PD20 Video Encode Quality

I run a much more intensive evaluation of each release of PD for my needs and my typical encoding streams and editing functions so I understand what I'm working with in each release for my particular needs. I've been burnt with PD way too many times. This writeup shares some of that testing with the broader forum group to show the quality of basic encoding with PD20 using the standard VMAF quality index from Netflix, https://en.wikipedia.org/wiki/Video_Multimethod_Assessment_Fusion .

This VMAF quality index, like many, is not without scrutiny but it can provide a good glimpse into the encode quality vs source video. This index does not determine the quality of the source video per say, but a comparative index on how close an encoded stream matches the source stream on a frame-by-frame basis. Each frame receives a VMAF score, for this forum simplistic needs, the composite average of all frames is used for overall quality metric of the encoded clip. A 100% VMAF would be an identical match to the source on a frame-by-frame basis, for the entire duration of the clip, something one would expect from a PD20 SVRT encoding (except the areas that are encoded per PD20's SVRT rules). A VMAF index deviation of 5% or more is often recognized as visibly inferior to the source.

Several forum contributors often claim higher equality with CPU encoding, others state encoding with a bitrate higher than source is not value added, does this VMAF metric agree with these statements?

One thing is certain, CPU encoding is typically less error prone for a real timeline that has effects, speed ratio's, PIPs, titles, transitions and such. Not that the CPU encoding quality is better, basically the encode task finished without significant hiccups or issues in the playback of the produced timeline. This is often echoed in the forums with statements of turning off hardware decoding in pref's and don't use hardware encoding on "Produce" tab. These are not necessarily issues with the hardware encoder or decoder, but perhaps more PD20's implementation.

For this simple video encode evaluation of quality using the VMAF metric, when CPU encoding is utilized, CPU decoding is also utilized. Likewise for GPU encoding, this will utilize GPU decoding.

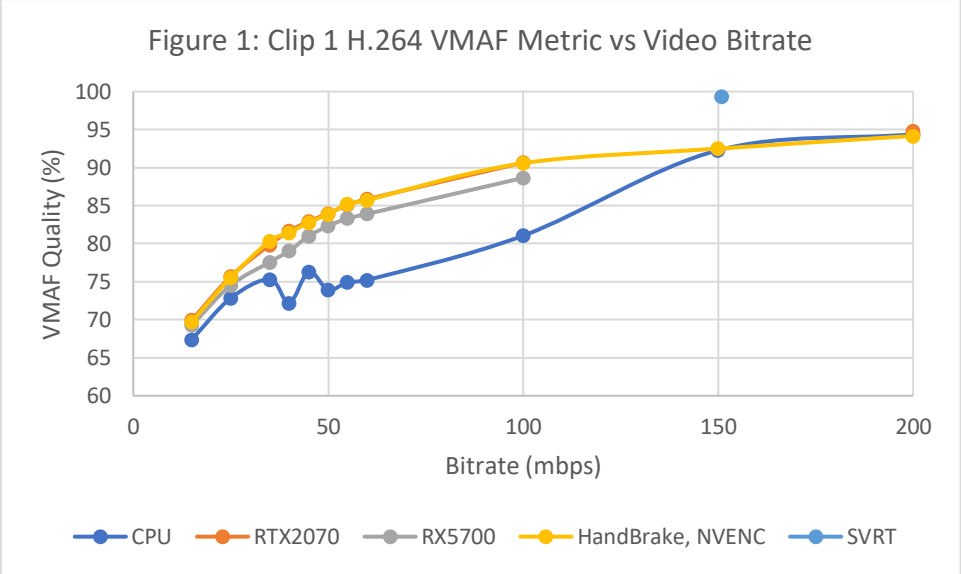
The PD20 encode evaluation utilized 2 sample clips from the internet and 1 personal clip. A few pertinent details of each clip are summarized below. The personal clip is of bubbling water in a swimming pool as encoding of such video often presents a quality challenge relative to the source quality. Similar to challenges of shimmering leaves or gravel trails/roads for riding enthusiasts often discuss loss of details in the forums. The timeline is nothing more than 1 of these clips at a time and produced to multiple video bitrate specifications.

Clip 1: H.264, 3840x2160 resolution, 150 Mbps video, 29.97 fps, Progressive scan, 14 sec duration
Clip 2: H.264, 3840x2160 resolution, 40 Mbps video, 30.0 fps, Progressive scan, 10 sec duration
Clip 3: H.264, 3840x2160 resolution, 25 Mbps video, 29.97 fps, Progressive scan, 34 sec duration

In addition to the evaluation of the 3 clips, 2 different GPU's were compared. An EVGA Nvidia RTX2070 (512.15 driver), and an XFX AMD Radeon RX5700 (Adrenalin 22.5.1) as well as the basic PD20 build 2815 software encoding via the CPU. Some improvements in AMD AMF/VCE but probably not linked in this version of PD20 https://www.tomshardware.com/news/amf-encoder-rivals-nvidia-av1-still-supreme
This evaluation discussion also compares the VMAF quality for the native source H.264 codec of these 3 sample clips as well as transcoding to H.265 at various video bitrates for the Clip 1 video stream.

Clip 1 Results

Figure 1 shows the VMAF quality of clip 1 for various video bitrate mbps encode settings. The PD20 default H.264, 3840x2160 MP4 profile was utilized as the basic profile definition. I did not want to taint this study with any user profile.ini edits for profiles. This profile matched frame resolution, fps, and scan type of the source video. The only thing changed in the profile is the video average bitrate Kbps value in PD20 dialog window. One would expect a fairly smooth and continuous quality improvement with increasing video bitrate. This is observed in the PD20 RTX2070 NVENC encoded stream and the characteristic and VMAF quality compares well (orange and yellow curves overlay each other) with HandBrake NVENC encoding, which is really ffmpeg under the hood. However, the CPU encoding often



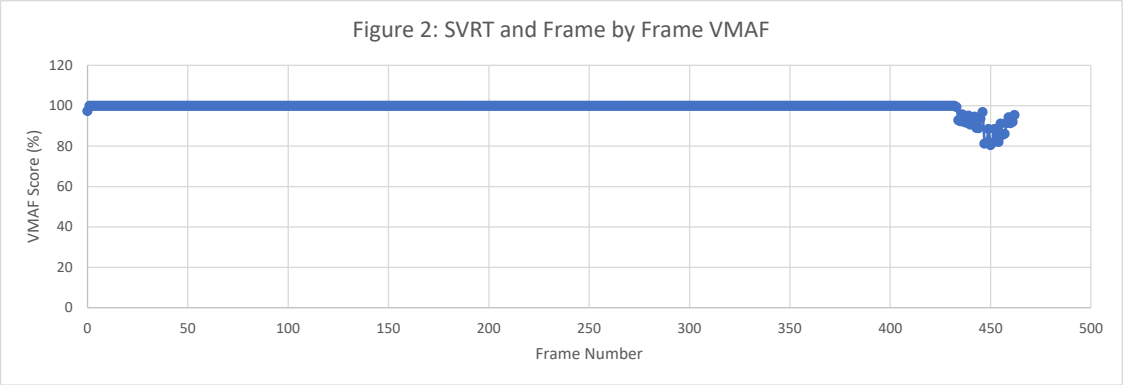
has unexplained dips like at 40 Mbps and the results from 50 Mbps to 150 Mbps really lacks integrity. Something really appears amiss with PD20 CPU encoding characteristics for this Clip 1 sample video.

These VMAF results for CPU encoding are of significant magnitude and represent a visible quality degradation to the source recording. Nvidia hardware encoding appears to display a very logical progression of quality with increased bitrate and appears to always have better VMAF metrics vs PD20 CPU's encoding.

The AMD RX5700 with VCE2.0 hardware encoding has about the same encode VMAF metrics as the Nvidia RTX2070 and also shows good logical progression with video bitrate. However, the RX5700 with PD20 does not produce clips with video bitrate higher than 100 Mbps, source clip is 150 Mbps, it reverts back and produces a clip with only 20 Mbps.

Another often discussed feature is PD20 ability to use SVRT, or Smart Video Rendering Technology. This CL technology only encodes the video in areas that it needs to per CL's documented SVRT rules, https://help.cyberlink.com/stat/help/powerdirector/20/win/enu/98_02_00_svrt_when.html?q=svrt . One can see the extremely high VMAF score for this video stream in figure 1, nearly 100 for a perfect match vs source. The reason figure 1 shows the SVRT clip average data point with a value slightly less than 100, part of the clip gets encoded even though this timeline has just the video clip and no other

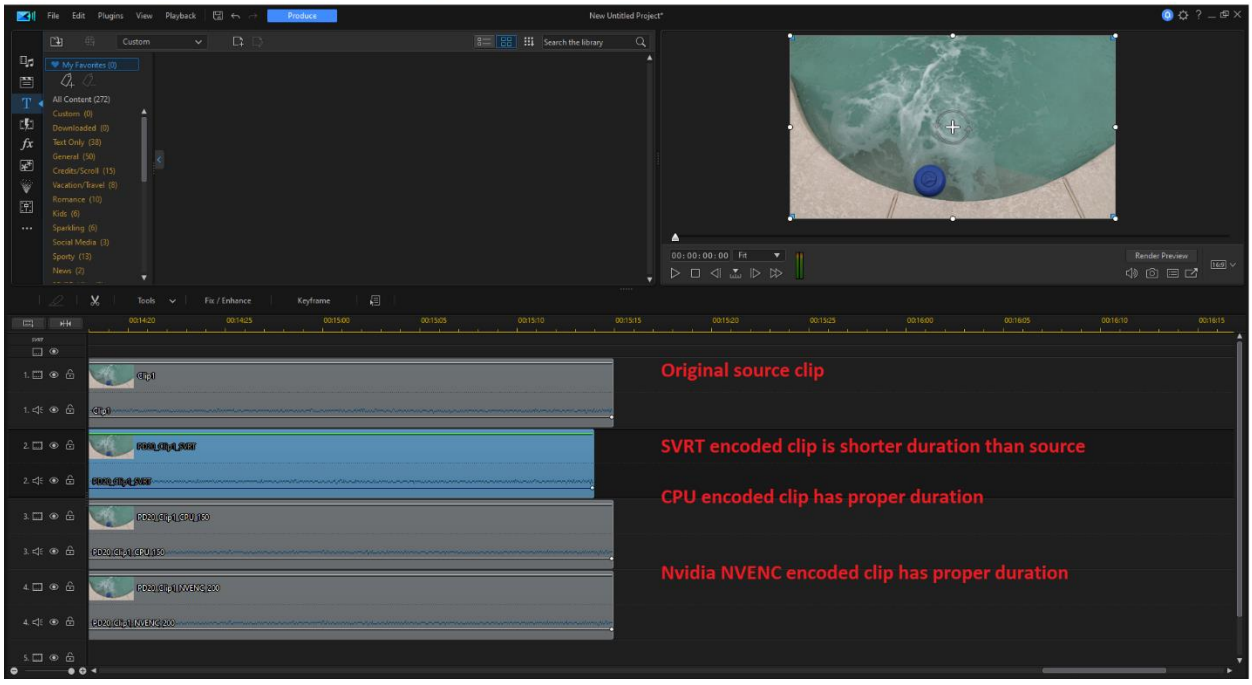
editing. Figure 2 shows the frame-by-frame VMAF score, one can clearly see the last chunk of frames are encoded vs passed through and this results in reduced VMAF quality associated with the CPU



encoding of this section of the clip. One can see the CPU encoded average of about 92% VMAF in this chunk of frames which agrees with figure 1 results at 150Mbps, which is the bitrate of the SVRT clip.

PD SVRT does not come without issue for this Canon clip, for that matter, all similar settings Canon source clips from this Canon Vixia HFG50. This issue with SVRT encoding is shown in figure 3. The

Figure 3: SVRT Encode Issues for Clip 1



resulting SVRT encoded stream is shorter in duration than the original source clip and also shorter than CPU encoded or Nvidia NVENC encoded streams of the source clip, so the issue surely lies with PD20 SVRT implementation and these Canon source clips. A simple review of the clip GOP details provides insight into the issue, but that's for CL developers, QA, and coders to provide features that work! To

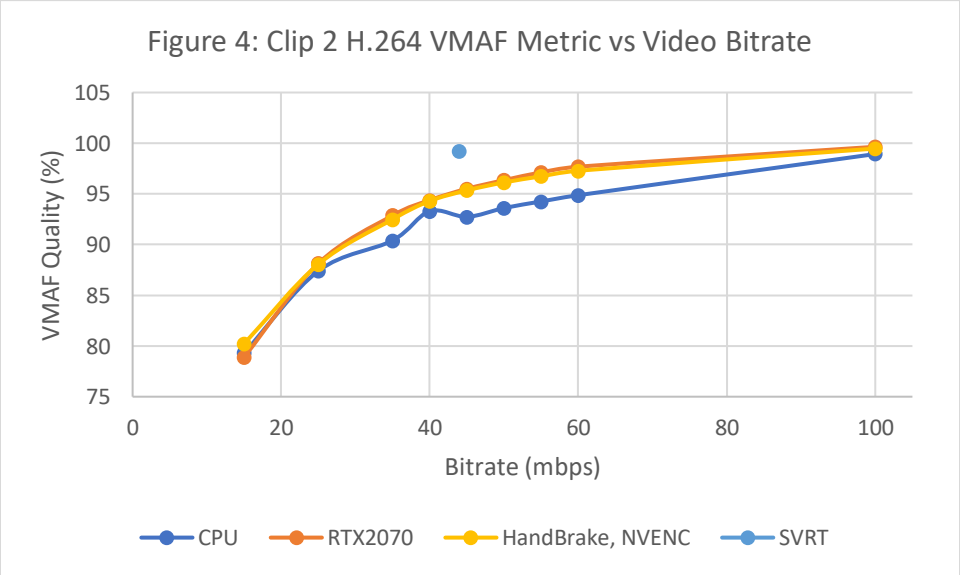
provide a proper frame by frame VMAF shown in figure 2, I've had to properly align the SVRT produced video frames with source to provide proper VMAF quality. If one does not take that care, the VMAF scores are meaningless as dissimilar frames are compared.

Basic Conclusions for Clip 1:

1) PD20 Nvidia NVENC hardware encoding and HandBrake NVENC encoding have very similar VMAF quality and expected characteristics with increasing video bitrate
2) CPU encoding struggles with this source clip to provide reasonable VMAF metrics except at source clip video bitrate and higher
3) Although PD20 generates a SVRT profile with its built-in tools, encoding with this profile has issues as final clip duration does not match the source or the CPU and/or Nvidia encodings

Clip 2 Results

Figure 4 shows the same VMAF quality of clip 2 for various video bitrate Mbps encode setting options. The default H.264, 3840x2160 profile was utilized again as the basic profile definition. This profile matched frame resolution, fps (adjusted to 30 vs 29.97), and scan type of the source video. The only thing changed in the profile is the video average bitrate Kbps value in PD20 dialog window. Again, one would expect a fairly smooth and continuous quality improvement with increasing video bitrate. This is observed in the PD20 RTX2070 NVENC encoded stream and the characteristic and VMAF quality again compares very well with HandBrake NVENC encoding. However, the CPU encoding again has unexplained bump and wiggles like at 25-45 Mbps range. PD20 CPU encoding has across the board a lower VMAF score vs Nvidia, but the magnitude is in the marginally perceptible area, with only ~3% VMAF shortfall relative to Nvidia. Still appears to provide rational that something appears amiss with



PD20 CPU encoding characteristics for this Clip 2 sample video.

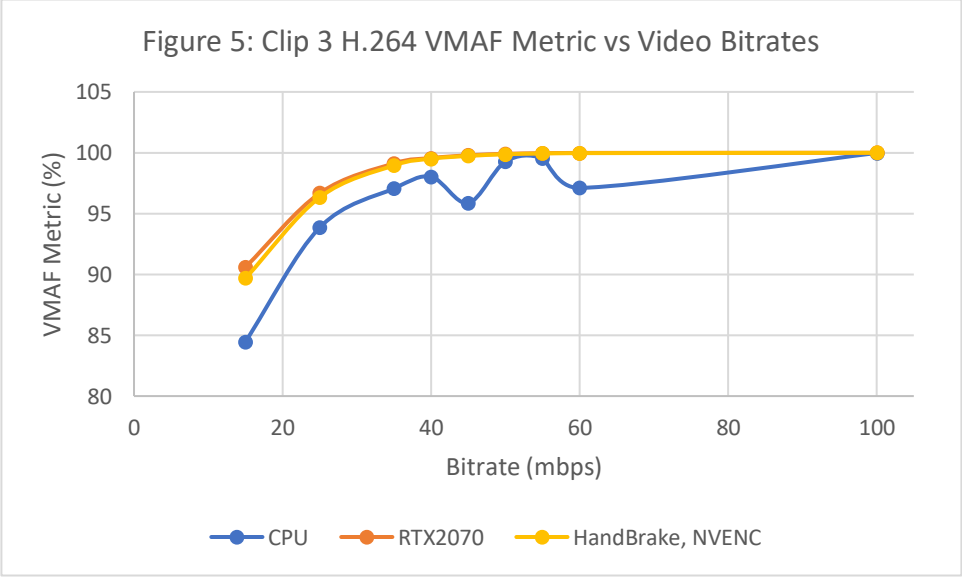
PD20 can create a successful SVRT profile for this clip and the produced duration matches the source clip. From figure 4, the SVRT produced clip had a VMAF of very near 100% at the SVRT profile bitrate of 44 Mbps. Again, the reason for the shortfall is the same as was shown in figure 2 and discussed there, the last chunk of frames are CPU encoded and have a CPU encoded based VMAF at that bitrate, or about 92%. This pulls the overall average VMAF score down from 100% as this clip is only 10 seconds long, 300 frames are averaged for the composite score.

Basic Conclusions for Clip 2:

1) PD20 Nvidia NVENC hardware encoding and HandBrake NVENC encoding have very similar VMAF quality and expected characteristics with increasing video bitrate
2) CPU encoding struggles again with this source clip to provide reasonably smooth VMAF metrics with increasing video bitrate
3) CPU encoding is again underperforming in quality vs Nvidia encoding
4) PD20 generates a SVRT profile with its built-in tools, encoding with this profile can provide excellent video quality vs source if no other issues are brought forth in a real timeline with effects, transitions, titles, and such

Clip 3 Results

Figure 5 shows the same VMAF quality of clip 3 for various video bitrate Mbps encode setting options. The default H.264, 3840x2160 profile was utilized again as the basic profile definition. This profile matched frame resolution, fps, and scan type of the source video. The only thing changed in the profile is the video average bitrate Kbps value in PD20 dialog window. Again, one would expect a fairly smooth and continuous quality improvement with increasing video bitrate. This is observed in the PD20



RTX2070 NVENC encoded stream and the characteristic and VMAF quality again compares well with HandBrake NVENC encoding. However, the CPU encoding again has unexplained dips and wiggles like

at 40-60 Mbps. CPU encoding has across the board a lower VMAF score vs Nvidia, the magnitude is in the perceptible area with nearly 5% VMAF shortfall relative to Nvidia. This clip 3 still appears to provide rational that something appears amiss with PD20 CPU encoding characteristics. Encoding the clip at bitrates higher than the source clip bitrate of 25Mbps can be seen as very beneficial. This is not abnormal as it really depends on the quality of the initial encoding of the clip. One forum contributor indicated "Assuming you're staying with the same encoding standard, like AVC to AVC, doubling the bitrate won't get you any higher quality because there is no more information in the original clip to be gained." in a particular post. In my view, I think that should be qualified that the same encoder/decoder was used or it's not really a valid statement. Many cameras have far superior onboard encoding during original video capture than many software products and often a poor man's method of overcoming the software encoding deficiency is to simply up the video bitrate during production of final edited timeline which can be advantageous to maintaining source quality. Of course, this comes at the expense of filesize.

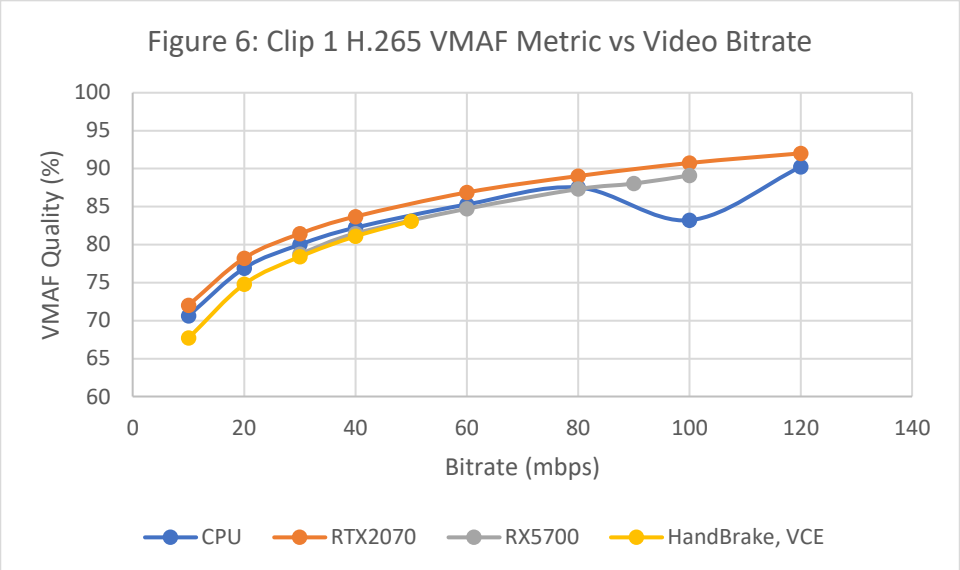
PD20 cannot create a successful SVRT profile for this clip.

Basic Conclusions for Clip 3:

1) PD20 Nvidia NVENC hardware encoding and HandBrake NVENC encoding have very similar VMAF quality and expected characteristics with increasing video bitrate
2) CPU encoding struggles again with this source clip to provide reasonably smooth VMAF metrics with increasing video bitrate and significantly falls short in quality if one needs to down sample the video bitrate
3) CPU encoding is again underperforming in quality vs Nvidia encoding
4) PD20 cannot generate a SVRT profile with its built-in tools

Clip 1 H.265 Results

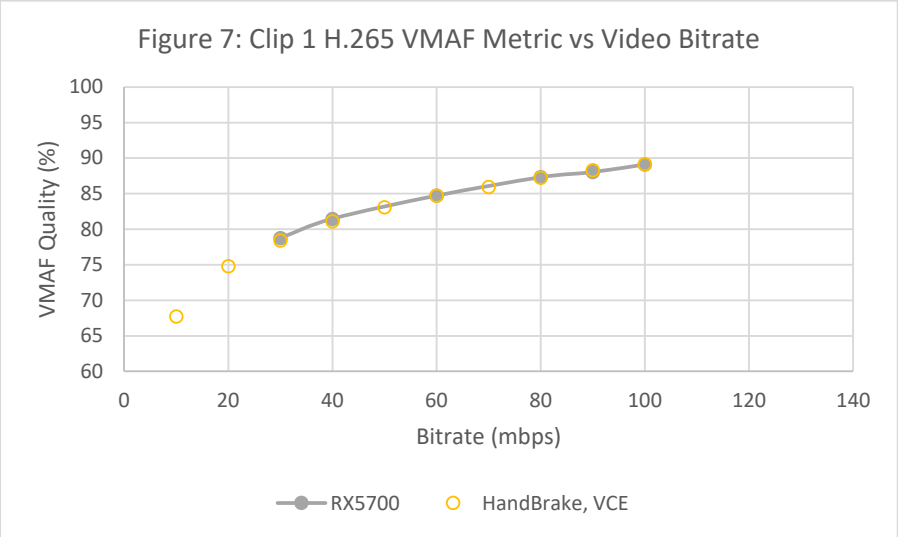
Figure 6 shows the VMAF quality of clip 1 for various video bitrate Mbps encode settings. The default H.264 video stream was transcoded to H.265 utilizing the default 3840x2160 H.265 MP4 profile as the basic profile definition. This profile matched frame resolution, fps, and scan type of the source video.



The only thing changed in the profile is the video average bitrate Kbps value. One would expect a fairly smooth and continuous quality improvement with increasing video bitrate. This is observed in the PD20 RTX2070 NVENC encoded stream. However, the CPU encoding is more consistent than H.264 observations but again has an unexplained dip at 100 Mbps. Nvidia hardware encoding appears to display a very logical progression of quality with increased bitrate and appears to always have better VMAF metrics vs PD20's CPU encoding.

The RX5700 had very unusual results, entering 10000 Kbps (10Mbps) in the PD video encode settings window for average bitrate setting, resulted in a produced video file with 30 Mbps bitrate, 20 Mbps produced 60 Mbps. It appears entered values 10000, 20000, 30000 Kbps are scaled by 3 perhaps, 40000, 60000, 80000 and 100000 Kbps were as entered. Additionally, no results above 100 Mbps could be obtained, anything entered above 100 Mbps produced a file with 30 Mbps video bitrate, yet defining 30 Mbps produced a file of 90 Mbps. The graph could be constructed by just entering some different numbers and then plotting the actual video encode bitrate with the analyzed VMAF. Not ideal for users to produce a given bitrate file, but it does provide a means to compare VMAF quality and it appears to be similar to Nvidia.

On the same WIN10 platform, HandBrake VCE encoding with the same RX5700 GPU results in proper produced encode bitrate parameters from 10 Mbps-100 Mbps in 10 Mbps increments and overlays on the PD20 RX5700 VCE VMAF results as shown in figure 7, so the issue would appear to be solely PD20's bitrate implementation. It could be a simple parsing error of the input parameters.



Obviously with the issues seen with the AMD RX5700 and PD20, one cannot in any good faith recommended that GPU or say it's compatible with PD20. It does "work" but that's not saying much, the Nvidia NVENC GPU encode option appears the best route, even better than CPU encoding.

If you made it this far, congratulations, I hope this writeup provided some value to your video encoding learning curve with PD20, if not, sorry for the wasted time for your reading.

Jeff (CyberLink Community Forum user JL_JL)