Supplementary Document: Video Coding Experiments

In this document, we describe the details of video coding experiments that led to the results presented in the submitted manuscripts.

1 VIDEO CODER AND TESTING SEQUENCES

We conduct all coding experiments on JVET 360° video testing sequence "Trolley" and "Chairlift" [1] using HEVC reference software (HM) [2] under JVET common test condition (CTC). "Trolley" is a stable 360° video shot by a fixed camera. It is represented in ERP format with 8192×4096 resolution (8K), 8 bits color depth (per color component) and 30 frames per second. "Chairlift" is a dynamic 360° video shot by a moving camera. It is also represented in ERP format with 8K resolution, but at 10 bits color depth. For PF and PF+ tiles, we use the low-delay-P mode with $IntraPeriod = -1$ meaning only the first frame is I frame and all following frames are P frames. For RI tiles, we use the intra mode with $IntraPeriod = 1$ meaning every frame is coded as I frame.

2 OPTIMIZING TILE SIZE

In tile-based video coding, the tile size affects both the video coding efficiency and transmission flexibility. Larger tile sizes will provide higher coding efficiency but it will also lead to more unused area outside the FoV. The bits for coding the unused area are wasted and do not provide any gain to the quality inside the FoV. Figure **??** shows an example of the actual FoV and the unused transmitted area for two different tile sizes. An early work explored this topic [4] and showed the optimal solution for 1080p and 4K videos. We have done a similar experiment for the 8K JVET testing videos.

In tile-based 360° video coding, the optimal tile size should minimize the total bit consumption of all needed tiles to cover the viewport averaged over all possible viewports for a fixed quality. Figure 1 illustrates the total bit rates needed when the tiles are coded in the inter-coding mode at a constant quantization parameter (QP) to cover a 90°×90° FoV for 5 different tile sizes. Generally, the number of tiles needed to cover different viewports differs depending on the viewport direction.

We found the 4 viewports with FoV centered on the equator achieve minimal bit rate consumption when using 22.5° tiles, slightly better than using a tile size of 11.25°. For the FoV faced to the top and bottom, there are relatively more wasted pixels in the boundary tiles. In this case, 11.25° tile size is slightly better than 22.5°. To simplify the system setting, we pick one tile size for the entire ERP instead of coding and transmitting different regions using 2 different tile sizes. Given that a smaller tile size offers more granularity in varying the sizes of RI and PF+, we have decided to choose 11.25° as our tile size, which contains 256 × 256 pixels for the 8K video sequence.

3 QUALITY-RATE FUNCTIONS FOR DIFFERENT CODED REGIONS

To get the quality-rate functions required for optimizing the rate allocation, we first assume the user's FoV does not change during the entire duration of the video sequence and did three coding experiments for fixed PF, PF+, and RI regions, separately. We then consider how to adjust the resulting Q-R functions to take into account of the dynamics of the FoV.

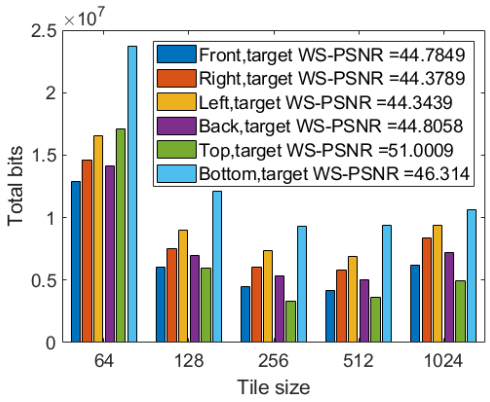

Figure 1: Total bit consumption to achieve the target quality (WS-PSNR) inside the FoV for 5 different side lengths of the square tile, i.e. 2.8125°, 5.725°, 11.25°, 22.5°, 45°, **which correspond to the side length 64, 128, 256, 512, 1024 pixels of square tiles for 8K resolution video. All the tiles are coded in the inter-coding mode except the first frame. A constant QP=30 was used. The resulting WS-PSNR in the FoV region are reported in the figure legends. Histograms in different colors are the results for FoV oriented to six directions, i.e. front, right, left, back, top and bottom. The results are for sequence "Trolley".**

3.1 Quality-Rate Function for the Predicted FoV region

To generate the Q-R functions for the PF region, we first code each tile video using the low-delay-P mode with a fixed QP throughput the entire sequence. We repeat this experiment using four QPs, i.e., 22, 27, 32, 37. Then, to generate the quality-rate curve for a FoV region, we determine the tiles needed to cover this FoV, and calculate the WS-PSNR for all the pixels inside the FoV and the total bits of all needed tiles to cover the FoV, for each QP. Figures 2(a) and 2(e) each shows six curves for six different FoV orientations. Here we assume the FoV size is 90° × 90°. The normalized bit rate is determined by dividing the total bits by 90° × 90°.

From the statistics of the user FoV behavior [3], more than 90% FoV centers are located in the range of equator±45°. Therefore, we generate the weighted average Q-R curves with a high weight (i.e., 0.8) to the 4 center FoV curves and a lower weight (i.e., 0.2) to the 2 polar FoV curves. The average Q-R curves are also shown in Figures 2(a) and 2(e). Finally, we approximate the weighted average Q-R curve by a logarithmic model:

$$Q_{PF}(R) = a_e + b_e \log R. \quad (1)$$

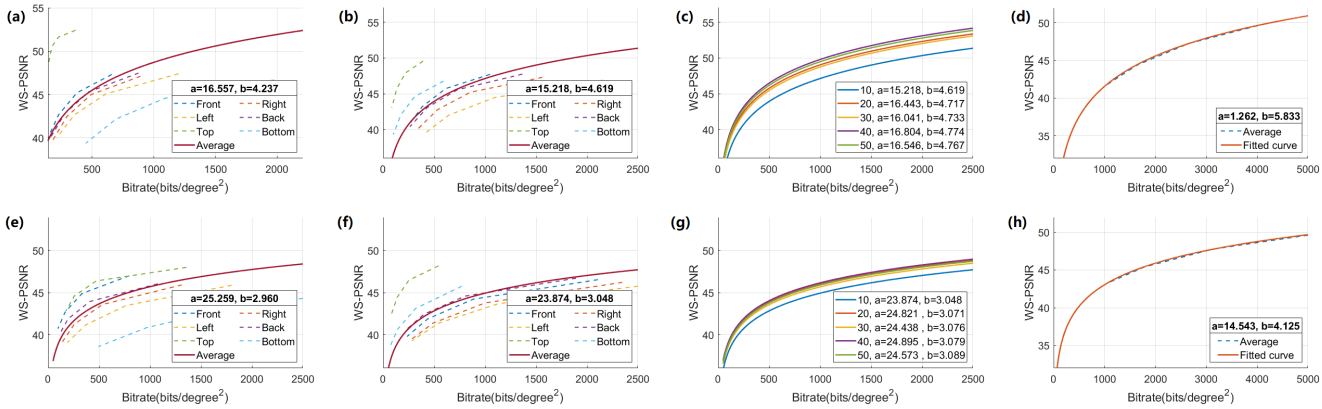

Figure 2: (a) - (d) are the results for Trolley video, (e) - (h) are results for Chairlift video. (a)(e): WS-PSNR vs. normalized rate for the predicted FoV region for six different viewing directions and the averaged WS-PSNR vs. normalized rate relation. (b)(f): WS-PSNR vs. normalized rate for the extended boundary outside the predicted FoV (PF+) region = 10° and the average.(c)(g): the averaged WS-PSNR vs. normalized rate when PF+ region size = 10°, 20°, 30°, 40°, 50°, respectively; (d)(h) WS-PSNR vs. normalized rate curves for RI region using intra-coding mode.

As indicated in the figure legends, the model parameters a_e and b_e are generally content-dependent. Because "Chairlift" video contains more dynamic motion, temporal prediction in inter-coding is more challenging, leading to lower average Q-R curves than those for "Trolly".

3.2 Quality-Rate Functions for the PF+ Region

As shown in Figure **??** in the manuscript, **Yixiang: you need to hard code the figure number** the PF+ region covers a border outside the PF region. The number of tiles needed to cover the PF+ region depends on the width of the border (in degree). In our experiments, we set border width to $10^\circ, 20^\circ, 30^\circ, 40^\circ, 50^\circ$ (10° means that the extended degree in each direction is 5°). For each candidate FoV orientation, we determine the WS-PSNR among pixels falling in the border region, and count the total number of bits used by the tiles needed to cover the PF+ region, for each target PF+ size in degree. The normalized rate is determined by dividing the total rate by the border size in square degree. For example, with FoV size of $90^\circ \times 90^\circ$, and border size of 10°, the border area is approximated by $100^\circ \times 100^\circ - 90^\circ \times 90^\circ$ The Q-R functions for different FoV orientations and the weighted average Q-R curve for the border width of 10° are shown in Figure 2(b) and 2(f). We find that the Q-R curves for different PF+ regions can also be approximated well by the logarithmic model:

$$Q_{PF+_i}(R) = a_{bi} + b_{bi} \log R, \tag{2}$$

where the model parameters depend on the PF+ border size. Figure 2(c) and 2(g) show the average Q-R curves for different border widths. Note that the coding efficiency is higher for a wider border due to the fact that fewer pixels in the coded PF+ tiles are wasted in such a case.

3.3 Quality-Rate Functions of the RI Region

The RI region is coded using the intra-mode. The quality is the average WS-PSNR of all pixels in a RI, while the rate is the total bits of all pixels in the RI normalized to the average spherical area represented by RIs in different locations on the ERP. Figure 2(d) and 2(h) show the average Q-R functions. Note that the Q-R function is the same regardless the RI size, because all the tiles in a RI are considered equally useful (with a probability to be viewed characterized by the hit rate of α_I). Again, this curve can be approximated well using a logarithmic function:

$$Q_I(R) = a_I + b_I \log R. \tag{3}$$

3.4 Rate-increase Factor due to Prolonged Time Lapse between the Coded Frame and the Reference Frame

The Q-R functions derived for the PF and PF+ regions in Section 3.1 and 3.2 assume the FoV and consequently the PF and PF+ regions do not change. In reality, a user will change the FoV over time so when coding a tile in the current PF or PF+ region, it may refer to the tile which is not updated in the previous frame. In this case, the bits needed to code this tile at a fixed QP depend on the time lapse τ since this tile was last coded. The rate increase factor $\rho(\tau)$ is defined as the ratio of the rate required for a given τ vs. the rate when $\tau = 1$.

In order to model $\rho(\tau)$, we code the testing videos using fixed QPs with different time lapses to measure the additional bits needed to achieve the same video quality. Figure 3 shows the measured $\rho(\tau)$ for $QP = 22, 27, 32, 37$ for τ. As shown in the figure, the rate-increase factor can be well fitted by a reverse exponential decay function:

$$\rho(\tau) = 1 + c * [1 - e^{-d*(\tau-1)}], \tag{4}$$

where the parameters c and d are content- and QP-dependent.

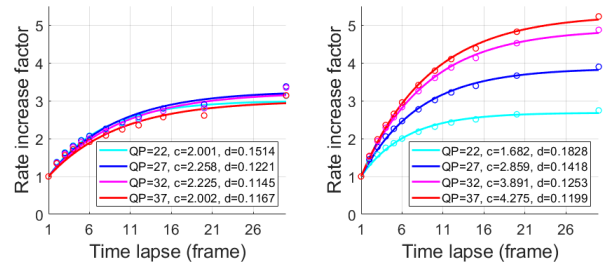

Figure 3: The rate-increase factor to maintain the same quality as a function of the time lapse between the inter-coded frame and the reference frame. Left: Trolley, fixed camera. Right: Chairlift, moving camera.

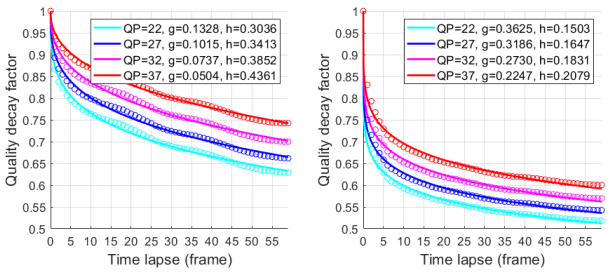

Figure 4: The quality-decay factor of pixels when time lapses. Trolley: fixed camera. Chairlift: moving camera.

3.5 Adjust Quality-Rate Functions for PF and PF+ regions

Given a QP, the actual rate required to code a tile inside the PF or PF+ region depends on the time lapse since this tile was last coded. The time lapse of each tile is spatially and temporal variant and depends on the FoV dynamics. To adapt the coding rates and the region sizes at the beginning of each segment, we adjust the Q-R functions derived in Section 3.1 and 3.2 as follows:

(1) Calculate the τ distribution of tiles in PF regions in the previous segment.
(2) Locate the quality and rate values for each of the 4 QP values on the original average Q-R functions in Figure 2.
(3) For the rate in each sample point, calculate the rate-increase factor for each τ value using equation (4) and hence the adjusted rate. Then determine the average rate among all possible τ's based on the distribution of τ. This will form a new Q-R point, where Q is the same as before, but R increased.
(4) Use the new Q-R points to fit a new average Q-R function.

The adjustment for the Q-R functions of PF+ region with variable region sizes can be done similarly.

4 QUALITY-DECAY FACTOR DUE TO FRAME COPY

Recall that the decay factor is the ratio of the quality of a tile that is not updated (hence replicated from the corresponding region when it is last coded) vs. the quality when it is last coded. The longer is the time lapse since this tile is last coded, the lower would be this ratio. In our experiment, we estimate the decay factor by doing the following simulation. In a video sequence, we code one frame at certain QPs. For each QP, we calculate the WS-PSNR and save it as WS-PSNR(0). For the following τ-th frame, we calculate the WS-PSNR between the coded first frame and the raw τ-th frame, represented by WS-PSNR(τ). The quality-decay factor is defined by $\kappa(\tau)$ = WS-PSNR(τ)/WS-PSNR(0). We repeat this experiment using each of the first 200 frames in each of the two videos as the initial frame, and determine the decay factor for the time lapse between 1 and 100. The average decay factors for all 200 samples is used to determine the average quality-decay function for each video. Results are shown in Figure 4. The quality-decay factor can be well fitted by a modified exponential decay model:

$$\kappa(\tau) = e^{-g*\tau^h}, \tag{5}$$

The parameters g and h are also content- and QP-dependent.

REFERENCES

[1] Jill Boyce, Elena Alshina, Adeel Abbas, and Yan Ye. 2017. JVET common test conditions and evaluation procedures for 360 video. *Joint Video Exploration Team of ITU-T SG* 16 (2017).
[2] IK Kim, K McCann, K Sugimoto, B Bross, WJ Han, and G Sullivan. 2014. High efficiency video coding (HEVC) test model 14 (HM 14) encoder description. Document: JCTVC-P1002. *JCT-VC, Jan* (2014).
[3] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. 2017. A dataset for exploring user behaviors in VR spherical video streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference*. ACM, 193–198.
[4] Mengbai Xiao, Chao Zhou, Yao Liu, and Songqing Chen. 2017. Optile: Toward optimal tiling in 360-degree video streaming. In *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 708–716.