264 in Perception Video Quality Using the SSIM Metric



(a) QCIF and CIF (b) 4CIF (c) 720P

(d) 1080P (e) 2160P

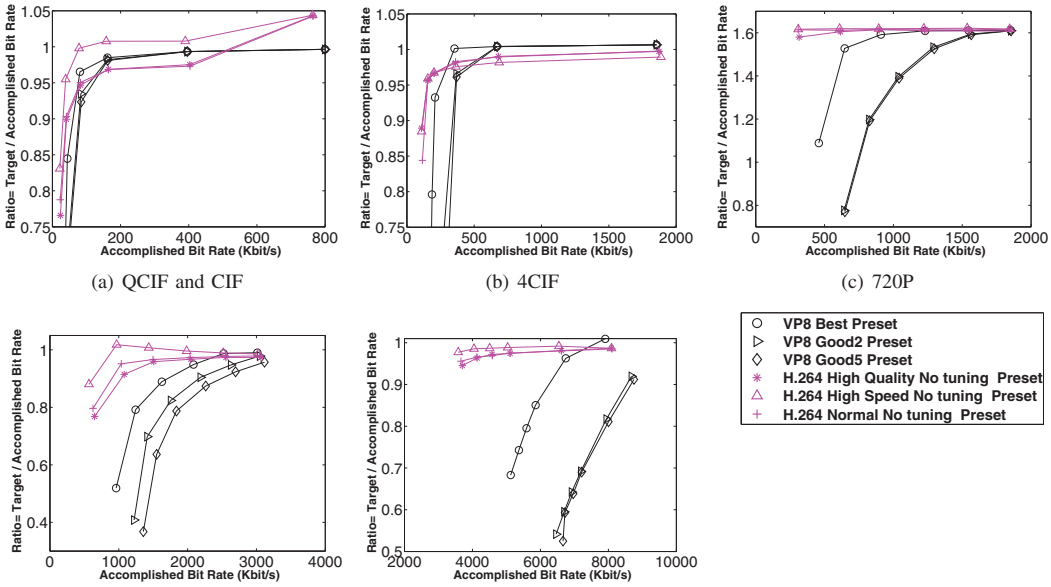
Fig. 3. Comparing VP8 and H.264 in Encoding Speed



(a) QCIF and CIF (b) 4CIF (c) 720P

(d) 1080P (e) 2160P

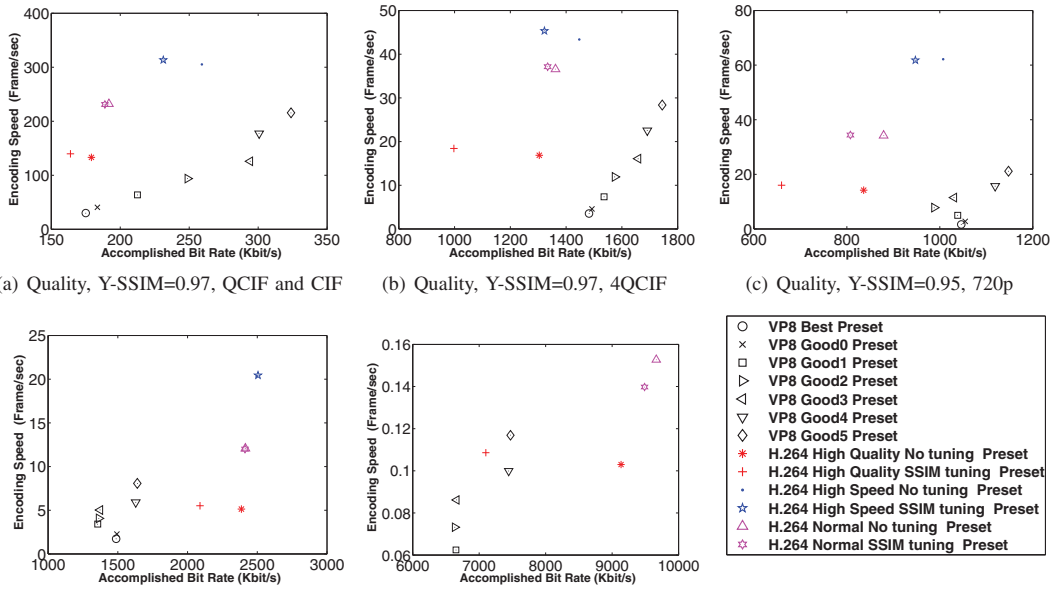
Fig. 4. Comparing VP8 and H.264 in Decoding Speed



(a) QCIF and CIF (b) 4CIF (c) 720P

(d) 1080P (e) 2160P

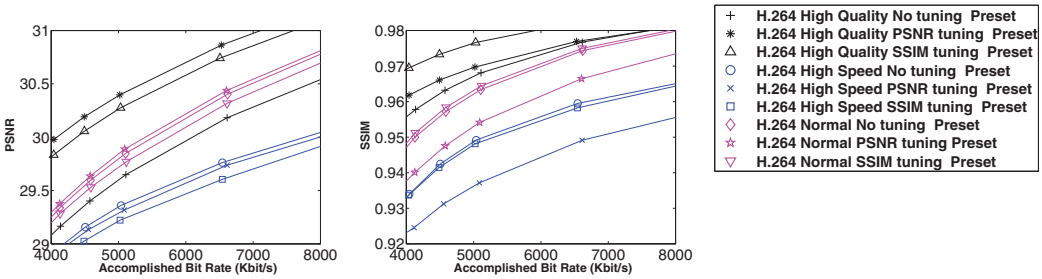
Fig. 5. Comparing VP8 and H.264 in Bitrate Handling



(a) Quality, Y-SSIM=0.97, QCIF and CIF (b) Quality, Y-SSIM=0.97, 4QCIF (c) Quality, Y-SSIM=0.95, 720p

(d) Quality, Y-SSIM=0.95, 1080p (e) Quality, Y-SSIM=0.980, 2160p

Fig. 6. Comparing VP8 and H.264 in Encoding Speed and Bitrate at Fixed Quality



(a) Quality, Y-PSNR (b) Quality, Y-SSIM

Fig. 7. Impact of Tuning Effect in H.264 [2160P]