Accepted in IEEE International Conference on Visual Communications and Image Processing (VCIP 2021), 2021
Authors accepted manuscript.The final published version available at https://doi.org/10.1109/VCIP53242.2021.9675400

CRC-Based Multi-Error Correction of H.265 Encoded Videos in Wireless Communications

Vivien Boussard*,† and Stéphane Coulombe† François-Xavier Coudoux*, Patrick Corlay* and Anthony Trioux*

†*Dept. of Software and IT Engineering*
École de technologie supérieure
Montreal, Canada
vivien.boussard.1@etsmtl.net,
stephane.coulombe@etsmtl.ca

**UPHF, CNRS, Univ. Lille, ISEN,*
Centrale Lille, UMR 8520 - IEMN, DOAE
Valenciennes, France
{francois-xavier.coudoux, patrick.corlay,
anthony.trioux, vivien.boussard}@uphf.fr

***Abstract*—This paper analyzes the benefits of extending CRC-based error correction (CRC-EC) to handle more errors in the context of error-prone wireless networks. In the literature, CRC-EC has been used to correct up to 3 binary errors per packet. We first present a theoretical analysis of the CRC-EC candidate list while increasing the number of errors considered. We then analyze the candidate list reduction resulting from subsequent checksum validation and video decoding steps. Simulations conducted on two wireless networks show that the network considered has a huge impact on CRC-EC performance. Over a Bluetooth low energy (BLE) channel with Eb/No=8 dB, an average PSNR improvement of 4.4 dB on videos is achieved when CRC-EC corrects up to 5, rather than 3 errors per packet.**

***Index Terms*—Cyclic Redundancy Check, Error Correction, H.264, H.265, Wireless Communications.**

I. Introduction

Reliable real-time video transmission over wireless networks, and especially mobile networks, is challenging. The mobility of both the transmitter and the receiver induces variable channel conditions. Moreover, since the retransmission of erroneous packets is impossible due to low delay constraints, error control must thus be performed at the receiver side to recover information. Several methods are available for handling missing or corrupted information in video communications, including video error concealment and error correction. Video error concealment methods [1] offer high reconstruction capabilities. However, such approaches suffer from several drawbacks. First, their performance depends on the video characteristics (i.e., motion and textures) of a sequence. Moreover, they consider corrupted packets as lost, and do not try to extract any information from them.

On the other hand, error correction methods use the received corrupted packet to reconstruct the originally transmitted one. They reconstruct the most probable video sequence either from soft bits information or from the coded bitstream's syntax [2]. Other techniques exploit the information available in the protocol stack to perform error correction [3]. In traditional wireless environments such as Wi-Fi [4], there are two error detection codes in the protocol stack, namely, cyclic redundancy check (CRC) [5] at the link layer and checksum [6] at the transport layer. CRC-based error correction approaches can be classified into three main categories: optimization-based, table-based, and table-free. Optimization-based CRC error correction, as described in [7], generates the most probable CRC-compliant sequence based on optimization techniques such as ADMM [8] and BP [9]. This approach has been applied to Bluetooth Low Energy (BLE) channels for IoT environments [10], and offers very good error correction capabilities, for up to 3 errors. However, as the methods are iterative, the accuracy of the results depends on the number of iterations performed. Therefore, achieving high-accuracy reconstructions becomes extremely complex.

Table-based approaches [11], [12] are based on the CRC syndrome computed at the receiver. In such methods, a list of error patterns and their associated syndromes up to a determined length are stored in a table prior to communications. Upon reception of a corrupted packet, the table is scanned for the computed syndrome. If a match is found, the corresponding bit position is flipped. One weakness with these approaches is that they automatically correct a packet without first validating the reconstructed bitstream. Moreover, to be adapted to larger packets and more errors, the table sizes must be increased to intractable lengths, as shown in [13]. In [13], we proposed a table-free method to handle the correction of multiple errors in corrupted packets, which can be adapted to any generator polynomial. It performs arithmetic operations on the computed syndrome in order to generate the list of all error patterns (i.e., candidates), comprising up to N errors, leading to such syndrome and where $N \geq 1$ is arbitrary. In this paper, we apply this method to video transmission over wireless networks. The novel contributions of this work are:

- Estimation of the number of candidate error patterns in CRC error correction, depending on the packet length, the generator polynomial and the maximum number of errors considered, N.
- Analysis of the impact of checksum validation and video decoding on the candidate list size reduction.
- Demonstration of error correction gains in wireless communications, using 802.11p [14] and BLE [15], when $N = 5$.

This work was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grant and UPHF.

This paper is organized as follows. In section II, we present related works on table-free CRC-based error correction. We propose theoretical improvements to this approach in section III, to estimate the number of candidate error patterns. Simulations and results in 802.11p and BLE wireless environments are illustrated in section IV. In section V, we conclude the paper and discuss future work perspectives.

II. CRC-based error correction

Prior works on CRC-based error correction are based on the definition of the CRC field. At the transmitter, for a desired generator polynomial (GP) of highest degree n, the CRC field is computed as the remainder of the long division of protected data, left-shifted by n positions, by the GP [13]:

$$\frac{d_T(x).x^n}{g(x)} = q(x) + \frac{r_T(x)}{g(x)} \tag{1}$$

where $d_T(x)$ is transmitted protected data, $g(x)$ is the generator polynomial, $q(x)$ is the quotient of the division and $r_T(x)$ the remainder of the division prior to the transmission, i.e., the CRC field. This field is appended to the data and transmitted to the receiver through the channel. At the receiver side, a long division of the received data appended by the CRC field is performed to obtain a new remainder:

$$\frac{d_R(x).x^n + r_R(x)}{g(x)} = q(x) + \frac{s(x)}{g(x)} \tag{2}$$

where $d_R(x)$ and $r_R(x)$ are the received data and remainder, respectively, and $s(x)$ is the remainder of the division at the receiver, also known as the syndrome. If no error occurred during the transmission, the syndrome will be zero. Otherwise, a non-null syndrome indicates that the received packet has been corrupted. The standard practice is to discard such a packet. However, based on the CRC's construction, the syndrome can be used for error correction purposes. In the approach, the syndrome is reversely processed by adding shifted versions of the generator polynomial to generate the list of error patterns that led to the syndrome. As the full list of such error patterns contains a large number of candidates (2^m, where m is the data bit length), we proposed in [13] to drastically reduce the number of candidates by considering only the patterns containing very few erroneous bits. Thus, we aim to correct slightly corrupted packets, from which we can extract valuable information to improve the visual quality of the reconstructed video sequence.

III. Analysis of the Candidate List

In this section, we analyze the different parameters that affect the number of entries in the list of possible error patterns composing the candidate list. We showed in [13] that increasing the maximum number of errors considered, denoted N, results in an increase in the number of candidates. We propose to quantify and predict such an increase as a function of the GP and of the packet length. We remind the reader that the error correction process described in [16] consists of three steps. First, the CRC error correction (CRC-EC) gives the exhaustive list of error patterns leading to the computed syndrome containing up to N errors. Then, the checksum is tested on each candidate to discard the non-checksum-compliant ones. Finally, the video decoder is used to validate the syntax of the remaining candidates. If there is a single candidate remaining at the end of the process, the correction is performed. If not, video error concealment is applied.

A. Estimating the number of candidates

It is known from [13] that as the packet length and the number of errors considered increase, so too will the number of candidates in the list generated by CRC-EC, on average. If there are many candidates remaining at the end of the process, we perform error concealment instead, as CRC-EC is unable to discriminate among these multiple candidates. Estimating the number of candidates in the list (NCL) can help determine if such error correction is realistic in a specific scenario, which in turn reduces the required computations. Clearly, generating a list containing too many candidates becomes pointless since it cannot lead to packet correction. It should be recalled that the cycle length of a generator polynomial, denoted C, corresponds to the number of unique single error patterns it can produce. Typically, for a generator polynomial of highest degree n with even parity, the cycle length is equal to $2^{(n-1)} - 1$. Based on this definition, a random syndrome $\mathbf{s}$ associated with a single error will produce a list of exactly one candidate if the packet's length L is equal to C. The probability that such a random syndrome $\mathbf{s}$ produces a candidate decreases with the packet length L when $L < C$. Hence, given a uniform distribution for the single error position, the NCL for 1 error can be expressed as:

$$\mathrm{NCL}_1(L, C) = L/C \tag{3}$$

It can be seen in (3) that as the length L tends towards the cycle length C, the number of candidates in the list tends towards 1. If $L > C$, the list contains one or more candidates. However, since $L < C$ for most communications, then (3) can be viewed as the probability that the list contains one candidate. To extend the estimation to N errors, we must reiterate the N-error CRC-based correction process that consists in forcing $(N-1)$ error positions and then searching for the last error position within the remaining length. Based on Fig. 1, we can observe that for any position i of the last forced bit F_{N-1}, there are $(i-1)$ positions containing $(N-2)$ forced positions $\{F_{N-2}, \ldots, F_1\}$ on the LSB side of F_{N-1}. This corresponds to all the combinations of $(N-2)$ forced positions within $(i-1)$ positions. On the MSB side of F_{N-1}, there are $(L-i)$ remaining positions on which the search for the last error P_1 is performed, which is equivalent to (3) but with length $L-i$. The estimation of the candidate list size is thus performed by computing, for each last forced position F_{N-1} from $(N-1)$ to $(L-1)$, the number of combinations for the $(N-2)$ other forced bits multiplied by the expected number of candidates

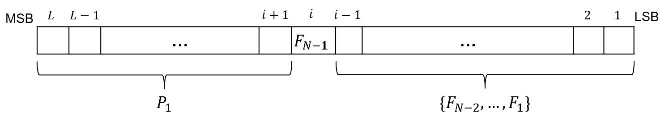

Fig. 1: Illustration of the last forced position F_{N-1} at step i

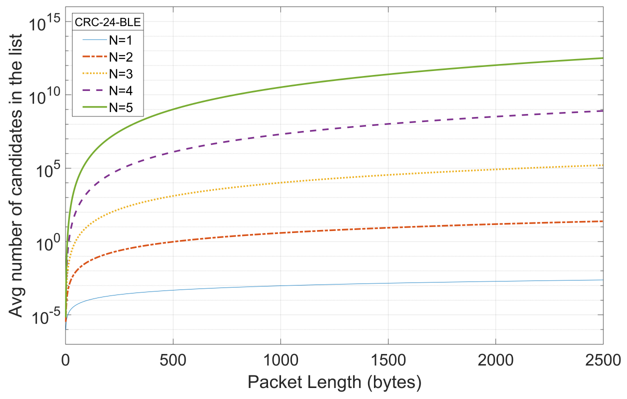

Fig. 2: Evolution of the NCL for different packet lengths, applied to CRC-24-BLE

in the remaining length $(L-i)$. When considering N errors, the list size can thus be expressed as:

$$\mathrm{NCL}_N(L,C,N) = \sum_{i=N-1}^{L-1} \binom{i-1}{N-2} \times \frac{L-i}{C} \tag{4}$$

In Table I, we compare the theoretical results obtained with (4) and the simulation results we observed by running the error correction process of [13] for different packet lengths and a number of errors $N=3$. Except for the lowest packet length considered, the difference between the experimental and theoretical values is less than 2%. We estimated the NCL for a CRC-24, as illustrated in Fig. 2, for $N=1$ to 5, and for packet lengths ranging from a few bits to 2500 bytes. We also applied it to CRC-32, which is used at the link layer of 802.11 communications, as illustrated in Fig. 3. We observe that the number of candidates greatly increases with the packet length, but the NCL for a given number of errors and packet length is significantly lower when using the CRC-32 versus the CRC-24. For example, when searching for 3 errors, considering packets length of 1500 bytes, using the CRC-24 produces a list of 34,347 candidates, while using the CRC-32 decreases this number down to 67. Having a list comprising fewer candidates increases the probability of identifying the error pattern that actually occurred, considering that N errors or less hit the packet, especially when combining the approach with the validation mechanisms presented in [16], namely, the checksum and the video decoder validations. Significantly decreasing NCL allows considering higher values of N. However, as illustrated by the green curve in Fig. 3, associated with $N=5$, NCL rapidly increases up to several millions of candidates when the packet length is over 500 bytes. Although increasing N to 5 helps identify more error patterns, it also leads to intractable numbers of candidates for high enough

TABLE I: List size estimation and actual list size comparison for different packet lengths, applied to CRC-24, with $N=3$.

Length (bytes)	List size (sim.)	List size (theor.)
250	159	160.6
500	1,312	1,287
1500	34,347	34,738
2500	158,465	160,815

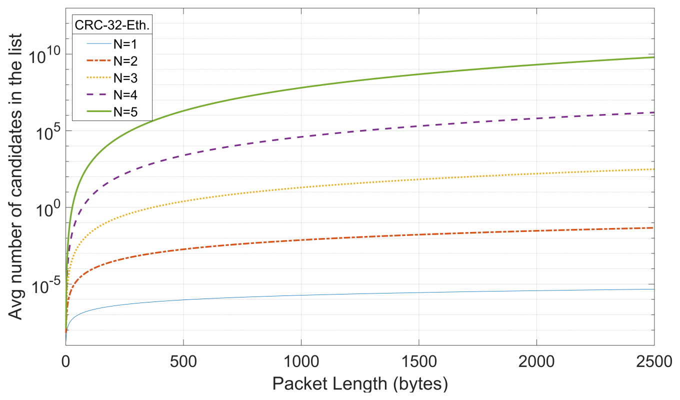

Fig. 3: Evolution of the NCL for different packet lengths, applied to CRC-32-Ethernet

packet payloads. Nevertheless, it is possible to increase N, and thus the error correction capability, while maintaining a reasonable number of candidates when using packets with low-enough payloads, i.e., up to several hundred bytes.

B. *Impact of the video codec*

In this section, we study the impact of the codec used on the final candidate list size. Table II shows the number of candidates at different stages of the correction process for H.264 Baseline [17] and H.265 Main [18]. In the example, the packet length tested has a size of 1500 bytes and the number of errors considered is $N = 3$. As can be seen, the number of candidates within the list is nearly the same, since it is independent from the codec and syntax of the transmitted video, but only dependent on the packet length. We observe a significant decrease in the number of candidates after the checksum validation (CV), but still nearly similar values for both codecs. The main difference lies in the average number of packets the video decoder can reconstruct without crashing or encountering invalid information. For the H.264 Baseline, which uses context-adaptive variable-length coding (CAVLC), an average of 3.40 candidates per corrupted packet can be decoded. In fact, with CAVLC, the bitstream contains a significant part of non-desynchronizing bits (NDBs) [19], which are bits that can be flipped without causing the decoder to crash. These bits represent about 30% of an H.264 Baseline encoded packet. However, since most candidates in Table II contain 3 errors, only about $(0.3)^3 = 2.7\%$ of the candidates remaining after CV are decodable. For H.265 Main, which uses context-adaptive binary arithmetic coding (CABAC), the number of NDBs is much lower, which results in a less error-permissive bitstream. Hence, only the repaired packet can be decoded successfully.

TABLE II: List size comparison after CRC error correction (CRC), after additional checksum validation (CRC+CV) and after additional video decoding check (CRC+CV+VDC), for packets' lengths of 1500 bytes and CRC-24 ($N = 3$).

Codec	CRC	CRC+CV	CRC+CV+VDC
H.264 Baseline	34,705	117.1	3.4
H.265 Main	35,383	118.9	1

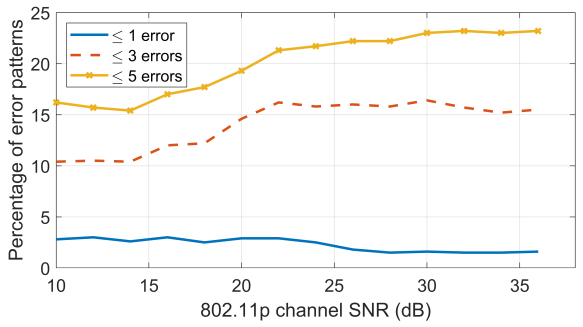

Fig. 4: Percentage of error patterns containing less than a determined number of errors, for simulated 802.11p channels. Tests are conducted on packets with payload length of 255 bytes

IV. Simulations Over Wireless Communications

As discussed in section III, for practical CRC-EC implementations, the number of errors searched must be set in accordance with the average packet length. In this paper, we propose to analyze the performance of 5-error correction. We encoded the sequences using HM-16.20 [20], setting a packet length limit of $L_{\mathrm{MAX}} = 255$ bytes. To simulate the mobility of the wireless network, we used the 802.11p UrbanLOS channel model available in the Matlab WLAN toolbox, which is based on field measurements [14]. In Fig. 4, we show the percentage of error patterns containing at most 1, 3 and 5 errors, respectively, for such a channel, for L_{MAX} and channel signal-to-noise ratios (CSNRs) from 10 dB to 36 dB. The increase in error patterns when going from 3 to 5 errors is significant. For example, it goes from 15.7% to 23.2% at CSNR=32 dB. This means that 7.5% more packets can be corrected. However, at such a CSNR, the packet error rate is still quite high for vehicular scenarios, leading to the corruption of an average of 1% of the packets. In such conditions, correcting 20% of the corrupted packets (i.e., not handling 80% of them) leads to slight visual and PSNR improvements, and the CRC-EC should be combined with application-layer forward error correction (AL-FEC) [21]–[24] or additional means to enhance its performance.

However, CRC-EC can, without any additional support, achieve much higher error correction rates on wireless channels where most packets contain very few errors. As demonstrated in [13], BLE transmissions correspond to such scenarios. Fig. 5 presents the error distribution of the BLE channel simulator available in the Matlab Communications toolbox [15], for different channel qualities. We can see the significantly higher percentages of error patterns containing very few errors. For instance, more than 90% of the erroneous packets contain no more than 5 errors at Eb/No=7.5 dB. Moreover, as the channel quality decreases, the benefit of considering 5 errors instead of 3 errors increases, going from handling 87.3% of the error patterns for CRC-EC3 to 97.7% for CRC-EC5, at Eb/No=8 dB. In Table III, we show the average PSNR for the CRC-EC methods for 3 and 5 errors (CRC-EC3 and CRC-EC5), as compared to the error-free sequence and to the error concealment available in the FFmpeg decoder, comprising a deblocking filter and iterative motion vector search (*Deblock+MVS*) over a BLE channel of Eb/No=8 dB, for different H.265 encoded sequences. When the CRC-EC3 or CRC-EC5 cannot correct a packet, they fall back to the *Deblock+MVS* concealment method. In such channel conditions, the packet error rate is high (5% of the packets are corrupted), and it can be seen that the error concealment is limited and cannot handle such degradation. However, over the different sequences tested, CRC-EC5 can achieve average PSNR gains of 4.4 dB and 12.46 dB, compared to CRC-EC3 and the concealment approach, which respectively present an average PSNR loss of 1.57 dB compared to the intact sequence.

TABLE III: Average PSNR comparison for different H.265 encoded sequences and QPs over BLE channel with Eb/No=8 dB. The following are the decoding methods compared: error-free (intact) sequence, *Deblock+MVS* concealed sequence, CRC-EC with $N = 3$ and $N = 5$.

Sequence	QP	Intact	Conceal.	N=3	N=5
Crew (704x576)	37	34.69	29.85	33.90	34.62
	32	37.03	28.36	34.85	36.53
	27	39.19	28.23	35.58	38.90
	22	42.02	28.27	36.90	41.29
Ice (704x576)	37	36.35	23.12	33.24	35.31
	32	38.99	23.71	35.02	38.09
	27	41.38	22.59	36.59	38.23
	22	43.76	21.62	32.22	40.72
Mobcal (704x576)	37	30.96	23.07	29.10	30.85
	32	33.75	19.69	28.96	32.82
	27	36.53	18.05	22.91	33.72
	22	40.07	18.79	22.84	33.85
Average Δ Intact		——	-14.03	-6.01	-1.57

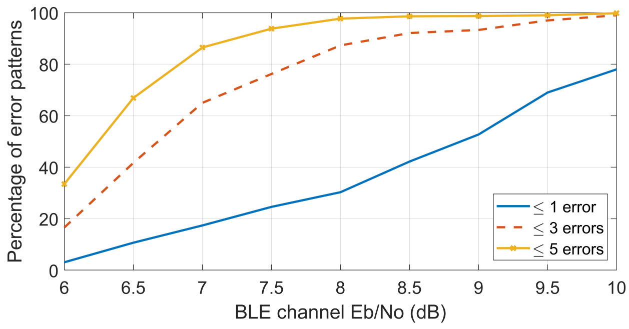

Fig. 5: Percentage of error patterns containing less than a determined number of errors, for simulated BLE channels. Tests are conducted on packets with payload length of 255 bytes

V. Conclusion and Perspectives

In this paper, we analyzed the properties of the CRC-EC candidate list for several errors and evaluated the applicability of multi-error CRC-EC to wireless environments such as 802.11p and BLE. We showed that increasing the number of errors considered led to enhanced error correction capabilities and PSNR gains, which are highly dependent on the channel under consideration. We showed very significant correction and PSNR gains when considering BLE. In future work, we will investigate an additional step allowing to sort the remaining candidates, after the decoding step, based on their visual quality, using deep learning to identify the best one.

References

[1] B. Chung and C. Yim, “Bi-sequential video error concealment method using adaptive homography-based registration,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1535–1549, 2020.

[2] F. Caron and S. Coulombe, “Video error correction using soft-output and hard-output maximum likelihood decoding applied to an H.264 baseline profile,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 7, pp. 1161–1174, 2013.

[3] P. Duhamel and M. Kieffer, *Joint source-channel decoding: A cross-layer perspective with applications in video broadcasting*. Academic Press, 2009.

[4] I. S. Association, “IEEE 802.11-2016, IEEE standard for local and metropolitan area networks—part 11: Wireless lan medium access control (MAC) and physical layer (PHY) specifications,” 2016.

[5] J. S. Sobolewski, “Cyclic redundancy check,” in *Encyclopedia of Computer Science*, 2003, pp. 476–479.

[6] R. Braden, D. Borman, and C. Partridge, “Computing the internet checksum,” *ACM SIGCOMM Computer Communication Review*, vol. 19, no. 2, pp. 86–94, 1989.

[7] E. Tsimbalo, X. Fafoutis, and R. J. Piechocki, “CRC error correction in IoT applications,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 361–369, 2016.

[8] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[9] R. Gallager, “Low-density parity-check codes,” *IRE Transactions on information theory*, vol. 8, no. 1, pp. 21–28, 1962.

[10] M. Collotta, G. Pau, T. Talty, and O. K. Tonguz, “Bluetooth 5: A concrete step forward toward the IoT,” *IEEE Communications Magazine*, vol. 56, no. 7, pp. 125–131, 2018.

[11] S. Shukla and N. W. Bergmann, “Single bit error correction implementation in CRC-16 on FPGA,” in *Proceedings. 2004 IEEE International Conference on Field-Programmable Technology (IEEE Cat. No. 04EX921)*. IEEE, 2004, pp. 319–322.

[12] A. Aiswarya and A. George, “Fixed latency serial transceiver with single bit error correction on FPGA,” in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*. IEEE, 2017, pp. 902–907.

[13] V. Boussard, S. Coulombe, F.-X. Coudoux, and P. Corlay, “Table-free multiple bit-error correction using the CRC syndrome,” *IEEE Access*, vol. 8, pp. 102 357–102 372, 2020.

[14] MathWorks, “802.11p packet error rate simulation for a vehicular channel,” 2018.

[15] MathWorks, “End-to-end bluetooth low energy phy simulation with rf impairments and corrections,” 2019.

[16] V. Boussard, F. Golaghazadeh, S. Coulombe, F.-X. Coudoux, and P. Corlay, “Robust H.264 video decoding using CRC-based single error correction and non-desynchronizing bits validation,” in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 1098–1102.

[17] I. T. Union, “ITU-T REC H.264 : Advanced video coding for generic audiovisual services,” 2019.

[18] I. T. Union, “ITU-T REC H.265 : High efficiency video coding,” 2019.

[19] F. Golaghazadeh, S. Coulombe, F.-X. Coudoux, and P. Corlay, “The impact of H.264 non-desynchronizing bits on visual quality and its application to robust video decoding,” in *2018 12th International Conference on Signal Processing and Communication Systems (ICSPCS)*. IEEE, 2018, pp. 1–7.

[20] K. Suhring, “H.265/HEVC HM reference software 16.20,” 2019, [Available] https://hevc.hhi.fraunhofer.de/.

[21] Y. Hou, J. Xu, W. Xiang, M. Ma, and J. Lei, “Near-optimal cross-layer forward error correction using raptor and RCPC codes for prioritized video transmission over wireless channels,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 9, pp. 2028–2040, 2017.

[22] S.-T. Chen, H.-T. Chiao, S.-Y. Chang, H.-M. Sun, and P.-S. Zeng, “An HD streaming system for WiFi multicast channels based on application-layer FEC,” in *2013 IEEE International Symposium on Consumer Electronics (ISCE)*. IEEE, 2013, pp. 85–86.

[23] H.-T. Chiao, S.-Y. Chang, K.-M. Li, Y.-T. Kuo, and M.-C. Tseng, “WiFi multicast streaming using AL-FEC inside the trains of high-speed rails,” in *IEEE international Symposium on Broadband Multimedia Systems and Broadcasting*. IEEE, 2012, pp. 1–6.

[24] Advanced Television Systems Committee, “ATSC standard: Signaling, delivery, synchronization, and error protection,” Doc. A/331:2021, 2021.