Overview and Analysis of Versatile Video Coding (VVC)

Bojan Dabevski, 181115
Ss.Cyril and Methodius University in Skopje
Faculty of Computer Science and Engineering
Skopje, North Macedonia
bojan.dabevski@students.finki.ukim.mk

Jana Markovikj, 181112
Ss.Cyril and Methodius University in Skopje
Faculty of Computer Science and Engineering
Skopje, North Macedonia
jana.markovikj@students.finki.ukim.mk

Sasho Gramatikov
Ss.Cyril and Methodius University in Skopje
Faculty of Computer Science and Engineering
Skopje, North Macedonia
sasho.gramatikov@finki.ukim.mk

***Abstract*—Versatile video coding (VVC), also known as H.266, is a new video compression coding standard finalized by the Joint Video Experts Team (JVET) on 6 July 2020. It is the successor to HEVC (High Efficiency Video Coding). It offers the highest compression efficiency available today. It can be used to reduce the bitrate of content while maintaining the same quality. In this paper, we will be conducting an in-depth analysis of the codec and its features and capabilities and seeing some side by side comparisons to its predecessor.**

Keywords—coding, compression, decoding, video codec, MPEG, filtering, quantization, HEVC, partitioning, blocks, VVC, intra-prediction

I. Introduction

Because video takes a significant amount of internet traffic (60% and increasing) [1], and with the occurrence of the new COVID-19 pandemic people are now using video conferencing all over the world and the number has skyrocketed. Video codec is the implementation of encoding and decoding digital video, which extremely reduces the bitrate of uncompressed video without any noticeable visual quality degradations. New codecs have been developed all over the years since the 1980s, each one more superior than its predecessor. Because of this, every new codec has made a major leap forward for video compression technology, including the newest standard, Versatile Video Coding. VVC offers the highest compression efficiency of all codecs, and it is particularly appropriate for higher resolution video streams due to its coding tools, which can operate on block sizes of up to 128x128 pixels. VVC's versatility also makes it an attractive choice beyond mainstream 2D video services. It had been designed to handle both traditional camera captured content as well as computer-generated imagery used in applications such as online gaming, e-sports, video streaming and screen-sharing applications.

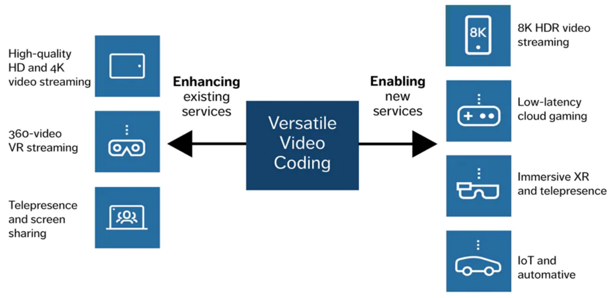

Figure 1. VVC application areas

II. Previous codecs

Because many practical video coding standards have been developed, we will be mentioning a few of the most popular ones used. H.120 was the first digital video compression standard, it provided important knowledge leading directly to its successors, such as H.261 which was the first practical video coding standard. The next one being H.262 whose main reason for usage was to support existing set-top-boxes. H.263, originally designed as a low-bit-rate compressed format for video conferencing. The most widely used have been the MPEG standards, used for compressing VHS-quality videos (MPEG-1) and DVD/SD digital television (MPEG-2). Currently the most extensively used standard is MPEG-4/H.264/AVC, because it is supported by practically all mobile devices and is heavily used by video streaming services such as YouTube, Netflix, Twitch etc. AVC has been succeeded by High Efficiency Video Coding (HEVC/H.265) developed in 2013. Mainly a standard for 4K broadcasting and recommended for HD (HDR) and 4K mobile streaming by multiple mobile telecommunication organizations. As the successor of HEVC, Versatile Video Coding (H.266) was developed and finalized on 6 July, 2020. It was based on many of the coding structures in HEVC and added further improvements mainly for 360° video, VR streaming and live streaming.

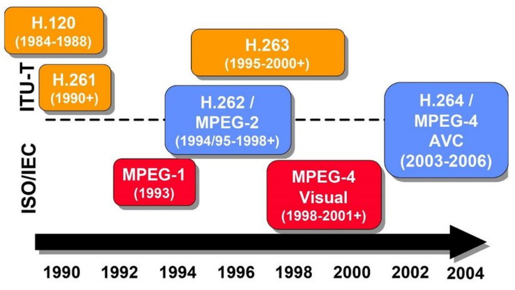

Figure 2. The history of video codecs

III. VVC Compression Tools

Versatile Video Coding has been designed to attain noteworthy improved compression compared to older standards such as HEVC and AVC, and at the same time to be very flexible for effective use in a wide range of applications. The superior coding efficiency of VVC is the result of combining many compression tools, each contributing with a small compression improvement. The analyzed tools in this paper are:

- Block partitioning
- Advanced inter prediction
- Dependent quantization
- Adaptive loop filtering
- Improved deblocking filtering

IV. Block partitioning

VVC uses a block-based hybrid coding architecture. This means that the codec itself does not code pixel values directly but actually predicts only error information to recompense for the inaccurate prediction. This activity actually runs on blocks of pixels instead of the entire picture itself, because it makes it possible to modify and alter to local video picture characteristics and minimizes the prediction error/failure.

Because of this, a well-structured picture partitioning scheme is needed to achieve high compression efficiency. VVC specifically uses three types: multi-type tree, separate chroma partitioning and virtual pipeline data units.

A. Multi-type tree

One chunk is partitioned into multiple coding tree units (CTU). A coding tree unit is the basic processing unit of the HEVC video standard and conceptually corresponds in structure to macroblocks (a processing unit based on linear block transforms) that were used in several previous video standards. The CTU can use larger block structures and can better sub-partition the picture into variable sized structures.

The minimum and the maximum sizes of the CTU are specified among the sizes of 8x8, 16x16, 32x32, 64x64 and 128x128. Then the CTU is partitioned into a number of coding units (CU) in order to adapt to various characteristics. For example: if CTU size was NxN where N is 128,64,32, or 16. The CTU can be a single CU or it can be split into four smaller units of equal sizes of N/2xN/2 which are nodes.

The CTU is partitioned using recursive quadree (QT) split (quadtree being a data structure in which each internal node has exactly four children/chunks and are used for partitioning a two-dimensional space into 4 partitions). VVC also supports binary-tree (divides a CU into two equal-sized CUs) and ternary-tree (divides a CU into three CUs of sizes 1/4th, 2/4th, 1/4th) split modes. After the CTU is split it can be done again and so on, that is why it is a recursive function. At the leaf of the tree (after splitting) there are nested recursive splits called Multi-type tree splits.

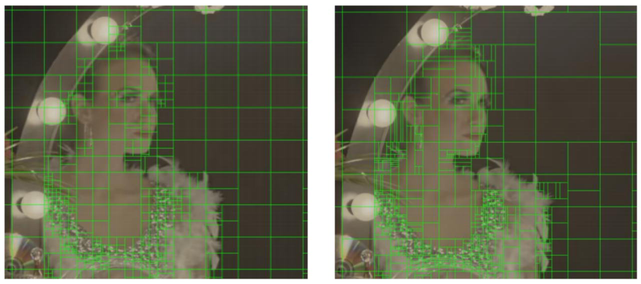

Figure 3. HEVC and VVC partitioning

When this is done, each split is called a CU. As seen on the picture, larger blocks are used in VVC which reduces the structure coding cost and partitions match more closely to the objects' boundaries.[4]

B. Separate Chroma Partitioning

Separate Chroma Partitioning or so-called Dual Tree Partitioning enables/creates two separate trees for the chroma(color) and luma (light) blocks. Specifically MTT coding tree. This is motivated by the fact that the color information has lower resolution and you would split the tree less for the chroma. If these are separated you could stop the partitioning earlier for chroma which makes the encoding much faster. Because there are separate partitionings, the blocks are not aligned and mechanisms are needed to be implemented.

C. Virtual Pipeline Data Units

The virtual pipeline data units (VPDUs) are non-overlapping cells in the image. In the hardware decoding process (which shifts the video processing to the GPU), consecutive VPDUs are processed in parallel by multiple pipelines. In most pipelines the VPDU size is proportional to the buffer. With this you can define a maximum granularity for pipelining (good for high block sizes). Critical for intra blocks that require neighbouring samples to be reconstructed. VPDUs may be bigger or smaller than CUs. This is why a VPDU may contain a CU or a CU may contain a VPDU. The size of the VPDUs may vary for luma and chroma data. The picture represents a 64x64 which is typical for VPDU [3]. You can also see different splits of the CTU on the same picture (triple and binary). The VPDU boundary: dashed, the CU boundary: red. The picture below is used to better illustrate this.

In order not to have one CU which spans over two of the 64x64 VPDUs, restrictions are implemented in the splitting process in VVC that prevent that from occurring. This is done so that the throughput for pipeline processing is improved.

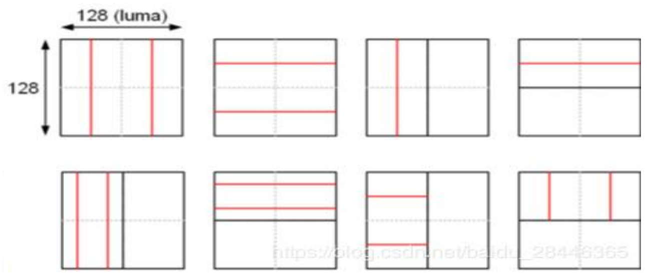

Figure 4. VPDU spanning CUs

V. Advanced inter prediction

One of the most efficient bit-saving techniques in video compression is inter-prediction, which simply means copying sample values from previously coded pictures. The information for where to copy from the horizontal and vertical displacement for a block, is stashed in a motion vector (key element in the motion estimation process). Techniques that improve inter prediction are responsible for a substantial part of the bitrate reduction difference between VVC and HEVC.

While there have been many techniques that have been used to achieve the advanced inter prediction in VVC, there were five that are particularly noteworthy:

- The affine motion model
- Adaptive motion vector resolution
- Bi-directional optical flow
- Decoder side-motion vector refinement
- Geometric partitioning mode

The techniques that are implemented in VVC which improve the inter prediction are intended to be a substantial part of the bitrate improvement.

The precision of motion vectors has also been increased to 1/16 pixel compared to the quarter pixel resolution of HEVC.

In previous video coding standards it was possible to compensate for translational motion (the decoder can be instructed to fetch sample values not from the same place in a previous picture but from another position). In VVC the affine motion model makes it possible to specify the distance, rotation and zoom which can save a lot of bits for sequences containing such motion.

Another bit- saving technique implemented in VVC is the ability to vary the precision of motion vectors. For example the encoder can send a signal to the decoder that the incoming motion vectors are an integer or four-times-integer resolution. This process can save bits when it is doing the process of predicting smooth areas where the exact sub-pixel precision does not help much with image improvement over integer precision and also when representing large motion vectors.

VVC's advanced inter prediction in VVC further uses and makes most use of the fact that in some cases the information about motion does not have to be transmitted as motion vectors but can instead be collected from similarly moving parts of the picture. For example, the bi-directional optical flow tool can deduce motion vectors by measuring the optical flow in the reference pictures and the decoder side-motion vector refinement can make motion vectors by minimizing differences between reference pictures.

Finally VVC's advanced inter prediction also uses motions of non-rectangular shapes to better line up with the shape of the objects that are in motion. This means that one half of the block can have one motion vector and the other half can have another motion vector with the halves separated by one geometrical line that is decided by an offset and an angle.

VI. Dependent quantization

The quantizer is a central part of a video codec that is directly linked to operational control of video bitrate and visual quality. Quantization itself is the procedure of mapping out input values from a bigger set to output values in a smaller set, often with a limited number of elements.With the adjustment to the quantization step, the encoder controls the transform coefficients which is then coded with a lossless data compression (entropy) coder and is sent into the video stream.

Previous coding standards have used scalar quantizers but because the entropy coder processes each transform coefficient in a coding block in sequential order there was an inefficiency if the quantization of the coefficient were to be determined independently. Some encoders solved this issue by using an algorithm called trellis quantization (it improves data compression in DCT-based encoding methods).

Dependent quantization was introduced in VVC. Dependent quantization means that the codec can switch between two shifted quantizers and thereby reduce the quantization error. Based on the way VVC is designed, the quantization levels for a particular transform coefficient depend on the values of the quantized coefficients prior to the current one. In order to use this tool effectively the encoder needs to evaluate how the determined quantization level for each coefficient impacts both the bit count and the total reconstruction error for the whole block. The new dependent quantization implemented in VVC removes the inefficiency of previous quantizing coefficients and it provides a substantial bit-reduction over HEVC.

VII. Adaptive loop filtering

The adaptive loop filter (ALF) is a new loop filter in VVC. This filter scans the picture after it has been decoded and selectively applies (on a CTU basis) one of several two-dimensional finite impulse responses filters before the picture becomes output or is used for prediction. The encoder calculates all the sets of filter coefficients that will lead to even the smallest of errors and then it transmits those to the decoder. These impulses are named luma component and chroma component. The luma component has 25 different filters which is a significant number and the determination for which of these 25 filters to choose is actually dependent on the so-called activity index and the so-called directionality index, this means that you basically adapt on the features that you observe on the 4x4 block. The chroma component has a single set of filter coefficients. The shape of the filter is a diamond shape. The luma component can be a diamond shaped in 5x5 or 7x7 blocks, whereas the chroma component is more simple and can only be a 5x5 block shape. The reason for this is because the chroma signals by tendency have way less detail than the luma signals so this is the reason they are only in 5x5 blocks.

The Adaptive loop filtering has the ability to clean up artifacts (visible edges at some block borders) in the picture and also contributes to a substantial bit-reduction over HEVC.

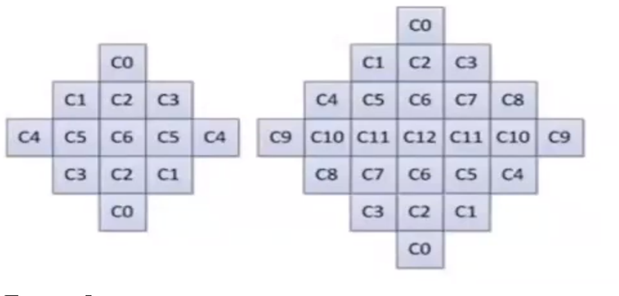

Figure 5. ALF diamond shapes in 5x5 and 7x7 blocks [8]

VIII. Improved deblocking filtering

VVC, HEVC and H.264 are all block based video codecs. The downside of this approach is that it can give rise to "block artifacts", these are visible edges at some block borders. As a way to fix this problem the deblocking filtering was created.

Deblocking filtering is a tool that was created to reduce these artifacts by selectively smoothing across the block boundaries. The deblocking filtering in VVC is based on HEVC deblocking filtering. On top of that VVC is also capable of using longer deblocking filters. The long deblocking filters allow the codec to be able to do stronger deblocking that can be more effective in hiding block artifacts, especially for larger blocks in relatively smooth areas. The long deblocking filters help improve subjective quality of VVC compared to HEVC.

IX. Performance comparison

In our work, we made a comparison between HEVC and VVC by using softwares to encode video files with different standards. The software used for HEVC encoding was ffmpeg [7], an open-source software designed for processing of video files on the command-line. For the VVC standard, we used the Fraunhofer Versatile Video Encoder (VVenC) software [6]. It is meant to be an efficient encoder implementation and, same as ffmpeg, it is used through the command-line. Along with this, we used the Cmake [8] as an additional software to control the compilation process.

Since the available VVC encoding software only accepts raw yuv (raster graphics) files, we needed to convert the original files into yuv format, encode it, and then convert it back to mp4/mov file. The main focus was the video quality and the impact it would have on the computer's performance whilst the encoding process was taking place. It is important to note that VVC's encoding process takes a lot longer to execute. In addition, this specific encoder is still in the early stages of development, despite the standard being issued. The parameters measured were the CPU usage, RAM usage, size and bitrate of the videos. The videos chosen to be encoded were:

1. A 5 minute mp4 video of 24MB (megabytes), with a resolution of 640x480.
2. A 7 second mov file of 8MB, with a resolution of 1920x1080
3. An 18 second mp4 video of 1,44 MB, with a resolution of 1280x720

The FramePerSecond and GroupOfPictures sizes were also measured. Fps of all the videos stayed the same at every video with every encoder. Whereas the GOP size changed with HEVC & VVC, but were the same for both encoders. The HEVC software was fast to encode, using an average CPU usage of 76% and using on average 661MB of RAM. During the encoding of the second and third video, however, it took a larger toll on the CPU and RAM usage.

The VVC software took longer to encode. It took on average 75% of the CPU usage and on average of 630MB RAM. All videos did have almost the same total bitrate as the original. The encoding of the second and third video especially went better with objectively finer results for the parameters.

With these tests the results showed that despite the fact that the VVC encoding software isn't fully developed, 2 out of the 3 encoding tests have proved to surpass the performance of HEVC. It should be noted that this is only based on our personal tests, and by using newer software and additional testing in the future more accurate results can be gathered. This would show the true improvement of VVC and its versatility.

	CPU	RAM	Size	Bitrate	FPS	GOP
Video 1	/	/	24 MB	639 kbps	29	153
HEVC 1	57%	460 MB	18.7 MB	490 kbps	29	206
VVC 1	83%	670 MB	19.2 MB	630 kbps	29	206
Video 2	/	/	8 MB	9588 kbps	25	166
HEVC 2	87%	767,4 MB	1,049 MB	1045 kbps	25	171
VVC 2	75%	447,7 MB	899 MB	9396 kbps	25	171
Video 3	/	/	2,44 MB	1136 kbps	29	64
HEVC 3	82,%	756,4 MB	1,32 MB	608 kbps	29	63
VVC 3	67,%	768,7 MB	1,1 MB	1113 kbps	29	63

Table 1. HEVC and VVC performance

X. Conclusion

Versatile Video Coding represents state-of-the-art video coding and is certain to play an important role in supporting a wide range of 5G uses to the IoT (Internet of Things) and many other industries. In addition, it supports new features of adaptive spatial resolution from picture to picture and scalability. These characteristics combined with its high compression efficiency make this new standard a flexible solution that is able to call attention to a variety of application and content types. VVC was designed to manage the high demand that increasing amounts of video poses on networks, research reveals that VVC achieves the best available compression performance at a computational complexity suitable for implementation in both software and hardware and is certainly the future standard in video compression software.

References

[1] https://www.researchgate.net/figure/Chronology-of-the-International-Video-Coding-Standards_fig1_326632519

[2] https://2020.ieeeicip.org/program/tutorials/versatile-video-coding-algorithms-and-specification/

[3] https://www.programmersought.com/article/9684838964/#:~:text=Defined%20as%20non%2Doverlapping%20cells,size%20cannot%20be%20too%20large.

[4] E. Francois, M. Kerdranvant, R. Jullian, C. Chevance, P. De Lagrange, F. Urban, T. Poirier, Y. Chen, VVC Per-tool Performance Evaluation Compared to HEVC, InterDigital, France, August 2020

[5] Gisle Bjontegaard, "Calculation of Average PSNR Differences between RD curves", ITU T SG16/Q6 VCEG 13th meeting, Austin, Texas, USA, April 2001, Doc. VCEG-M33 (available at http://wftp3.itu.int/av-arch/video-site/0104_Aus/).

[6] https://github.com/fraunhoferhhi/vvenc

[7] https://www.ffmpeg.org/

[8] https://cmake.org/